The 14th Irish Conference on

Artificial Intelligence & Cognitive Science

AICS 2003

Proceedings

Trinity College Dublin
17-19 September 2003

Programme Chairs

Pádraig Cunningham, Tim Fernando, Carl Vogel

Programme Committee

Francisco Azuaje,	University of Ulster at Jordanstown
Derek Bridge,	University College Cork
Ruth Byrne,	Trinity College Dublin
Arthur Cater,	University College Dublin
Fintan Costello,	University College Dublin
Roddy Cowie,	Queen's University Belfast
Brian Crean,	Galway-Mayo Institute of Technology
Fred Cummins,	University College Dublin
Pádraig Cunningham,	Trinity College Dublin
Marc van Dongen,	University College Cork
Malachy Eaton,	University of Limerick
Tim Fernando,	Trinity College Dublin
Josephine Griffith,	National University of Ireland, Galway
Mark Keane,	University College Dublin
Nick Kushmerick,	University College Dublin
Saturnino Luz,	Trinity College Dublin
Paul Mc Kevitt,	University of Ulster at Magee
David McSherry,	University of Ulster at Coleraine
Mike McTear,	University of Ulster at Jordanstown
Michael Madden,	National University of Ireland, Galway
Fionn Murtagh,	Queen's University Belfast
Diarmuid O'Donoghue,	National University of Ireland, Maynooth
Greg O'Hare,	University College Dublin
Michael O'Neill,	University of Limerick
Colm O'Riordan,	National University of Ireland, Galway
Barry O'Sullivan,	University College Cork
Ronan Reilly,	National University of Ireland, Maynooth
Conor Ryan,	University of Limerick
Reinhard Schaeler,	University of Limerick
Barry Smyth,	University College Dublin
Humphrey Sorensen,	University College Cork
Richard Sutcliffe,	University of Limerick
Adrian Trenaman,	National University of Ireland, Maynooth
Josef van Genabith,	Dublin City University
Carl Vogel,	Trinity College Dublin
Ray Walshe,	Dublin City University
Andy Way,	Dublin City University
Adam Winstanley,	National University of Ireland, Maynooth

Additional Reviewers

Eva Maguire, Julie Berndsen, Brian Murphy, Lorraine Gilleece, Martin Emms, Mark O'Neill, Rowan Nairn, John Kelleher, Lorcan Coyle, Ann Devitt, John Dunnion, Jennifer Foster, Finlay Smyth.

Welcome

We welcome you to the 14th AICS, the Irish Conference on Artificial Intelligence and Cognitive Science in Trinity College Dublin. This is the second time for the AICS to be held in TCD. The 7th AICS was held here in 1994.

The AICS conferences have been a valuable forum for interaction between Irish researchers on Cognitive Science and AI over the years and we hope that this continues with the 2003 conference.

Well over forty papers were submitted for oral presentation, and with an international base. While only 15 oral presentations could be accepted, both oral and poster submissions are included in these proceedings and record a range of advances in Artificial Intelligence and Cognitive Science that are relevant to current activities distributed around Ireland, and obviously relevant to the corresponding global community of researchers.

The industrial session, which draws on the Irish community of firms that apply and develop results germane to this conference, represents the gamut from small to large in terms of corporate size, with operations located in Ireland, and clear interactive relevance to the subject matter at the root of this conference.

The primary themes in AI this year are constraint processing and collaborative filtering. Within Cognitive Science, natural language and theories of concepts dominate. Other issues also emerge. We think it is a diverse and stimulating programme

We would like to thank all who submitted papers to AICS 2003, and all who provided reviews, on the basis of which we put together the current programme. Our special thanks is due to John Loughrey for his invaluable help with the local organization.

Padraig Cunningham
Tim Fernando
Carl Vogel

Invited Speakers

Padhraic Smyth
University of California, Irvine
www.ics.uci.edu/~smyth/

Probabilistic Learning and Artificial Intelligence: A Review and Update

Abstract.
This talk will consist of three parts. In the first part of the talk I will discuss how probabilistic techniques have become increasingly important in artificial intelligence in recent years. The availability of large data sets, foundational advances in the theory of supervised and unsupervised learning, and conceptual contributions such as graphical models (or Bayesian networks), have all played an important role. In the second part of the talk I will discuss three specific applications from my research group based on probabilistic learning methods. The three applications are (1) clustering Web users based on their surfing patterns, (2) automated extraction of topic models from text, and (3) simultaneous clustering and alignment of sets of curves. In the third and final part of the talk I will briefly discuss new and emerging research directions in probabilistic learning.

Ruth Byrne
Trinity College Dublin
www.tcd.ie/Psychology/Ruth_Byrne/

Human Deductive Reasoning

Abstract.
One of the primary issues for cognitive scientists in modelling human reasoning is how to choose between the main alternative theories of human reasoning. These theories have each been experimentally corroborated and implemented in computer programs. They provide different perspectives on a second key issue, whether human reasoning depends on a single system, perhaps overlain with strategies and heuristics, or whether it depends on multiple systems. I will describe recent developments in the novel domain of reasoning from counterfactual conditionals, and discuss their implications for resolving these issues.

Table of Contents

Automatically converting Text Documents into XML

Shazia Akhtar[1], **Ronan G. Reilly**[2], **John Dunnion**[1]

Abstract. In this paper, we present a novel system which automatically converts text documents into XML by using machine-learning techniques. In the first phase, the system uses the Self-Organizing Map (SOM) algorithm to arrange marked-up documents on a two-dimensional map such that the documents similar in content appear closer to each other. In the second phase, it then uses the inductive learning algorithm C5.0 to automatically extract and apply markup information (in the form of rules) from the nearest SOM neighbours of an unmarked document. The system is designed to have an adaptive behaviour, so that once a document is marked-up into XML, it learns from its errors to improve accuracy. The resulting marked-up document is again categorized on the SOM. The results of our experiments with a number of document sets from different domains, indicate that our approach is practical.

1 INTRODUCTION

With the steady growth of World Wide Web, large amounts of information are available in electronic form. The electronic documents must be stored and managed in such a way that they should be easily accessible, portable, flexible and system independent. The adoption of XML as a standard for electronic publishing can provide a great opportunity for effective management and retrieval. XML was inspired by the limitations of HTML and the complexity of SGML. It provides key features such as extensibility, validation and structure. HTML uses a set of predefined tags but unlike HTML, XML tags can be generated according to the user's need. The XML tags can provide clues to distinguish ambiguous words in the corpus of documents and facilitate different applications to process encoded information properly. Search engines can use XML tags to exploit the logical structure of documents, which should improve search results and avoid irrelevant searches. Documents can often be quite long and only a small portion of the document can be relevant to a certain query in many cases. XML makes explicit the logical structure of documents therefore rather than searching the entire document, relevant portions of documents can be retrieved to provide more precise information. However, despite the general consensus on XML's benefits, we are still lacking large XML repositories and the search engines can use the XML structure and content for better retrieval only in the presence of vast amounts of XML data on the Web.

Manually converting collections of text documents into XML is not easy because it is tedious, expensive and time-consuming. For text documents to be efficiently and effectively converted into XML, the process of markup must be automated. Currently automatic XML markup is a significant challenge. Most systems that have been developed are limited to certain domains and require considerable human intervention. In addressing the need of more general automatic markup we present a novel hybrid system that produces tagged document collections by using two machine learning techniques, namely the Self-Organizing Map (SOM) algorithm [1], [2] and the inductive learning algorithm C5.0 [3], [4]. The process of automatic markup is based on the previously marked-up valid XML documents to be used as training data by the system.(A valid XML document is one which is well-formed and which has been validated against a DTD).

[1] Department of Computer Science, University College Dublin, Belfield, Dublin 4.
[2] Department of Computer Science, National University of Ireland Maynooth, Maynooth, Co Kildare.

2 SYSTEM ARCHITECTURE

The hybrid architecture of our system combines the SOM and C5.0 algorithms for the adaptive automatic markup of text documents into XML. The overall approach is shown in Figure 1.

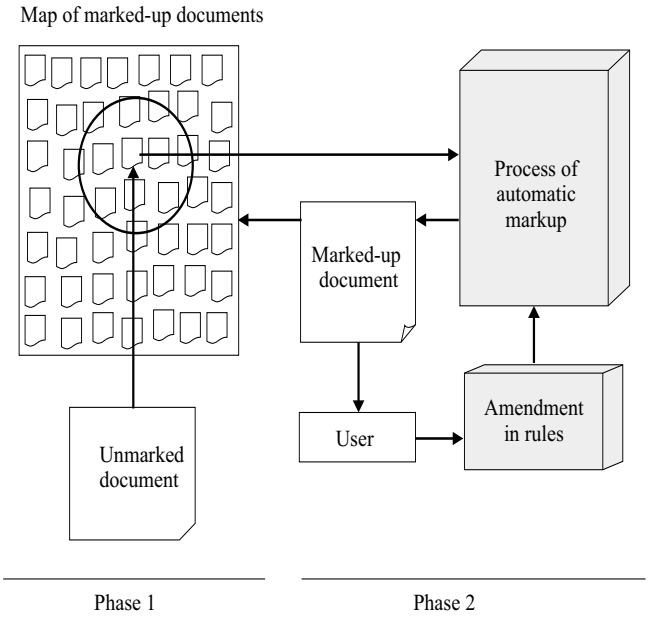


Figure 1. Architecture of the hybrid system. *Phase 1* deals with the formation of a Self-Organizing Map. *Phase 2* deals with the auto-tagging of text documents into XML by using the inductive learning algorithm C5.0

The first phase of our system deals with the formation of a map of a collection of marked-up documents, using the SOM [5] algorithm. SOM is a neural network-based unsupervised learning algorithm, which maps higher-dimensional statistical data onto a lower-dimensional grid or map such that similar documents appear close to each other on the map. The second phase of the system deals with the automatic markup of text documents and is implemented as an

independent XML markup system, which is described in section 3. Eventually these two phases will be combined to form an integrated hybrid system, which is described below.

Once a map has been formed, the system automatically extracts information from the SOM neighbours of an unmarked document in phase 2. This information is extracted in the form of rules by using an inductive learning algorithm. These rules together with text segmentation heuristics derived from the set of marked-up documents are used to markup the unmarked text document into XML. These two phases of the system are currently implemented independently but will eventually be linked together to form an integrated hybrid system. Phase 2, which is the focus of this paper, is currently implemented as an independent XML markup system and is described in section 3.

3 AUTOMATING THE PROCESS OF MARKUP

The automatic markup process (Phase 2 of the hybrid system) is shown in Figure 2. It has two main modules, a rule extraction module and a markup module.

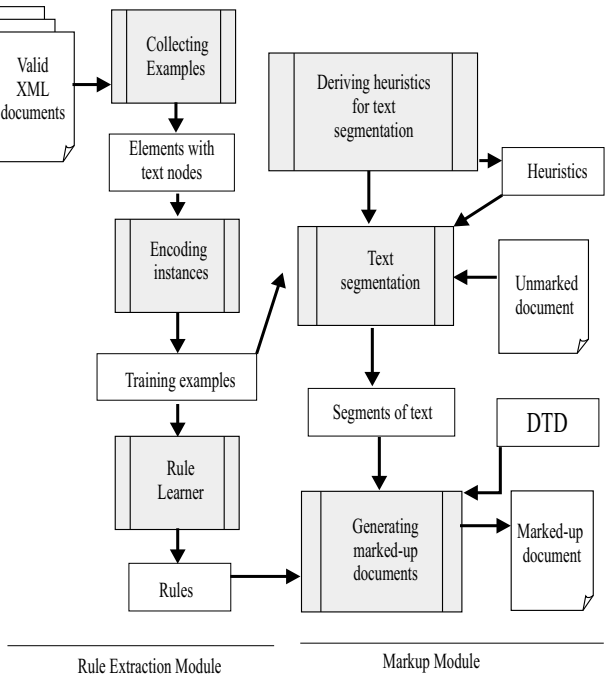


Figure 2. The Process of Automatic Markup

The rule extraction module learns rules from a collection of marked-up documents using an inductive learning approach [5]. In this module, training examples are collected from a set of valid XML documents. These documents should be from a specific domain and their markup should be valid and comply with the rules of a single Document Type Definition (DTD). An XML document can be represented as a tree-like structure with a root element and other nested elements. Only elements having text are considered appropriate for the automatic markup process. Each training instance corresponds to a leaf element containing text from the collection of marked-up documents. An example of a training instance from a collection of letters taken from The MacGreevy Archive [6], [7] is given below.

```
<DATE>11th October, 1940</DATE>
```

The texts enclosed between the start and end tags of all occurrences of each element are encoded using a fixed-width feature vector. These encoded instances are used subsequently for learning the rules. Thirty-one features, such as word count, character count, etc., are used to encode the training instances. The system pre-classifies the encoded instances by the tag name of the element. These pre-classified encoded instances are used by the system to learn classifiers for the elements with that tag name. The learned classifiers are later used in the process of markup. We have used the C5.0 learning algorithm to learn classifiers. C5.0 algorithm and its previous version C4.5 are the extension of Quinlans famous inductive learning ID3 algorithm [3]. These two algorithms i.e. ID3 and C5.0 are used for inducing classification models or decision trees from the data. The advantages of this learning algorithm are that it is very fast, it is not sensitive to missing features and it is incremental. C5.0 is best suited for our system because it is not sensitive to missing features. Our system deals with documents from different domains, so some of the features are not relevant to the documents of all domains. Sets of rules are generated in a given domain from a collection of marked-up documents and are used to markup the unmarked text documents from the same domain. A paraphrase of a rule learned from the set of letters is given below:

> *If the text segment contains the name of the month and up to six white space characters then it is specified as the text node of the* **DATE** *element.*

The second module converts an unmarked text document into a marked-up document. The unmarked document should be from the same domain as the documents used for learning the rules. For markup, the unmarked document is segmented into pieces of text using a variety of heuristics. These heuristics are derived from the set of training examples. By applying the rules of the DTD (The DTD provides us with a set of rules using a number of operators for sequence elements (’,’), repeated elements (’+’), optional elements (’?), and alternatives for recognizing the logical structure of a document), the rules extracted by using the C5.0 algorithm and the text segmentation heuristics, the hierarchical structure of the document is obtained and a marked-up version of the text document is generated.

The unmarked document produced by the system can be validated against the DTD by using any XML parser. However XML processors can only validate the syntax of an XML document. Since they cannot recognize the content of a document, a human expert is required to evaluate the accuracy of the markup process.

4 EXPERIMENTS

For our experiments, we have used collections of documents from a number of different domains. These include letters from The MacGreevy Archive, a database of employee records, Shakespearean plays [8], poems from the Early American Digital Archives [9] and scientific journal articles [10]. A letter taken from The MacGreevy Archive and marked-up by our system is shown in Figure. 3. Figure 3 shows 100% correct markup (of the letter) automatically produced by our system.

Another example taken from *A Midsummer Night’s Dream* automatically tagged by our system is shown in Figure. 4.

The underlined text, with the start and end tags of the element *STAGEDIR*, is not tagged by our system. This represents an error made by our system.

```
<?xml version="1.0"?>
<!DOCTYPE LETTER SYSTEM "letter.dtd">
<LETTER>
 <DATE> Friday 7th July 1936</DATE>

 <INSIDEADDRESS> 19, WESTGATE
  TERRACE,<LINEBREAK/>
REDCLFFESQURE.<LINEBREAK/>
S.W.10.<LINEBREAK/>
</INSIDEADDRESS>

 <SALUTATION> Dear Mr McGreevy
</SALUTATION>

 <BODY>
      <PARA> I heard from the lawyer today, and I am glad to
say that he will be able to make some payments in advance to
Miss H. Dowden -</PARA>
 </BODY>

<CLOSING> Yrs sincerely</CLOSING>

 <SIGNATURE> C.E. Harrison </SIGNATURE>
</LETTER>
```

Figure 3. A letter taken from The MacGreevy Archive automatically marked-up by our system

```
…
<SCENE>
    <TITLE> SCENE I. Athens.  The palace of THESEUS.
    </TITLE>
    <STAGEDIR> Enter THESEUS, HIPPOLYTA,
    PHILOSTRATE, and Attendanrs</STAGEDIR>
    <SPEECH>
       <SPEAKER>THESEUS</SPEAKER>
       <LINE>Now, fair Hippolyta, our nuptial hour</LINE>
        <LINE>Draws on a pace; four happy days bring
in</LINE>
       <LINE>Another moon: but, O, me thinks, how
       slow</LINE>
       <LINE>This old moon wanes!  she lingers my
       desires,</LINE>
       <LINE>Like to a step-dame or a dowager</LINE>
       <LINE>Long withering out a young man revenue. </LINE>
    </SPEECH>
    <SPEECH>
       <SPEAKER>HIPPOLYTA</SPEAKER>
       <LINE>Four days will quickly steep themselves in night;
       </LINE>
       <LINE>Four nights will quickly dream away the time;
       </LINE>
       <LINE>And then the moon, like to a silver bow</LINE>
       <LINE>New-bent in heaven, shall behold the night</LINE>
       <LINE>Of our solemnities</LINE>
       </SPEECH>
 …
```

Figure 4. Part of a scene taken from *A Midsummer Night's Dream* automatically tagged by our system

All the documents sets used in our experiments except the scientific journal articles were tagged by applying the rules extracted by using the C5.0 algorithm, the text segmentation heuristics and the rules of the appropriate DTD. For the scientific journal articles we have used additional heuristics devised specifically for this domain. We hope that these heuristics can be used effectively for articles from most journals. The marked-up journal articles used as training documents for our experiments were downloaded from the World Wide Web [10] along with the DTD (article.dtd) devised for these articles. From the same site, the HTML versions of articles were downloaded, converted to text files and automatically marked-up into XML by our system. The XML DTD used for these articles is complicated and requires the presence of another DTD (biblist.dtd) devised for references and bibliographies. For the automatic markup of articles, currently we only consider those elements of DTD that describe different sections of the article for example, title, author name, author affiliation, headings, paragraphs, references, etc. We have ignored the elements embedded in the text containing elements. These include the elements representing physical representation of different sections of the articles, e.g. `<b>`, `<i>` etc. Part of a scientific journal article automatically marked-up by our system is shown in Figure. 5.

Again, our system did not markup the underlined text with start and end tag of *title* and *orgName*. Although the system makes some mistakes, it still works reasonably well with our domain-specific heuristics and automatically marks up most of the sections of the journal articles.

5 EVALUATION

We have used three performance measures to evaluate the performance of our system. These measures are:

- The percentage of marked-up elements correctly determined by the system
- The percentage of marked-up elements incorrectly determined by the system
- The percentage of marked-up elements not determined by the system

When describing the accuracy of our system, we use the first of these measures, i.e. the percentage of the marked-up elements correctly determined by the system. Evaluation of the performance of our system for letters (from The MacGreevy Archive) indicates that it achieves an accuracy of 96%. For the Shakespearean plays, our system achieves 92% accuracy and for the poems taken from the Early American Digital Archives, it achieves 96% accuracy. For the scientific journal articles, the accuracy of markup process is 97%.

6 CONCLUSION

We have described a novel approach towards automatic markup of text docuements into XML. Our system uses Self-Organizing Map (SOM) algorithm and the inductive learning algorithm C5.0 to produce XML markup of unmarked documents. Various experiments with our system show that our approach provides promising results.

```
<?xml version="1.0"?>
<!DOCTYPE article SYSTEM article.dtd">
<article>
 <front>
    <docCiteAs>&amp;nbsp;MRS Internet J. Nitride Semicond.
                Res.3, 14.</docCiteAs>
    <cpyrt>&amp;nbsp;1999 The Materials Research Society</cpyrt>
    <title>Surface Morphology of MBE-grown GaN on GaAs(001)
      as Function of the N/Ga-ratio</title>
    <Authors>
      <auth><pn>O. Zseb&amp;ouml;k</pn></auth>
      <auth> <pn>J.V. Thordson</pn> </auth>
      <auth><pn>T.G. Andersson</pn></auth>
      <aff>
        <orgName>Chalmers University of Technology
        </orgName>
      </aff>
    </authors>
    <history><date>Tuesday, June 23, 1998</date></history>
    <history><date>Monday, August 24, 1998</date></history>
    <abstract>
      <p>Molecular beam epitaxy growth utilising an RF-plasma
nitrogen source was used to study surface reconstruction and surface
morphology of GaN on GaAs (001) at 580 &amp;deg;C. While both the
nitrogen flow and plasma excitation power were constant, the grown
layers were characterised as a function of Ga-flux. In the initial growth
stage a (3x3) surface reconstruction was observed. This surface
 ....
```

Figure 5. Part of a scientific journal article automatically marked-up by our system

The functionality of our system can be used to produce large document collections in XML These document collections can be used by many applications.

Currently, we are trying to figure out the points, which can improve the accuracy of markup produced by our system.

ACKNOWLEDGEMENTS

The support of the Informatics Research Initiative of Enterprise Ireland is gratefully acknowledged. The work was funded under grant PRP/00/INF/06.

REFERENCES

[1] T. Kohonen, Exploration of very large databases by self-organizing maps, *In Proceedings of ICNN'97, International Conference on Neural Networks*, PL1-PL6. IEEE Service Center: Piscataway, NJ (1997a)

[2] T. Kohonen, *Self-Organizing Maps*, Springer Series in Information Science, 1997b.

[3] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kauffmann, Los Altos, CA, 1993.

[4] J. R. Quinlan, Data Mining Tools See5 and C5.0, *[http://www.rulequest.com/see5-info.html]*, 2002.

[5] T. M. Mitchell, *Machine Learning* McGraw-Hill, NewYork, NY, 1997.

[6] S. Schreibman, The MacGreevy Archive, *[http://www.ucd.ie/ cosei/archive.html]*, 1998.

[7] S. Schreibman, The MacGreevy Archive, *[http://jafferson.villiage.Virginia.edu/-macgreevy]*, 2000.

[8] J. Bosak, The Plays of Shakespeare in XML, *[http://www.oasis-open.org/cover/bosakShakespeare200.html]*, 1999.

[9] S. Schreibman, Early American Digital Archives, Hosted by Maryland Institute of Technology *[http://www.mith.umd.edu]*, 2003.

[10] E. Hellman, D. Ephron, M. Poindexter, Openly Informatics Inc., *[http://www.openly.com/efirst]*, (1999-2000).

A Formal Specification of a Multi-Agent Trading System

Elaine Barrett[1] and **Sharon Flynn**[2]

Abstract. This paper considers how formal methods might be useful and how they may be applied to the area of agent-based systems. The main aim of this paper is to investigate the application of formal methods to multiagent systems through a multiagent trading system case study. We present a brief overview of formal methods, some background information on multiagent systems as well as information on agent communication and interaction. A brief explanation of the hypothetical multiagent trading system case study is given. Previously, there have been some attempts to model agents, however little attention has been paid to formally modelling agent communication and cooperation. We plan on using the Z specification language to provide a formal model of individual agents in isolation. Further developments of this work are to show that LOTOS is a useful method of specifying communication, cooperation and negotiation between multiple agents within the trading environment.

1 INTRODUCTION

This paper deals with an initial attempt to write a formal specification of a multiagent system. The multiagent system that is used is an agent trading system. The primary problem that we hope to address is to specify how the agents cooperate, communicate and negotiate within their environment. In approaching this problem, we plan on using the Z specification language to specify agents in isolation and the knowledge needed by each agent as well as their objectives. LOTOS is the language we use in order to specify the communication and interactions between trading agents.

The paper is structured as follows. The second section briefly mentions related research. The third section describes formal methods and briefly discusses the benefits of formalising systems. Agents and multiagent systems are outlined in section four. Section five gives an overview of the specification languages, Z and LOTOS. The multiagent trading system case study is introduced in section six. An informal specification of this system is briefly outlined. We describe how the Z and LOTOS languages are applied to the case study. Section seven compares our approach to other research. Finally the paper finishes with some conclusions about the case study and the specification languages used as well as future work that is proposed.

2 RELATED RESEARCH

There have been, to date, a few attempts to apply formal methods to agent-based systems. Luck and d'Inverno [15] use the Z specification language to give an account of agent systems. The framework provides unambiguous meanings for concepts and terms, something which is recognisably lacking in current agent research. Other approaches involve using modal and multi-modal logics to model agents, to provide both an internal agent language to be used by an agent in its own reasoning, and external metalanguages to specify, design and verify the behavioural properties of agents [5].

Although there has been some attempt to model agents, little attention has so far been paid to formally modelling communication and cooperation between agents. We suggest that a formal process algebra such as LOTOS would be a suitable formal method to use for this purpose.

Misra [16] has developed an interesting programming model called Seuss. Misra views a multiprogram as a set of actions. Each action deals with an aspect of the systems functionality. His work incorporates ideas from: serialisability and atomicity in databases, Cooperative Problem Solving (CPS), input/output automata, notions of objects and inheritance and finally temporal logic of actions. Peyton Jones et al. [17] apply formal semantics and functional programming to the area of financial and insurance contracts. Their design is presented as a combinator library embedded in Haskell. They define a carefully chosen set of combinators, and through an extended sequence of examples in Haskell they illustrate that these combinators can be used to describe a wide variety of contracts.

3 FORMAL METHODS

Formal Methods refer to the use of mathematical tools, techniques and notations in Computing Science, with a view to providing a theoretical underpinning to support proofs of properties and proofs of correctness. Formal methods have been demonstrated to result in systems of the highest integrity. Formal methods not only refers to the specification, verification and refinement languages, it also involves both formal reasoning about the specification as well as informal reasoning about the relationship between a formal model and a real world problem [20].

One of the more successful ways in which Formal Methods can be used is to model a system (software or hardware). Once a formal model, i.e. using a formal language, has been described, it can be used to prove properties of the system, or to develop/refine that system.

Based on his experience of using formal methods while working on an industrial project, Hall [9] found that formal methods helped identify errors early on in the project. He believes that formal methods force the specifier to think very hard about the proposed system. He also states that formal methods can help clarify the

[1] Department of IT, National University of Ireland, Galway, Ireland, email: elaine.barrett@nuigalway.ie
[2] Department of IT, National University of Ireland, Galway, Ireland, email: sharon.flynn@nuigalway.ie

system and help clients understand what they are buying.

Pfleeger and Hatton's [18] paper is a report on an investigation carried out on the effects of using Formal Methods to develop an air-traffic-control information system. Their aim was to show how Formal Methods influenced code quality. The result of their study revealed significant evidence of high code quality. They felt that formal specifications produced small and independent modules that led to clear-cut and uncomplicated unit testing. They concluded that formal design combined with other techniques resulted in highly reliable code.

4 MULTIAGENT SYSTEMS

Agent technology is an exciting and rapidly growing area of computing science. Agents have the potential to resolve problems that have been beyond the scope of automation, problems that very often have no existing knowledge or technology that is capable of solving them. A key advantage of agents is their ability to unravel problems in an easier, faster and cheaper way [23].

To date there is much dispute and disagreement over an agreed universal definition to describe agents. However Wooldridge presents the generally accepted definition of an agent as a computer system that is located in an environment and is capable of autonomous action in order to meet its design objectives [22].

Agents are autonomous and should operate without direct human intervention. They have a certain element of control over their state and actions. Agents are capable of interacting with other agents. They can be reactive, meaning that they can observe and understand their environment and react to changes that may occur. They may also display goal-directed actions by taking the initiative. Agents continuously run processes actively in the forefront or passively in the background [10].

Generally a multiagent system contains two or more agents, which have the ability to interact with each other and their environment. Each agent's expertise and knowledge of the environment may differ from another agents. A multiagent system is characteristically a distributed system with a number of different components. Each agent within the multiagent system is an independent problem-solving agent. These agents join together to form a coherent system. It is essential for agents to cooperate so that they can interact effectively preventing goal duplication, which may result in obstructing other agents achieving their goals [4].

4.1 Agent Characteristics

Communication. Agent communication protocols allow agents to exchange and understand messages. It is necessary for agents to communicate in order to achieve their goals. Communication allows agents to synchronize their actions and behaviour. Agents can communicate to exploit and ensure coordination and cooperation. An agent can also influence other agents to take on its goals [11]. Agent communication languages are the standard means of exchanging messages. The knowledge query and manipulation language (KQML) is a well-known agent communication language. It is a language and protocol for exchanging information and knowledge. The FIPA agent communication language is also based on the speech act theory. The syntax is quite similar to KQML. Each message holds a set of message elements. With FIPA-ACL, the communicative acts are described in both a narrative form and a formal semantics [7, 8].

Interaction. Agent interaction protocols allow agents to have conversations with each other. They may need models of each other in order to interact and to decide on their success and failure [22]. Agent interaction is a central part of multiagent systems. Interactions are revealed in different ways - including cooperation, coordination and collaboration. However [12] states that the most powerful means for managing inter-agent dependencies is negotiation.

Negotiation. Agent negotiation is a procedure where agents communicate with one another and attempt to reach an agreement on a particular goal or objective of shared interest. Negotiation protocols are a set of rules that regulate negotiation. Agents must agree on a negotiation protocol before negotiation can commence [12, 14].

5 FORMAL SPECIFICATION LANGUAGES

We will now briefly look at the two specification languages that will be applied to our multiagent trading system:

5.1 Z Specification Language

The Z specification language [19] is an example of a model-oriented approach to specification. Z was made an international standard by ISO in 2002. Z involves the construction of a model of the concept to be described, taking advantage of available mathematical tools. The associated operations of the concept are then specified with respect to the particular model which has been used. Z has, as its mathematical basis, familiar mathematical concepts and notations such as set theory and first order predicate logic.

Z has been used previously with some success to describe individual agents in a more general setting [6, 15]. We decided on using Z to model the agents in the trading system because it provides a clear and straightforward notation which permits the precise description of our agent properties. The result is an accessible specification which can be easily read by practitioners in Formal Methods and Artificial Intelligence, as well as by software engineers.

5.2 LOTOS Specification Language

LOTOS stands for Language of Temporal Ordering Specifications and is a specification language that was developed within the ISO for the formal specification of OSI systems. However it can also be applied to synchronised and distributed systems. LOTOS is an algebraic specification language, which consists of two main parts: an abstract data type language such as ACT ONE and also process algebra such as CCS and CSP [21].

The aim of LOTOS was to help specifiers who need precision, conciseness and clarity, it is also intended to assist implementers who need clear guidelines as to what to build and also testers, who need to have implementation options precisely defined [2].

In LOTOS, a system is perceived as a collection of processes. A process is a unit that is capable of executing internal and unobservable actions. Processes interact, cooperate and exchange messages

among each other as well as with their environment. Events symbolise a synchronisation between processes. When an event is offered it gives the process an opportunity to participate and get involved in the event. Two or more processes are necessary in order for an event to occur. The interaction of processes occurs through event gates. A process interacts with its environment through these gates [21].

LOTOS may be useful in order to specify agent communication and interaction in the multiagent trading system. It can be applicable to distributed systems. LOTOS was chosen because of its application in areas where communication is prevalent. It is also somewhat useful for the verification and validation of specifications [21].

Logrippo et al. [13] present a paper which investigates the usefulness of LOTOS in an industrial environment. They found LOTOS to be a powerful and expressive specification language of protocols and distributed systems. Due to the formal semantics of LOTOS, they perceived the specifications as being precise and unambiguous as well as helping to identify inconsistencies and incompleteness in the requirements.

6 APPLYING FORMAL METHODS TO A MULTIAGENT TRADING SYSTEM

The hypothetical case study that we use to apply formal methods is a multiagent trading system. This multiagent trading system contains a set of agents. The agents can buy, sell or trade goods with each other on behalf of the agent's owner or user. The trading system is similar to that of an auction. Agent communication and negotiation are the major characteristics of the system. Agent are required to negotiate in order to sell and acquire products and to maximise their profits.

We decided on using the Z specification to specify agents in isolation and the knowledge needed by each agent. Z has been used previously with success to describe individual agents. The reason we chose to use Z to model the agents in our trading system is because Z provides a clear and straightforward notation which allows the precise description of our agent properties. Having considered the behaviour of the agents using the formal specification language Z, we then need to address agent communication and interaction. We use another specification language LOTOS to specify agent negotiation and the interactions between agents in the trading system.

Agent negotiation plays a large part in the trading system environment. Agents need to negotiate in order to buy and sell their products. To negotiate, agents must communicate their positions, which may conflict. They must then attempt to make an agreement by making allowances or perhaps taking an alternative route.

Agents may be required to cooperate and collaborate with other agents in order to achieve their goal of buying the required quantity of products.

Initially we need to identify an agents' knowledge/beliefs. goals/desires and its intentions. Agent knowledge/beliefs refer to information the agent has about the world. This knowledge may be incorrect or incomplete [6]. The agent knowledge identified in this case study include knowledge of:

- the environment;
- how each agent communicates;
- protocols an agent uses to negotiate;
- how to sell a quantity of products including knowledge of the sale product name, product reference number and the quantity for sale;
- how to buy a quantity of products including knowledge of the required product name and the quantity required by the user;
- the set minimum selling price per sale product;
- the set maximum buying price per product required;

Every agent possesses a set of goals. The agent goals identified in our case study include:

- the agent is required to maximise profit;
- the agent needs to buy products at a price less than or equal to the maximum buying price that is specified by the user;
- similarly the agent needs to sell products at a price greater than or equal to the minimum selling price that is stated explicitly by the user;
- all agents need to trade or collaborate in order to acquire the required products.

An agents' chosen goals are its intentions. The agent should continue to try to achieve an intention until it believes that the intention is satisfied or is no longer attainable [6].

6.1 Informal Specification

An informal specification acts as an preliminary stage in the development of a more formal specification. We informally describe the behaviour of the multiagent trading system that we intend to specify.

6.1.1 Buying Agent

An agent that requires to buy a quantity of products has four trading options:

1. the agent can submit an offer,
2. the agent may trade a required quantity of products for a quantity of products that the agent needs to sell,
3. the agent may collaborate with another agent in order to acquire the necessary required products,
4. or the agent may opt to withdraw from bidding.

Regardless of the trading option that an agent chooses, the offer or value of the trade offer needs to be validated. We need to validate the offer so that the maximum specified buying price is not exceeded. The bid, collaboration offer or trade offer can either be accepted or rejected by the selling agent.

6.1.2 Selling Agent

Selling agents firstly declare a quantity of products for sale. The agent accepts offers from other agents. The highest offer is determined. The highest bid needs to be validated. The bid must be greater than the specified minimum selling price.

6.2 Formal Specification

We use the Z specification language to specify single agents in isolation. We need to model the knowledge required by each agent in order to begin reasoning on whether or not they should buy or sell a product.

We begin the specification by identifying the basic type definitions. In our Z specification we declare the three following types:

[PRODUCT, BID, TIME]

The easiest way to define an object is to declare it. If the object is a given set or a basic type, its name is written between square brackets as illustrated above [3]. The type *PRODUCT* is the basic type for products and represents the set of all products. Similarly *BID* and *TIME* are basic types for bids and time and represent the set of all bids and time limits.

The full specification is explained in greater detail in [1].

The first schema of the specification is the state schema *saleProductList*.

saleProductList

$$saleList : \mathbb{P}\, PRODUCT$$
$$saleMinPrice : PRODUCT \pfun BID$$
$$saleResTime : PRODUCT \pfun TIME$$

$$\mathrm{dom}\, saleMinPrice = saleList$$
$$\mathrm{dom}\, saleResTime = saleList$$

The state of the model is described by *saleList* which consists of a set of products and the relations *saleMinPrice* and *saleResTime*. The variable *saleList* is a set of products. The relations *saleMinPrice* and *saleResTime* are elements of *saleList*. Each product for sale has an associated minimum sale price and a time limit for sale.

The initial state of the system must be provided in order to demonstrate that at least one valid state exists. The initial state is always referred to as the after state. The notation '$\emptyset$' denotes the empty set of products, for example *saleList'* = $\emptyset$ contains no products:

InitialsaleProductList

$$saleProductList'$$

$$saleList' = \emptyset$$
$$saleMinPrice' = \emptyset$$
$$saleResPrice' = \emptyset$$

The first operation that we specify is *AddSaleProduct*, this full operation comprises of two subcases *addSaleProductOk* and *saleProductInList*. This is described by the disjunction:

$$AddSaleProduct \mathrel{\widehat=} addSaleProductOk \lor addSaleProductError$$

We need to model the first subcase *addSaleProductOk* of the *AddSaleProduct* operation. This subcase allows a new product to be added to the agents list of products that need to be sold:

addSaleProductOk

$$\Delta saleProductList$$
$$p? : PRODUCT$$
$$time? : TIME$$
$$bid? : BID$$
$$r? : REPORT$$

$$p? \notin saleList$$
$$saleList' = saleList \cup \{p?\}$$
$$saleMinPrice' = saleMinPrice \cup \{(p? \mapsto bid?)\}$$
$$saleResTime' = saleResTime \cup \{(p? \mapsto time?)\}$$
$$r! = product\_on\_list$$

In the above schema $\Delta saleProductList$ indicates that there is a change in the state of the list and that the state variables *saleList*, *saleMinPrice* and *saleResTime*, along with their invariants are included in the new schema, *addSaleProductOk*.

p? is an input that represents a product, *time?* is an input that represents time and *bid?* represents an input, bid. Before a new product is added to the list, the product *p?* must not already be an element of *saleList*.

saleList' = *saleList* $\cup \{p?\}$ specifies that the new product p? (which is not already an element of *saleList*) is added to the new updated list *saleList'*. The associated minimum selling price and the time limit is added to *saleMinPrice* and *saleResTime* respectively. When this is successfully completed a report *r!* is outputted indicating the successful addition of the new product to the list.

The second subcase of the operation is the *addSaleProductError* subcase. This subcase is when a product has already been previously added to the list. It is described as follows:

addSaleProductError

$$\Xi saleproductList$$
$$p? : PRODUCT$$
$$time? : TIME$$
$$bid? : BID$$
$$r? : REPORT$$

$$p? \in saleList$$
$$r! = already\_in\_list$$

The declaration $\Xi saleProductList$ says that the before and after states with their variables and invariants are included in the schema and the state does not change. The specification checks that *p?* is an element of *saleList*.

The next operation that we demonstrate is *BidForProduct*, this full operation comprises of the subcases *checkReceivingBid* and *ExceededTimeLimit*. This operation is described by the disjunction:

$$BidForProduct \mathrel{\widehat=} checkReceivingBid \lor ExceededTimeLimit$$

The first subcase *checkReceivingBid* checks each received bid:

checkReceivingBid

$$\Xi saleProductList$$
$$p? : PRODUCT$$
$$newbid? : BID$$
$$time? : TIME$$
$$r? : REPORT$$

$$p? \in saleList$$
$$newbid? \geq saleMinPrice(p?)$$
$$time? \leq saleResTime(p?)$$
$$r! = success$$

The ΞsaleProductList says that the before and after states with their variables and invariants are included in the schema and the state does not change. As previously, *p?* is an input of type *PRODUCT*, *newbid?* is of type *BID* and *time?* is of type *TIME*.

The receiving bid is deemed valid if the bid is greater than the specified minimum selling price and if the time limit has not been exceeded. The bid must also be for a product that is on the selling list.

The second subcase *ExceededTimeLimit* checks the time limit to ensure that the limit is not exceeded.

Other operations included in the specification include: *buyProductList*, *InitialbuyProductList*, *addBuyProductOk*, *addbuyProductError*, *removeBuyProductOk* and *removeSaleProductOk*. These operations are included in [1].

Having considered the behaviour of agents in isolation using the formal approach of Z, we can now begin to address communication and negotiation between agents. We use the specification language LOTOS to formally specify the systems interactions and negotiation. The first process in the specification is the *TradingSystem* process. The process can either follow the behaviour specified by the process *BuyProduct* or the process *SellProduct*:

process TradingSystem [decideOffer, validate, valid, invalid, sendReq, reply, sendOffer, offerAccepted, offerRejected, requiredProd, tradeProd](prd_id:prd_list, bid:amount, value:amount,agent_id:agent_list, tradeValue:amount, reqValue:amount): noexit :=
(
BuyProduct [decideOffer, validate, valid, invalid, sendReq, reply, sendOffer, offerAccepted, offerRejected, requiredProd, tradeProd](prd_id, bid, value,agent_id, tradeValue, reqValue)
[]
SellProduct [announce, bid, send, validate, accepted, rejected, sendReq, withdraw](prd_id:prd_list, bid:amount,value:amount, agent_id:agent_list,tradeValue:amount, reqValue:amount))

≫
TradingSystem [decideOffer, validate, valid, invalid, sendReq, reply, sendOffer, offerAccepted, offerRejected, requiredProd, tradeProd](prd_id, bid, value, agent_id, tradeValue, reqValue)
endproc

Process TradingSystem has the following events; *decideOffer, validate, valid, invalid, sendReq, reply, sendOffer, offerAccepted, offerRejected, requiredProd, tradeProd*. The interaction of processes occur through these event gates. *prd_id:prd_list, bid:amount, value:amount, agent_id:agent_list, tradeValue:amount, reqValue:amount* are parameters of the TradingSystem process. The parameter *agent_id:agent_list* is a list of agent id's and *prd_id:prd_list* is a list of the product id's. *noexit* indicates that the process is non-terminating.

The choice operator '[]' denotes the choice between two processes: *BuyProduct* or *SellProduct*. The enable operator '≫' means that on successful termination of *BuyProduct* and *SellProduct* the process *TradingSystem* is enabled. If *BuyProduct* or *SellProduct* do not terminate successfully then the process *TradingSystem* will not be enabled.

The *buyProduct* process is as follows:

process BuyProduct [decideOffer, validate, valid, invalid, sendReq, reply, sendOffer,offerAccepted, offerRejected,requiredProd, tradeProd](prd_id:prd_list, bid:amount,value:amount, agent_id:agent_list,tradeValue:amount,reqValue:amount) : exit :=
(
Trade [requiredProd, tradeProd, bid, send, validate, accepted, rejected, sendReq](prd_id:prd_list, tradeValue:amount, reqValue:amount, agent_id:agent_list)
[]
Collaboration [decideOffer, validate, valid, invalid, sendReq, reply, sendOffer, offerAccepted, offerRejected]
(prd_id:prd_list,bid:amount, agent_id:agent_list)
[]
MakeOffer [decideOffer, validate, invalid, valid, sendOffer, offerAccepted,offerRejected](prd_id:prd_list, bid:amount, agent_id:agent_list)
[]
withdraw; exit;
≫
BuyProduct [decideOffer, validate, valid, invalid, sendReq, reply, sendOffer,offerAccepted, offerRejected, requiredProd, tradeProd]
)
endproc

In the process *BuyProduct*, there are four choices:

1. product trading
2. collaborating
3. making an offer or
4. the agent can withdraw from bidding and exit the process.

The process *Trade* specifies the behaviour when an agent decides to trade a quantity of products in order to acquire the products that it requires. Firstly, the agent decides on the product that it requires. Values are attached to identify the id of the product required, the quantity required and the value of the quantity required. Next the agent must decide on the product that it intends on using in the trade. Again, values are attached to identify the product, the quantity and the value of the quantity. The trade value cannot exceed the maximum price specified. The value can be deemed to be either valid or invalid. If the value is invalid the agent can choose another trading option or it has the option of withdrawing and exiting the process. If the value is valid a trade request is sent to the selling agent. The request can either be accepted or rejected. If the request is rejected, the agent can try another trade or it may opt to exit the process. If the request is accepted the buying agent successfully obtains the required quantity of products.

The process *Collaboration* specifies the behaviour when an agent decides to collaborate with another agent in order to acquire a quantity of a product that they may require. The bid must be validated. If valid, a request is sent to other agents in the *agent_list* asking for an agent to collaborate and join bids in order to acquire a quantity of a product. If an agent agrees, they send the quantity that they require and the amount they are willing to pay for that amount. An offer is made to the selling agent. If the bid is accepted the agent exits the process. If the bid is denied the agent can try another collaboration request or it can withdraw from collaborating.

The process *MakeOffer* validates a bid made by the buying agent. The bid must be less than the maximum bid specified. If the bid

is deemed invalid, the agent can withdraw from bidding or else the BuyProduct process is called and the agent can rebid for same product or another product. On the other hand, if the bid is deemed valid an offer is made.

7 COMPARISONS

Many similarities as well as differences are noticeable when comparing our approach to Misra's work on Seuss [16]. We feel that while both basic program structures are comparable, the LOTOS structure seems to be easier to apply and more precise than Misra's approach. Misra's model does not have processes as its basic concept while processes are a central component to the LOTOS specification language. A notable feature of LOTOS is its handling of concurrency while Misra's model has no in-built concurrency. Also Seuss has no specific communication or synchronisation mechanism. LOTOS has specific synchronisation operators as well as parallel operators that allow behaviours to unfold independently in parallel.

While researching formal methods we felt that there was a noticeable lack of non-theoretical material describing the application of formal specification languages. Much of the material on LOTOS is applied to areas of networks and communication protocols. Our aim is to bridge this gap by applying specification languages to a real-world scenario of trading systems. Compared to other papers our specification is quite high-level, whereas, Wooldridges' and Jennings' paper on Cooperative Problem Solving [24] formalises a model at a much lower level and can seem mathematically and logically complex to a formal methods novice. They express their model as a theory in quantified multi-modal logic.

We also found that some papers such as [5, 6, 24] focus solely on providing abstract formal models of their systems. They specify non-specific agent beliefs, goals and plans. We decided to avoid this approach and concentrate solely on applying formal methods to an actual system with actual knowledge, goals and intentions.

8 CONCLUSIONS AND FUTURE WORK

The fields of agents and multiagent systems are a vibrant and rapidly expanding area of research. This paper has combined two separate areas of study and attempts to present a formal specification of an agent-based system through the multiagent trading system case study.

We find that using both Z and LOTOS has been an advantage to our trading system case study. We feel that the Z specification language is particulary useful in specifying agents in isolation along with their individual knowledge and goals and that LOTOS is more useful in specifying the interactions and communications between the agents. We feel that by applying Z and LOTOS to our case study it helps us to look at these two languages in a more practical rather than theoretical manner. Applying LOTOS to the trading system case study had enabled us to learn and gain an understanding for the language. We find that through our experience of using LOTOS, formal methods force the specifier to think at a low-level about trivial interactions, that in other cases may not be considered. We find that applying formal methods to our case study greatly helps to clarify the systems' requirements and behaviours.

Immediate future work will include the testing of the success of the specification. One particular aspect that is worth exploring in the future is the idea of integrating different formal methods, such as Z and LOTOS used in this case study.

REFERENCES

[1] Barrett, E., *A Formal Specification of a Multiagent Trading System*, Master's thesis, National University of Ireland, Galway, Galway, Ireland, September 2003.

[2] Bolognesi, T. and Brinksma, E., 'Introduction to the ISO Specification Language LOTOS', *Computer Networks and ISDN Systems*, **14**(1), 25–59, (1987).

[3] Davies, J. and Woodcock, J., *Using Z: Specification, Refinement and Proof*, Prentice Hall, 1996.

[4] D'Inverno, M. and Luck, M., *Understanding Agent Systems*, Springer-Verlag, Berlin, Germany, 2001.

[5] D'Inverno, M. et al., 'Formalisms for Multi-Agent Systems', *The Knowledge Engineering Review*, **12**(3), (1997).

[6] D'Inverno, M., Kinny, D., Luck, M. and Wooldridge, M., 'A Formal Specification of d'MARS', Proceedings of the Fourth International Workshop on Agent Theories, Architecture and Languages. Springer-Verlag, (1998).

[7] Finin, T., 'KQML as an Agent Communication Language', Proceedings of the 3rd International Conference on Information and Knowledge Management (CIKM'94), pp. 456–463, Gaithersburg, MD, USA, (1994). ACM Press.

[8] Finin, T. and Labrou, Y., *Agent Communication Languages*, World Wide Web, http://www.cs.umbc.edu/ finin/papers/asama99tutorial.pdf, 1999.

[9] Hall, J.A, 'Seven Myths of Formal Methods', *IEEE Computer*, **7**(5), 11–19, (1990).

[10] Hermans, B., *Intelligent Software Agents on the Internet*, World Wide Web, http://www.hermans.org/agents/index.html, 1996.

[11] Huhns, M. and Stephens, L., *MultiAgent Systems. A Modern Approach to Distributed Artificial Intelligence*, chapter Multiagent Systems and Societies of Agents, The MIT Press, Cambridge, Massachusetts, 2000.

[12] Jennings, N.R., Parsons, S., Sierra, C. and Faratin, P., 'Automated Negotiation', Proc. 5th Int. Conf. on the Practical Application of Intelligent Agents and Multi- Agent Systems (PAAM-2000), pp. 23–30, Manchester, UK, (2000).

[13] Logrippo, L. and Melanchuk, T. and Du Wors, R.J., 'The Algebraic Specification Language (LOTOS): An Industrial Experience', in *Proceedings of the ACM SIGSOFT International Workshop on Formal Methods in Software Development*, (1990).

[14] Lomuscio, A., Wooldridge, M. and Jennings, N., 'A Classification Scheme for Negotiation in Electronic Commerce', *In International Journal of Group Decision and Negotiation*, **12**(1), 31–56, (Jan. 2003).

[15] Luck, M. and d'Inverno, M., 'A Conceptual Framework for Agent Definition and Development', *The Computer Journal*, **44**(1), (2001).

[16] Misra, J., 'A Discipline of Multiprogramming', *ACM Computing Surveys*, **28**(4es), (1996).

[17] Peyton Jones, S., Eber, J.M., and Seward, J., 'Composing Contracts: An Adventure in Financial Engineering', *ACM SIG-PLAN Notices*, **35**(9), 280–292, (2000).

[18] Pfleeger, S. and Hatton, L., 'Investigating the Influence of Formal Methods', *IEEE Computer*, **30**(2), 33–43, (1997).

[19] Spivey, J., *The Z notation: Reference Manual*, Prentice Hall International, 1989.

[20] Thomas, M., 'Formal Methods and their role in developing safe systems', *High Integrity Systems Journal*, **1**(5), 447–451, (1996).

[21] Turner, K., *The Formal Specification Language LOTOS: A Course For Users*, Department of Computing Science and Mathematics, University of Stirling, Scotland, 1989.

[22] Wooldridge, M., *An Introduction to MultiAgent Systems*, John Wiley and Sons Ltd, Chichester, England, 2002.

[23] Wooldridge, M. and Jennings, N., *Agent Technology, Foundations, Applications, and Markets*, Springer-Verlag, Berlin, Germany, 2002.

[24] Wooldridge, M., and Jennings, N.R., 'The Cooperative Problem-Solving Process', *Journal of Logic and Computation*, **9**(4), 563–592, (1999).

Toward Understanding Variable Ordering Heuristics for Constraint Satisfaction Problems

J. Christopher Beck[1] and **Patrick Prosser**[2] and **Richard J. Wallace**[1]

Abstract. Most previous work on understanding variable ordering heuristics for constraint satisfaction problems has focused on the ability to recover from bad decisions. It has been demonstrated however that this ability cannot be the full explanation of the quality of a variable ordering heuristic [8]. In this paper, we develop a more complete framework for analyzing heuristics based on optimal policies. We consider a second policy (in addition to the principle of quick recovery), which we call *promise*, which is simply to make choices that maximize the likelihood of successful search. We then develop a method for measuring the degree to which a heuristic conforms to the promise principle. Using this measure, we show that variable ordering heuristics vary with respect to their promise, and that for problems with many solutions the degree of promise correlates highly with efficiency of search.

1 INTRODUCTION

When presented with a number of choices, intelligence dictates that we select the best one according to some reasonable criterion. But how do we go about making a good decision? If we are on the path to reaching our goal, and we are presented with a number of options, a good decision will bring us closer to our goal. But if we have made a bad decision unknowingly and are no longer on a path to our goal, then we want to make decisions that will show us this as quickly as possible, so we can discover the bad decision and choose again. This means that in solving hard problems incrementally, we must deal with two different situations, where the assessment of decision quality may depend on different criteria.

Constraint satisfaction problems (CSPs) involve assigning values to variables in order to satisfy constraints among them. In this case, choices made during problem solving involve selecting which variable to consider next or which value to assign to this variable. Typically, some rule or *heuristic* is used to guide decision making at each step. For the most part, these rules are applied on a more-or-less ad hoc basis, and the reasons for the sometimes dramatic improvements in search efficiency are not yet clear. One criterion has been suggested for assessing heuristics, the well-known *fail first* principle. However, it has been demonstrated that adherence to this principle alone can actually impede search [8].

The main contention of this paper is that to understand heuristic performance, we must distinguish the two situations described above. We propose that each situation can be related to a basic principle or policy that defines what a good decision is, and that heuristics must be evaluated in relation to both principles. We call these principles *promise* and *fail-firstness*. *Promise* is the ability to make choices that lead to a solution when one exists, while *fail-firstness* is the ability to detect a wrong decision as soon as possible.

This paper focuses on the promise principle and on heuristics for variable selection. Although some variable ordering heuristics are known to enhance fail-firstness [6, 8], there is as yet no demonstration that they also vary with respect to promise. (On the other hand, it is expected that performance differences for value ordering heuristics will be due solely to differences in promise, although the size of these differences is unknown.) We first show how promise can be measured, which allows us to move from an abstract discussion of principles to concrete assessment of heuristics in these terms. We demonstrate that different heuristics do exhibit different degrees of promise and that the level of promise of a heuristic is correlated with its quality as measured by search cost. By taking performance principles like promise into account in evaluating heuristics, we may be able to clarify their effect on search performance. This may lead to more intelligent selection of heuristics and even to intelligent heuristic design.

1.1 Constraint satisfaction problems and search

In a constraint satisfaction problem (CSP) there is a set of variables, each with a domain of values, and a set of constraints acting between variables to restrict the set of possible value assignments. The problem is to find an assignment of values to variables that satisfies the constraints, or to prove that no such instantiation exists [9]. Many real world problems can be modeled as CSPs: scheduling problems, problems of design, routing, workforce management, etc. Typically a CSP is solved using a complete backtracking search. That is, a variable is selected for instantiation (the *current* variable) and is assigned a value. The un-assigned (*future*) variables are then filtered, removing all values from their domains that can be proved to be inconsistent with the past variable assignments. If all future variables have non-empty domains, we select from the future a new current variable. Otherwise we re-instantiate the current variable, or backtrack. The order that we select the current variable can have a profound effect on search effort [6, 7, 5]. We can use dynamic or static variable ordering heuristics. In a static variable ordering (SVO) heuristic variables are ordered before search starts, and are then always selected in that order. In contrast, dynamic variable ordering (DVO) heuristics select the current variable using information that is made available during the search process. Another primary component of search is the algorithm used to infer inconsistent values in the domains of future variables. A common consistency enforcement algorithm called forward checking [6], checks all values in the domain of the future

[1] Cork Constraint Computation Center Computing Science Department, University College Cork, Ireland {c.beck,r.wallace}@4c.ucc.ie
[2] Department of Computing Science, University of Glasgow, Scotland, pat@dcs.gla.ac.uk

variables against the assignment made to the current variable. If a value is inconsistent, it is removed from the domain.

Haralick & Elliott [6] proposed the *fail first* principle as a heuristic: *first try the places most likely to fail*. This was realized as SDF: the "smallest domain first" DVO heuristic which selects the variable with fewest values remaining in its domain. The justification for this was that it would reduce the average path length in the search tree, and this in turn would reduce search effort. Nudel [7] suggested another motivation for a heuristic, namely that it should minimize the number of nodes within the search tree. He presented an analytical model of the forward checking search process and showed that SDF will minimize the expected size of the search tree. In [8] Smith & Grant took the fail first principle to an extreme and engineered a heuristic that aggressively attempts to fail early in the search. While their heuristic did indeed reduce the average path lengths in the search tree, it did so with an increase in search cost. Clearly, the fail-first principle is not a full explanation for the quality of variable ordering heuristics.

2 RULES FOR DECISIONS: POLICIES AND HEURISTICS

When decisions are made during the course of search, it is helpful to have rules to guide the selections. Rules of this sort seem to have two basic forms, that have been called policies and heuristics, and that can be distinguished by the functions they perform. A *policy* identifies goals or end-results that are desirable. A *heuristic* identifies features of the situation that serve to distinguish among choices, such that a selection in these terms increases the likelihood of achieving the desirable goals.

It is significant that researchers in other disciplines have also seen fit to make this distinction. The following quotation is from an article in behavioral ecology (the study of behavior as adaptation) [3]:

> A policy model in our terminology is one which prescribes a general rule for maximizing payoff (the goal), without identifying any specific procedure which would enable the animal to follow the rule. (A policy for maximizing the goal of offspring production might, for example, be "Cross roads in a way that minimizes the chance of being run over", and the procedure for doing this might be "Look in both directions and cross if clear").

These authors talk about procedures or algorithms rather than heuristics, but the latter clearly form a part of any procedure in which nondeterministic choices must be made. Other workers in this field have, in fact, spoken of animals as following certain "rules-of-thumb" that allow them to approximate optimal policies.

There is an overall policy for search problems, which is to maximize the efficiency of the procedure, i.e. to minimize the effort required to find a solution, according to some relevant measure. An important measure of this sort is the number of nodes in the search tree, which is equal to the total number of assignments considered. Other measures, such as number of constraint checks and run time, attempt to better measure overall effort.

As indicated in the Introduction, two subordinate policies can be distinguished in this domain. This is because, for NP-complete and other hard combinatorial problems, search cannot be expected to make the best choice in all cases. Search may, therefore, enter a state where the partial solution cannot be extended to a complete one, i.e. the subtree below it does not contain any solutions. In this case, in order to succeed as quickly as possible, search should fail as quickly as possible so that it can get out of this subtree and return to a path that leads to a solution. This is in direct contrast to the situation where one is on such a path; in this case success should be followed by future success, to as great a degree as possible.

Promise has always been accepted implicitly as a policy in constraint solving, one that was probably too obvious to be described as such. On the other hand, the fail first principle is not so obvious, so its identification was (rightly) perceived as a discovery or a new proposal, and it has taken its place as a discernible strand in the lore of the field. Moreover, this proposal was first made in the context of the all-solutions problem, where for many algorithms the promise of different heuristics is the same. (Roughly, this is because all currently viable values in a domain must be tested in order to find all solutions.) As a result, the promise principle has tended to be disregarded. A further result is that the proper context of fail first has often been overlooked.

Another reason for the neglect of promise as a 'complementary' principle is that it is typically associated with value selection. In fact, an excellent value ordering heuristic is called the "promise" heuristic [4]. Value selection in turn is not usually expected to support fail-firstness. On this basis, one might hypothesize that while value ordering heuristics should support promise, variable ordering heuristics should support fail-firstness. In this case, one might question whether the distinction between policies and heuristics is important in practice, or even in theory. On the other hand, if variable ordering heuristics can also vary with respect to promise, then there can be no argument about the importance of this distinction, or the potential usefulness of considering heuristics in terms of how well they conform to basic policies.

3 A MEASURE OF PROMISE

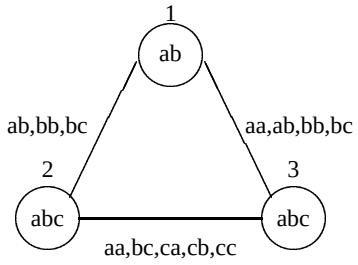
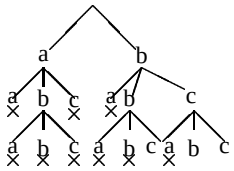


Figure 1. Example CSP with three variables, showing domains and the set of tuples in each constraint.

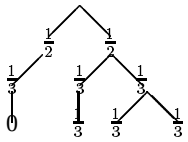
The promise policy is simply to make selections that have the greatest likelihood of succeeding. Because in practice we do not know which selection will maximize promise, we are forced to use heuristics. "Likelihood of success" can be treated probabilistically. That is, for a given decision there is some probability over all possible subsequent decisions that the choice will lead to success. We can also consider promise with respect to an entire problem, i.e. as an expected value over all possible sequences of choices. In this sense, we can speak of the promise of a problem in terms of its relation to a perfect selection. This measure of promise also has a natural minimum and maximum: 0 (for insoluble problems) and 1 (if every n-tuple is a solution). This gives us a universal measure (across all problems) that is linked directly to an optimal policy; if we can obtain a value for this measure given a problem and heuristic, we can tell how well that heuristic conforms to the ideal policy on that problem. (Whether "promise" refers to the policy or to the measure based on this policy should be clear from the context.)

These points can be illustrated with a toy problem, in which promise can be worked out exactly for different consistency algorithms as well as for the entire problem. The problem consists of three variables, each with two or three values in its domain (see Figure 1).

For this problem, the search tree for simple backtracking (no filtering), using lexical variable ordering is:



To calculate promise, we consider the probability of choosing each viable value from a domain, when any value is equally likely to be chosen:



From this, we can calculate the overall promise for backtracking on this problem, by summing the path-products. The latter are:

$$\frac{1}{2}(\frac{1}{3})(0) = 0$$

$$\frac{1}{2}(\frac{1}{3})(\frac{1}{3}) = \frac{1}{18}$$

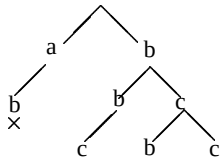
$$2\left(\frac{1}{2}(\frac{1}{3})(\frac{1}{3})\right) = \frac{2}{18}$$

And the resulting sum is:

$$0 + \frac{1}{18} + \frac{2}{18} = \frac{1}{6}$$

So the overall promise for this problem and this consistency algorithm is $\frac{1}{6}$. For backtracking, this is, in fact, the same as the solution density (cf. Section 5).

Now, we consider forward checking, using the SDF ordering. Here the search tree looks like:



Making the same calculations as before for this tree, we have the path-products,

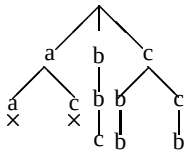
$$\frac{1}{2}(0) = 0, \quad \frac{1}{2}(\frac{1}{2})(1) = \frac{1}{4}, \quad \frac{1}{2}(\frac{1}{2})(1) = \frac{1}{4}$$

which gives the sum:

$$0 + \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$$

So the overall promise for this problem, given this variable ordering and the forward checking algorithm, is $\frac{1}{2}$.

In this case, if we use a different variable ordering, say 3-2-1 instead of 1-2-3, we, of course, get a different search tree:



Making the same calculations for this tree, we have path-products:

$$\frac{1}{3}(0) = 0, \quad \frac{1}{3}(1)(1) = \frac{1}{3}, \quad \frac{1}{3}(\frac{1}{2})(1) = \frac{1}{6}, \quad \frac{1}{3}(\frac{1}{2})(1) = \frac{1}{6}$$

and the sum:

$$0 + \frac{1}{3} + \frac{1}{6} + \frac{1}{6} = \frac{2}{3}$$

So the overall promise for this problem and this algorithm is $\frac{2}{3}$.

From this we can draw some important conclusions:

- Promise can vary depending on the variable ordering, as well as the consistency algorithm. (In the above three examples, three different values were obtained for the overall promise.)
- Promise is not in general equivalent to solution density; the equivalence holds only if there is no consistency maintenance.
- There does not appear to be a particular variable ordering that is guaranteed to give the best value for promise. (One might have expected SDF to do better than the alternative, and it may indeed on average.) Therefore, the degree of promise of different variable ordering heuristics will probably have to be decided empirically.

3.1 Probing for promise

Although the expected value of success gives an ideal measure of promise, it is not clear from the above whether such a measure can be obtained in general. Fortunately, it is possible to assess the overall promise of a problem under a given variable ordering with the following procedure, which we call *probing*. The basic idea is to choose, with appropriate randomization, variables and values until a dead end is encountered. At this point search stops and re-starts from the beginning with a new random seed. This process is repeated until a complete solution is found; the number of probes required to do this gives us a measure of promise, as shown below. Roughly, the greater the promise the fewer the number of probes required, on average, to obtain a complete solution.

This procedure avoids contamination by fail-firstness because we never try to recover from a deadend: the number of probes should be a reflection of promise alone. If used with a random value ordering heuristic over many runs, this technique also avoids effects of value selection on promise. We can use any consistency enforcement algorithm as part of the probing procedure. This allows us to assess this aspect of search for its effect on promise.

3.2 The probe procedure estimates promise

We can prove that the number of probes is a direct estimate of the promise for a problem, under a particular variable ordering. This is because the expected number of probes can be expressed by the following formula

$$\sum_{k=1}^{\infty} kp^{k-1}q$$

where q is the probability that the probe will succeed, and is therefore equal to the overall promise, and p is the probability that this probe

will fail. Using $1-p$ instead of q, we have.

$$\sum_{k=1}^{\infty} kp^{k-1}(1-p) \;=\; \sum_{k=1}^{\infty} kp^{k-1} - kp^{k}$$

This summation can be seen to involve a "telescoped" expression, so it can be simplified as follows. First, we consider the sum in its finite form,

$$[p^0 - p] \;+\; [2p - 2p^2] \;+\; [3p^2 - 3p^3] \;+\; \ldots +$$

$$[(n-1)p^{n-2} - (n-1)p^{n-1}] \;+\; [np^{n-1} - np^n]$$

where $n \rightarrow \infty$. This can be reduced to:

$$p^0 \;+\; p^1 \;+\; p^2 \;+\; \ldots +\; p^{n-1} \;-\; np^n$$

$$=\; (\sum_{k=0}^{n} p^k) \;-\; np^n$$

By a well-known equivalence, the summation within the parentheses is equal to

$$\frac{1-p^n}{1-p}$$

As $n \rightarrow \infty$, the simplified summation therefore goes to $\frac{1}{1-p}$, while the second expression (after the minus sign) goes to zero. The first statement is obvious. For the second, the following proof will suffice (and a variant can be used for the first statement as well).

Proof. The ratio of successive terms as n is incremented is:

$$\frac{(n+1)p^{n+1}}{np^n} \;=\; (\frac{n+1}{n})p$$

Since this ratio goes to p as $n \rightarrow \infty$, there must be some point at which it becomes less than 1, say, where $n = N$. Therefore, if we consider successive ratios from this point (all of which are now less than 1),

$$\frac{(N+1)p^{(N+1)}}{Np^N},\; \frac{(N+2)p^{(N+2)}}{(N+1)p^{(N+1)}},\; \ldots$$

the ratio of successive numerators in this sequence to the first denominator becomes progressively closer to 0. But then the original expression must $\rightarrow 0$. □

What this argument shows is that *the expected number of probes is equal to the reciprocal of the overall promise*. We can, therefore, use the probe technique to obtain a direct estimate of promise for any heuristic on any problem, as well as estimating the average promise of a heuristic for a class of problems.

For example, for forward checking using SDF on our toy problem, the summation is

$$\sum_{k=1}^{\infty} k(\frac{1}{2})^{k-1}\frac{1}{2} \;=\; 2$$

while for the reverse ordering, it is

$$\sum_{k=1}^{\infty} k(\frac{1}{3})^{k-1}\frac{2}{3} \;=\; 1.5$$

These are, of course, the reciprocals of the values of the overall promise calculated by hand above.

4 EMPIRICAL INVESTIGATIONS

In the last section, we have shown that on a toy problem different variable orderings result in different measures of promise and that *probing* allows us to estimate promise. In this section, we will use the probing procedure to investigate two hypotheses:

1. Different variable ordering heuristics from the literature exhibit different levels of promise.
2. Promise is inversely correlated with search effort. We do not expect promise to be perfectly correlated with search effort since fail-firstness surely also has an impact on search effort. However, we expect that for easier problems, promise will be strongly correlated with search cost. As problems become more difficult, we expect the effect of promise to decrease: fail-firstness should be more important to search effort.

4.1 Experimental details

We investigate the following set of heuristics and anti-heuristics:

- SDF: "smallest domain first". The variable chosen is the one with the smallest domain.
- max-static-degree: The variable chosen is the one with the maximum degree in the original constraint graph. This is a static heuristic since degree does not change during search.
- max-forward-degree: The variable chosen is the one with the maximum degree to non-assigned variables.
- brelaz [2]: The variable chosen is the one with the smallest domain. Ties are broken by choosing the variable with smallest domain and maximum forward degree.
- domdeg [1]: The variable chosen is the one that minimizes the ratio of domain size to forward degree (i.e. the number of adjacent uninstantiated variables).
- random: A random (unassigned) variable is chosen.
- LDF: "largest domain first". The variable with the largest domain is chosen.
- min-static-degree: The variable chosen is the one with smallest degree in the original constraint graph. This is a static heuristic.
- min-forward-degree: The variable chosen is the one with smallest degree to non-assigned variables.
- anti-brelaz: The variable with largest domain is chosen. Ties are broken by choosing the variable with largest domain which has minimum forward degree.
- anti-domdeg: The variable that maximizes the the ratio of domain size to forward degree is chosen.

Forward checking (FC) [6] is used with each variable ordering heuristic.

All test problems are randomly generated with 15 variables and 10 values per variable. The density is fixed at 0.7. Tightness varies from 0.30 to 0.39 in steps of 0.01. There are 100 problems in each subset corresponding to the different tightness values. The problems are not filtered and therefore (depending on the mean solubility of the set) contain a mix of soluble and insoluble problems.

To operationalize our probing technique for our experiments, we repeat the probing procedure 100 times for each problem and variable ordering heuristic with different seeds for the random number generator. The median reciprocal of the number of probes over the 100 runs is then considered our estimate of promise for that problem and variable ordering heuristic. Recall that for problems with no solutions, we define the promise to be 0. For a set of problems

and a variable ordering heuristic, we calculate the mean estimate of promise by finding the arithmetic mean of the promise estimate over each problem in the set for that variable ordering heuristic.

To estimate the search effort for a problem and a variable ordering heuristic, we follow a similar procedure as for estimating promise. The difference is that instead of probing for solutions, we simply use chronological backtracking. Our measure of search effort is the number of consistency checks required to find a solution. Again, for a given problem and variable ordering heuristic, we run this process 100 times with differing random seeds and define the search cost to be the median number of constraint checks over the 100 runs. For a set of problems the mean search cost is the arithmetic mean of the search cost for each problem.

4.2 Results

Figure 2 presents the mean promise estimates for each variable ordering heuristic. A log-scale is used on the y-axis to make the rankings of the heuristics easier to see. Figure 3 presents the search cost for each variable ordering heuristic.

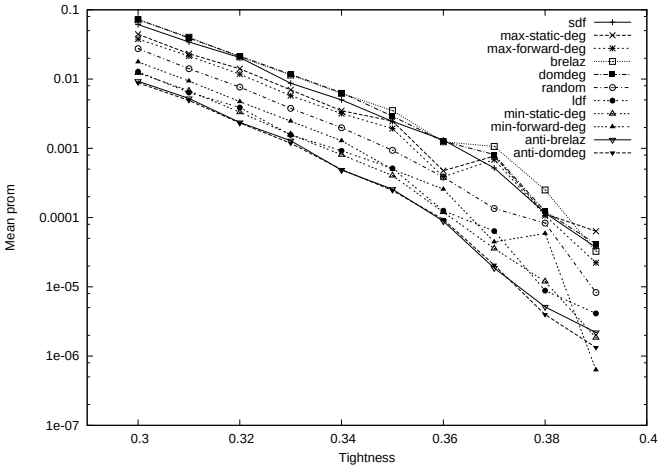


Figure 2. The mean promise estimates for each set of problems.

Figure 2 clearly shows that different variable ordering heuristics exhibit different levels of promise. As the tightness increases, there are fewer soluble problems and the level of promise correspondingly decreases. Even when we remove the insoluble problems, however, the promise decreases with increasing tightness as fewer solutions result in a lower promise for a given variable ordering heuristic.

Comparing Figures 2 and 3, we can see that the most successful heuristics (i.e., those with lowest search cost: domdeg, brelaz, and SDF) also exhibit the highest levels of promise. Furthermore, the rankings seem relatively consistent: with some exceptions, the ith best heuristic exhibits the ith highest level of promise.

Figure 4 presents a measure of the correlation between promise and search cost. The plot labeled "all" examines 1100 points for each problem set composed of the search cost and promise values for each of the 100 test problems and each of the 11 variable ordering heuristics. The plot shows that for low values of tightness, the variation in promise accounts for 70-80% of the variation in search cost. As the problems become tighter the R^2 value drops to close to 0. This is consistent with our expectations.

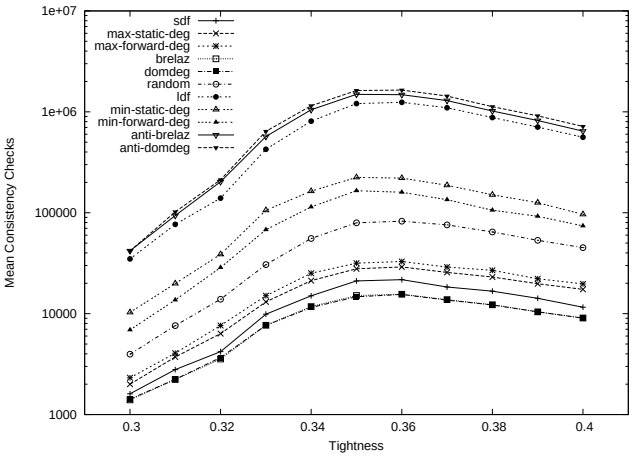


Figure 3. The mean search cost for each set of problems.

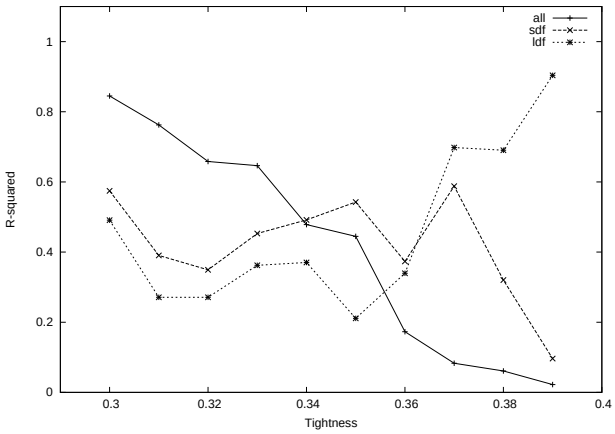


Figure 4. The R^2 values for the correlations between promise and the reciprocal of search cost for each set of problems and for selected variable ordering heuristics.

Figure 4 also presents the R^2 values for SDF and its anti-heuristic, LDF. This plot is similar to that seen with the other heuristic/anti-heuristic pairs based on domain size (i.e., domdeg and brelaz): at low values of tightness the R^2 values are somewhat noisy and around 0.4. For the tighter problem sets, the anti-heuristic, LDF, starts to have a much stronger correlation with search cost while the correlation of the heuristic, SDF, declines. This is somewhat surprising as we would expect the promise of a heuristic to be quite poorly correlated with search cost when the problems are insoluble. However, a closer look at the data shows that for problems with no solutions, the anti-heuristics have quite a small variance in search cost. The anti-heuristics do poorly, of course, but each anti-heuristic appears to do uniformly poorly across the insoluble problems at a tightness value. Given that there are very few soluble problems at high tightness values (i.e., one for set 0.39) this leads to a strong correlation between promise which is, by definition, 0, and search cost. Another surprising result is that for the problem sets with low tightness, the R^2 values are much lower than the "all" plot. In other words, the relationship between promise and search cost is weaker within any single heuristic, but over all heuristics the trend of higher promise corresponding to lower search cost for loose problems is clear.

The heuristics and anti-heuristics based only on variable degree have a similar behaviour as the domain-based heuristics for low values of tightness. As the tightness increases, however, the R^2 values of both the heuristics and anti-heuristics drop as expected. Lending support to our above explanation of the domain-based anti-heuristics, the degree-based anti-heuristics have a larger variance on insoluble problems than their domain-based counterparts.

5 DISCUSSION AND OBSERVATIONS

We have shown that promise has a high (inverse) correlation with search effort for problems with many solutions but a low correlation when the number of solutions decreases. We have also shown that the number of probes to find a solution correlates with the effort for complete search to find a solution. But why should heuristics such as SDF, domdeg, and brelaz show greater degrees of promise than other variable orderings? These heuristics give preference to variables with small domain sizes. If we assume that each value in the domain of a variable has equal probability of occurring in a solution, when domain sizes are small the probability of any value in that domain being in a solution is relatively high. That is, the values are more promising. Therefore we should expect that heuristics that prefer variables with small domains will be promising, and this is just what we have seen. This explanation may also account for the differences between heuristics and anti-heuristics that are based on degree alone, although this has not yet been tested.

These results should not come as a surprise. What we have demonstrated is that the extent to which a variable ordering affects the likelihood of finding a solution has an inverse relationship to the overall search effort. Qualitatively, this is only to be expected. But heretofore it has not been clear that differing degrees of promise actually play a (non-negligible) role in performance differences based on variable ordering. In fact, prior to this work no means of analysis has been available to address the issue. It is certainly conceivable that the ability to escape dead ends is so important that search cost depends entirely on fail-firstness and the promise of a heuristic is irrelevant or, at best, of marginal importance. Indeed, the focus on the fail-first principle in the literature and the fact that promise has, to our knowledge, never been explicitly investigated, might lead one to this expectation.

In our toy example we demonstrated that when using forward checking, different variable orderings have different levels of promise. When calculating the promise for the entire tree we are only interested in paths to solutions (all other paths are zero valued). To increase promise we want to reduce the degree of nodes in that tree, and we can do this by increasing the level of consistency during search, i.e. increasing levels of consistency will be increasingly promising. If our search process does not perform any domain filtering, but checks backwards (i.e. the algorithm is BT), then all variable orderings will have the same promise. Consider a variable X. This will correspond to an interior node in the search tree with out degree equal to the domain size of that variable. Each value will then be equally promising. In any rearrangement of the tree X will have the same out degree, and each value will have the same promise. Therefore, all heuristics will be equally promising. This reasoning has been confirmed empirically, with the problems used in the experiments above.

The likelihood that a value in the domain of a variable is in a solution will increase as we increase the level of consistency. At the extreme, with n-consistency, all domain values are in solutions, and again all heuristics will be equally promising. However, between the extremes of BT and n-consistency we should expect that as consistency levels increase the difference in promise between heuristics will decrease. Therefore, promise should actually be defined on the triple $\langle problem, heuristic, consistency\ level \rangle$.

6 FUTURE WORK AND CONCLUSION

Symmetrical to promise will be fail-firstness, i.e. the ability of a heuristic to detect a bad decision. Since heuristics are fallible, when problems are soluble both promise and fail-firstness will come into play, and when problems are insoluble promise will be irrelevant. We conjecture that variable ordering heuristics will exhibit differing levels of fail-firstness as well as promise. While formalizations of the fail-first principle have been made [6, 7], no one, to our knowledge, has developed a method for measuring fail-firstness that is independent of promise. We plan to apply the same methodology used with promise to fail-firstness and then to evaluate the extent to which search cost is correlated with the combination of promise and fail-firstness.

Why did Smith and Grant's aggressive attempts to fail earlier [8] result in poor search performance? It could be that as they increased fail-firstness, they reduced the heuristic's promise. We plan to empirically test this hypothesis. Furthermore, it would be interesting to see if we could design stronger heuristics by investigating the trade-off between promise and fail-firstness and, ideally, developing a heuristic with higher promise and higher fail-firstness.

We have presented what we believe to be an explanation of what makes DVO heuristics perform well, i.e. promise and fail-firstness. Promise guides search to a solution when one exists and fail-firstness delivers short proofs of insolubility. Variable ordering heuristics, such as SDF, brelaz and domdeg exhibit promise as well as fail-firstness. Why is this interesting? Up till now we have had conflicting guidance on what makes a good heuristic, making it difficult to design new heuristics. The present approach offers a way out of this impasse by developing a more thorough analysis of heuristics starting from first principles in the form of optimal policies.

ACKNOWLEDGEMENTS

This work was supported by Science Foundation Ireland under Grant 00/PI.1/C075.

REFERENCES

[1] C. Bessiere and J-C. Regin, 'MAC and combined heuristics: Two reasons to forsake FC (and CBJ?) on hard problems', *Principles and Practice of Constraint Programming-CP'96*, 61–75, (1996).
[2] D. Brélaz, 'New methods to color the vertices of a graph', *Communications of the ACM*, **22**(4), 251–256, (1979).
[3] J. Cheverton, A. Kacelnik, and J. R. Krebs, 'Optimal foraging: constraints and currencies', in *Experimental Behavioral Ecology and Sociobiology*, eds., B. Hölldobler and M. Lindauer, 109–126, Sinauer, Sunderland, MA, (1985).
[4] P. A. Geelen, 'Dual viewpoint heuristics for binary constraint satisfaction problems', in *Proc. 10th European Conf. Artif. Intell.*, pp. 31–35, (1992).
[5] I.P. Gent, E. MacIntyre, P. Prosser, B.M. Smith, and T. Walsh, 'An empirical study of dynamic variable ordering heuristics for the constraint satisfaction problem', *Principles and Practice of Constraint Programming-CP'96*, 179–193, (1996).
[6] R. M. Haralick and G. L. Elliot, 'Increasing tree search efficiency for constraint satisfaction problems', *Artif. Intell.*, **14**, 263–314, (1980).
[7] B. Nudel, 'Consistent-labeling problems and their algorithms: Expected-complexities and theory-based heuristics', *Artif. Intell.*, **21**, 263–313, (1983).
[8] B. M. Smith and S. A. Grant, 'Trying harder to fail first', in *Proc. 13th European Conf. Artif. Intell.*, pp. 249–253, (1998).
[9] E. Tsang, *Foundations of Constraint Satisfaction*, Academic Press, 1993.

Quantifying Temporal Quantifiers

Maria Buckley [1]

Abstract. A psycholinguistic analysis of the meaning of quantifying expressions is undertaken. Research which explores scalar ranking of quantificational determiners is replicated, albeit in an alternative medium. Additionally, more controls against confounding factors are implemented (e.g. cross-ranking of simultaneously presented lists with responses to randomly ordered individual presentation of determiners). Issues associated with obtaining access to participants and administering experiments via the Internet are discussed. The resulting ranking of determiners correlates highly with prior findings.

1 INTRODUCTION

Quantifiers, frequency expressions and probability expressions are an integral part of natural language. They are found in scientific textbooks, legal documents including government constitutions, ordinary conversations, etc. They are ubiquitous. However, despite their quantificational nature these expressions are somewhat vague and imprecise e.g. *quite a few, an awful lot, not quite all*. Even quantifiers with seemingly universal force are at least subject to implicit restriction:

(1)It always rains in Ireland.

The sentence in (1) may well be accepted as true by natives of Ireland, even though it clearly is not in the most literal sense of universal quantification over events or times, even when restricted to particular regions of Ireland. Psycholinguistic research in the area of these expressions has focused on a few main areas; some are highlighted below. Corresponding research on the expressions in the formal semantics literature is not discussed, as although obviously a considerable literature addresses the expressions, and partly motivates the present discussion, the primary interest here is on psycholinguistic issues.

1. **Focus effects of quantifiers:** Sanford, Moxey, and Paterson (1996) claimed that quantifiers can be distinguished by properties of focus. Differences in focus are "differences in the availability of the various subsets that constitute the logical representation of quantified statements". Experiments were carried out to test this claim and the main result was that positive quantifiers will lead to focus on reference sets whereas negative quantifiers will lead to focus on complement sets. An example is might be: *Some of the football fans went to the game*. The reference set will be the nonempty set of those fans who did go to the game, the complement set is the fans who did not go to the game. Note that conversational convention rather than semantic content is all that suggests the nonemptiness of the complement set, through implicature. In later work by Moxey, Sanford, and Dawydiak (2001), they explored what it is that "drives these focus effects and corresponding reference patterns." Experiments were again carried out to investigate this and it was concluded that denials produced more complement set reference responses than affirmations.
2. **Generalizations:** Another area of research has been the theory of *implicit quantification*. Newstead (1994) raises the possibility that verbs might contain information about the quantity of the subject or object. An example sentence is given *Nurses like accountants*. Implicit quantification could lead this sentence to mean that a lot of nurses like accountants. A study by Gilson and Abelson (1965) found that it is the verb which will influence the acceptability of the kind of generalizations made above. Episodic verbs like *produce, buy, have, steal* have a higher acceptance rate than subjective verbs like *get angry with, understand, like, avoid*. The subjects and objects of the sentences do not appear to play a part in the acceptability of the generalizations. Newstead (1994) carried out 3 experiments to investigate whether verbs are acting as implicit quantifiers. The overall conclusion was that verbs do act as implicit quantifiers.
3. **Effect that context has on the interpretation of quantifiers:** Many researchers have investigated the possibility that context influences how people make judgements on quantifiers. Pepper and Prytulak (1974) had subjects give a numerical value between 1 and 100 for five quantitative expressions in 6 contexts. The results found that the higher the base-rate, then the higher the value given to the frequency expression. An

[1] Computational Linguistics Group, Trinity College, University of Dublin, Dublin 2. In the places within the text where the first person plural pronoun is used, rather than singular, my research supervisor, Carl Vogel, is included in the intended reference.

interesting aspect of this research is that it presumes quantification to be based on a percentile scale.

4. **Scales:** Finally, one of the main areas that research in quantifiers has focused has been mapping quantifiers onto a scale. This is the focus of this paper and will be discussed in section 2.

This paper primarily involves a replication of work by Hakel (1968) on testing adult rankings of quantifying determiners on a linear scale. The second section of this paper will discuss the replicating strategy used to carry out this experiment with an alternative sampling method for inviting participation. It discusses the online experimentation system that was used. The following section outlines the methodology of the experiment itself and compares it to the original experiment. Finally, the results of the experiment are discussed and compared to the results from the original experiment.

The findings can be summarized as follows: the replication using a new methodology to additionally test whether there were effects of ordering of experimental items supported the original findings. The median rank scores assigned to quantifiers correlated with the original findings. Moreover, a separate test (not reported as conducted by Hakel (1968) in his correlations with other work, as far as I am aware) of the actual rank orderings of quantifiers was not significantly different. Thus, while replicating the experiment using a more robust sampling method (discussed further in section 3.1) the results provide further verification of the rank ordering of the temporal quantifiers experimented with. Section 5 elaborates on the ramifications of this. In particular, it is not clear that the scalar ranking of quantifier interpretations has firm semantic foundations, although the demonstration provided here of reliability in scalar ranking, both across experiments (and decades) and within experimentally controlled variations is compelling.

2 Scales

Moxey and Sanford (1993) report a study by Zimmer (1983) which found that people who are completing questionnaires would prefer to answer quantifier–related questions rather than questions with numbers. This is one of the main motivations for the vast amount of research into the area of determining a scale for quantifiers. The aim is to find a set of quantifiers that would "optimally discriminate between points on a scale". Bass and O'Connor (1974) carried out a magnitude estimation experiment where subjects gave a numerical value for *sometimes* and then they had to base the remainder of their judgements on the judgement they gave for *sometimes*. In other words, if *sometimes* led to a judgement of 40 and then *rarely* was believed to be half as often as *sometimes*, then 20 would be given for *rarely*. There were two conditions:

- **Condition 1**. expressions of frequency with important topic and expressions of amount with unimportant topic
- **Condition 2**. expressions of amount with important topic and expressions of frequency with unimportant topic

The main result of this study was that although it was possible to "fix quantitative meanings that are associated with verbal judgements of frequency and amount", there is a large amount of overlap of judgements between the points and this underlies the fact that it is quite difficult to obtain a proper scale for quantifiers.

A study by Hakel (1968) also attempted to obtain a scale for frequency expressions. 100 university subjects took part in the experiment that was carried out. The experiment was in the form of a questionnaire. Subjects were told that the aim of the experiment was to determine what the expressions meant to them. They had to give a numerical value between 1 and 100 for 20 frequency expressions. All of the quantifiers appeared on the same sheet of paper. There is no information regarding whether these quantifiers were randomized for each subject, nor alternative information which might have suggested how order effects were controlled for. The results found that *always* had the highest median value and *never* had the lowest median value. Hakel reports that variability is widespread in the results. Simpson (1944) had results which were similar with a .99 correlation between the rank orders of the medians. Thus, while the actual values supplied by individual participants varied widely, the rank orderings were quite reliable.

In this paper, I report a replication of an experiment described in further detail above. One difference between our experiment and his experiment, is that our subject sample was drawn from a much wider audience than the university students used in Hakel's experiment. It is worth replicating experiments in order to confirm results that have been previously found. It shows that the results that were reported were not due to some other factor that was not accounted for and ensures results are credible and valid. Equally experiments can be replicated in order to question obscure results. I used a slightly different methodology to the one used in the original experiment and this might influence the results.

3 REPLICATING STRATEGY

In the past, experiments, like the ones reported in this paper, have been face–to–face experiments. The experimenter administers them manually and in person. However, with the advent of the World Wide Web, there is now the facility to conduct experiments online. This is how the replication of Hakel's experiment was carried out for the research described by this paper.

Online experiments have many advantages over face–to–face experiments. They are relatively cheap to admin-

ister, there is ease of access and time flexibility for subjects, the opportunity to recruit a wide range of different subjects from all over the world, the usually error prone process of collecting and analyzing results is now virtually automatic. However, there are also many disadvantages to this Internet–experimentation. Some technical knowledge is needed by the experimenter and the subjects. Internet samples may be skewed and it is also surprisingly difficult to obtain a sample. Mann and Stewart (2000) address some of the relevant issues.

When we were carrying out this experiment we had some difficulty attracting subjects to participate in the experiment. We took the guidelines of Hewson, Yule, Laurent, and Vogel (2003) and sent messages to multiple newsgroups. Originally we targeted high–membership, high–trafficked groups. The idea behind this was that if the group had a high membership, more people would see the message and this would lead to a big sample. Also high–trafficked groups were chosen as it was thought that if many of the members were using the group actively, then this would lead to many subjects taking part in the experiment. However, this did not prove to be the case. There were many difficulties. As a result of our experience, it would appear that our strategy was wrong and we would propose actually contacting high-membership but low-trafficked groups. This would mean that our message would not get lost in a vast quantity of messages and there is perhaps more of a chance that members would read our message and so take part in the experiment. Another option is to send messages to **moderated** email discussion lists. It is important that the list is moderated so that the email is not deemed to be spam, since the subscribers will know that the message was authorized by their own list moderator.

We also include a subject line in our messages e.g. *Off-topic call for participation in an online experiment*. This is a token of respect to topic-area lists, acknowledging that it is a posting which they might wish to ignore. In the body of the message we also assure the readers that their email addresses are not collected nor passed on to third parties. Even with such assurances, many potential participants are skeptical. Appendix B provides an is an example posting in response to one of my calls for participation that was posted to the open newsgroup `misc.consumers` (reproduced in Appendix A), which is typical of one style of reaction to such calls. Fortunately, it is not the only style of response, and a useful number of people have responded constructively. However, one caveat is that using this sort of sampling technique it is virtually impossible to measure response rates; however, alternative metrics of response rates appropriate to internet based research is something that we are thinking about.

3.1 METHOD

The system that was used to carry out this experiment has been developed by the Computational Linguistics Group in Trinity College Dublin (Buckley & Vogel, 2003). It is an online experimentation tool. We created the experiment online, and solicited participation online.

The first screen that is presented to subjects had the instructions for the experiment. These were virtually the same instructions as given in the original experiment. The next screen had the questionnaire itself. It had all 20 quantifiers with a text box for the subjects judgement. Although information is not given in the Hakel paper regarding randomization, the order of the quantifiers on this screen was randomized for each subject in order to remove ordering effects that might come from ordering of items. However, we added a further dimension to this experiment. The original experiment left itself open to context effects influencing the results. We decided to obtain judgements on the same set of quantifiers for a second time but this time with only one quantifier appearing on each screen. The aim of this was to see if the judgements that people make would be very different when they are presented with just one quantifier, than when they are presented with a whole list of quantifiers on the same screen. When all of the quantifiers are on the same screen, then there is the possibility that subjects would base their judgement of the second quantifier on the value that they had given for the first quantifier. The intention of this modification to the original experimental design was to provide a within-experiment cross check on reliability of projections of the set of quantifiers onto a linear ordering.[2]

The research uses the same set of English temporal quantifying expressions employed by Hakel (1968): *always*, *frequently*, *very often*, *usually*, *generally*, *about as often as not*, *often*, *rather often*, *now and then*, *sometimes*, *usually not*, *occasionally*, *seldom*, *not often*, *once in a while*, *hardly ever*, *very seldom*, *rarely*, *almost never*.

The web based system on which the replication is constructed was designed to automate randomized presentation of multiple items on a single visual field as well as to distribute items across multiple visual fields, individually. The user could not navigate backwards across visual fields during the experiment. The order of these 20 screens were also randomized. 32 subjects took part altogether. The mean age was 41, with 8 females and 25 males.

[2] Many different changes could be made to this methodology, e.g. addditional cross reference approaches. One reviewer suggested that a corpus based study of the frequency of use of some of these expressions be carried out. Although this would indeed be interesting, this sense of frequency is distinct from the sense of frequency which is being used in this paper, i.e. the frequency each expression's interpretation arises, rather than the frequency of use of the expression. It would be interesting to determine the reliability of human judges' estimation of scales for varying contextualized instances of the quantifiers made available by corpora.

To test the results, I examined three issues. One is a similar test to that constructed by Hakel (1968), who measured the correlation of rank orderings. Two other tests were also intended. One was an alternative test of the similarity of rank orderings (both within the experiment and with the results of Hakel (1968)), using a chi-squared test. The other was a correlation, within-experiment, and with respect to the replicated study, of mean scalar scores for each of the temporal quantifiers examined.

4 RESULTS AND DISCUSSION

Results from our experiment

Frequency Expression	Mean	Median	Standard Deviation
always	99.55	95	1.30
frequently	76.73	80	9.92
very often	84.10	75	5.65
usually	79.67	65	11.70
generally	72.45	60	15.45
about as often as not	45.97	50	13.57
often	71.97	35	9.84
rather often	66.36	25	18.28
now and then	25.70	20	13.96
sometimes	34.18	15	13.17
usually not	17.09	12	15.61
occasionally	30.67	10	14.83
seldom	14.61	10	8.75
not often	23.85	10	20.03
once in a while	17.88	6	9.97
hardly ever	7.42	5	4.66
very seldom	8.66	3	5.52
rarely	7.27	2	3.84
almost	3.64	1	2.45
never	0.03	0	0.17

The table shows the mean scores, median values and standard deviations given for the quantifying expressions in the replication experiment which was carried out for this paper. The expressions are ordered according to the median scores. The average standard deviation 9.93 which is very large and shows that the values given for the expressions are very variable. *Usually, generally, about as often as not, rather often, now and then, sometimes, usually not, occasionally, not often, once in a while* all had standard deviations greater than the average standard deviation of 9.93 which shows the great variability in the results. Looking at some of the individual results, some subjects *frequently* was another subjects *very often* . In other words, the value that some subjects gave for *frequently* was the same as the value that other subjects gave for *very often*. Each of these expressions were assigned the value 75 by some subjects. *Not often* was the most variable expression with a standard deviation of 20.03. This is due to two very big outliers, as 2 subjects gave values of 90+ for it. Looking at the results obtained for *about as often as not*, it is clear that such a high standard deviation would not be expected seeing as how the mean and median values given for it are so similar. The reason for the very high standard deviation for this expression was that a few subjects gave values in the 1–5 range for it, whereas most of the other subjects judged it to be around 50. Usually it is recommended to remove outliers from statistical analysis, however, because these values were within the 1–100 range, we believe this would not be appropriate and so we have included all of the results. There is a .99 correlation between the median values of the results here and the median values found by Hakel. Thus it is possible to make the same conclusion as was made by him that subjects are “exceedingly stable about being exceedingly imprecise”.

Additionally, a correlation on the rank orderings realized in the original experiment and replication was computed — the value was 0.95 — and demonstrated a high positive correlation between the rankings in the original work, and the work here (which involved random ordering of the items, and in two different senses). Separately, a chi square test of significance was carried out to see if the rank ordering of our results was different to the rank ordering of the results obtained in Hakels experiment. A value of 1.19 was obtained with 19 degrees of freedom so it is possible to conclude that there is not a significant difference between the rank ordering of the two sets of results.

As mentioned, the subjects in this experiment were required to give a second judgement for each quantifier. This time each quantifier appeared on separate screens, allowing the possibility of double-checking individual responses to the quantifiers. When presented all together in one visual field, even if randomized in presentation for each subject, the potential for responses to items to influence each other could have an impact on the results. Because reverse navigation through the materials is not possible, both order effects and reliability checking of individual responses is also available in the modified replication reported here. There is a .99 correlation between the mean values obtained for the expressions when they were on separate screens and when they were all on the same screen. There was greatest variation for the expression *now and then*, with the mean for when it was on the same screen as the other quantifiers being 25.7, but when it appeared on its own screen it was 33.77.

As before in relating these results to those of Hakel (1968), two additional tests were carried out to examine rank ordering of temporal quantifiers. The first was a correlation computed as 0.99 between the ranks of items presented in a single visual field and those presented individually. Secondly, a chi square test was again carried out and a chi square value of 0.213 with 19 degrees of freedom was obtained which again shows that there is not a significant difference in the rank ordering of the quantifiers when they are presented on the same screen or on different screens.

Results from our experiment

Frequency Expression	Means (separate screens)	Means (same screen)
always	99.03	99.55
very often	81.85	84.1
usually	76.12	79.67
frequently	75.12	76.73
often	73.18	71.97
rather often	72.18	66.36
generally	70.91	72.45
about as often as not	45.24	45.97
sometimes	35.21	34.18
now and then	33.77	25.7
occasionally	28.52	30.67
once in a while	22.52	17.88
not often	19.42	23.85
usually not	17.15	17.09
seldom	15.39	14.61
very seldom	10.79	8.66
rarely	8.82	7.27
hardly ever	8.33	7.42
almost never	5.12	3.64
never	0.06	0.03

5 FUTURE WORK

Moxey and Sanford (1993) reviews research carried out in the area of mapping quantifiers onto a scale. Much of the research, including the experiments discussed in this paper have involved subjects making judgements about all of the quantifiers in the experiment. This leads to some within–subjects bias issues. Perhaps then the experiment described in this paper should be replicated again but this time with subjects only making one judgement for one quantifier.

Moxey and Sanford (1993) also raise the point, that because quantifiers are vague, then their scalar values could vary over time within an individual. A further experiment could be to have subjects make judgements on the same quantifiers but at a different time and then investigate how well correlated these results are. However, the experimental paradigm adopted here, attracting participants from the Internet, doesn't obviously lend itself to longitudinal studies. Certainly, this is an area to explore. Note that experiments like those of Hakel (1968) which depend on undergraduate students required to participate as a part of course work for subject solicitation are liable to their own difficulties in guaranteed longitudinal participation.

Moxey and Sanford (1993) conclude that seeing as how quantifiers are quite vague and confusing, then their "communicative impact may perhaps be found in other aspects of their meaning", which then leads them to concentrate research on the focus effects of quantifiers. In the work presented here, complement sets and their accompanying focus effects have not been explored at all. It would be interesting to examine the relationship between these two factors.

(2)Sandy often runs to campus.

(3)Leslie rarely runs to campus.

For example, in (2) referential access is highlighted for those events in which Sandy runs to campus, while in (3), there is easy access to both the events of Leslie running, and the complement set of events that involve Leslie adopting other modes of transport to arrive at campus. While the projection of temporal quantifiers onto a linear scale may seem at odds with virtually all work in model theoretic semantics, the robust reliability that has been demonstrated in making those projections has to be explained. If further work were to consider the scalar projection of complement set size estimation, then there would be both an additional reliability check on the scalar results, and perhaps a demonstration that work in formal semantics could profit from an alternative construal of temporal quantification.

Schutze (1996) argues at length about the empirical methods used in linguistic theory, focusing both on the factors that impinge on linguistic judgements and proper methods for eliciting judgements. While it isn't clear that forcing a scalar interpretation of temporal quantifiers is well founded given formal model-theoretic semantic approaches to interpreting quantifiers, this paper does demonstrate a resounding reliability in adult judgments about projections of temporal quantifiers onto a linear scale. This addresses one of the criticisms that Schutze (1996) makes about comparability of experiments that purport to depend on linguistic judgements through what might otherwise seem 'just' replicating results of prior work. The research demonstrates that even with a wider subject pool, and with controls against ordering effects, and with rigorous testing of correlations and rankings, the same rankings obtain, reliably.

This paper has presented a replication of past research into one sort of potential mental representation of temporal quantification in terms of scales. The paper verifies the earlier results with an experimental paradigm that appeals to a more diverse subject pool than the original (and potentially, though not actually, a larger subject pool size). The materials are controlled against effects of order of presentation, and with additional reliability checks involving second-presentation of items to be classified, in individual random orders. The results are compared with the earlier results in the mean scalar values assigned to each quantifier, and in the overall rank ordering. The results are also internally compared with respect to the single-visual-field presentation and the randomized individual item presentation for each subject. In the latter comparison correlations of mean scalar value per quantifier and rank orders were also analyzed. The primary finding was a verification of the earlier results, with high positive correlations and no evidence of significant differences in distributions for the rank ordering.

A secondary finding is in methodology for conducting experiments using Internet mediated research in attract-

ing subjects. It seems that high-membership yet low-volume newsgroups are best to target, and also moderated email lists, in order to avoid the perception that a call for participation in an experiment is not spam, and to maximize chances of participation.

Acknowledgements

We would like to thank the reviewers for their constructive input into the final version of this paper.

References

Bass, B. & O'Connor, E. (1974). Magnitude estimations of frequency and amount. *Journal of Applied Psychology*, *59*, 313–320.

Buckley, M. & Vogel, C. (2003). Improving internet research methods: a web laboratory. In *International Association for development of the information society*.

Gilson, C. & Abelson, R. (1965). The subjective use of inductive evidence. *Journal of Personality and Social Psychology*, *2*, 310–10.

Hakel, M. (1968). How often is often?. *American Psychologist*, *23*, 533–534.

Hewson, C., Yule, P., Laurent, D., & Vogel, C. (2003). *Internet Research Methods – a practical guide for the social and behavioural sciences*. Sage Publications.

Mann, C. & Stewart, F. (2000). *Internet Communication and Qualitative Research – A Handbook for Researching Online*. Sage Publications.

Moxey, L. & Sanford, A. (1993). *Communicating Quantities*. Lawrence Erlbaum Associates.

Moxey, L., Sanford, A., & Dawydiak, E. (2001). Denials as controllers of negative quantifier focus. *Journal of Memory and Language*, *44*, 427–442.

Newstead, S. (1994). Do verbs act as implicit quantifiers?. *Journal of Semantics*, *11*(1).

Pepper, S. & Prytulak, L. S. (1974). Sometimes frequently means seldom: context effects in the interpretation of quantitative expressions. *Journal of Research in Personality*, *8*, 95–101.

Sanford, A., Moxey, L., & Paterson, K. (1996). Attentional focusing with quantifiers in production and comprehension. *Memory and Cognition*, *24*(2), 144–155.

Schutze, C. (1996). *The Empirical Base of Linguistics: Grammaticality Judgments and Linguistic Methodology*. The University of Chicago Press.

Simpson, R. (1944). The specific meanings of certain terms indicating differring degrees of frequency. *Quarterly Journal of Speech*, *30*, 328–330.

Zimmer, A. (1983). Verbal versus numerical processing of subjective probabilities. In R.W. Scholz (Ed), Decision making under uncertainty. Amsterdam:Elsevier–North Holland.

A Sample Call for Participation

This is a representative example of a posting to a newsgroup. The postings are open to public view without restriction via normal newsreaders, or via facilities such as is provided by Google news; thus, this appendix merely archives texts essentially just as servers like Google do.

```
From: Maria Buckley (bucklem@tcd.ie)
 Subject: off-topic online experiment
        - call for participation
 Newsgroups: misc.consumers
 Date: 2003-07-30 11:43:20 PST


Greetings from Dublin, Ireland,

I am a postgraduate student and am running a
small experiment which involves assessing responses
to English words. You may have seen past
calls for participation that I have posted
for related experiments.  However this call is for a
different experiment which I am carrying
out at the moment.

I can assure you of at least two things:
1) your email address will not be recorded
2) I am aware of the complications involved
   in sampling methods

To participate you will have to navigate through
webpages with one question on each page. It takes
no longer than 10-15 minutes to complete the experiment.
The experiment does not involve consumers but
is mentioned here to secure a diverse range of participants.

If you would like to participate, please send an email to
bucklem@cs.tcd.ie before Aug 6, 2003 (preferably sooner)
and I will send in return the url and password necessary
to access the experiment.

Participation is anonymous and interested parties may
contact my department to request a final copy of the
ultimate report if they so desire.

Thanks for your time,

Maria Buckley
```

B Sample Recalcitrant Response

This is a representative response expressing skepticism about the research, posted to the same newsgroup, and still publically available via servers like Google's groups. Of course, many respondents simply just participate in the experiment, as planned.

```
From: xxx yyy (the_xxx_yyy@fastmail.fm)
 Subject: Re: off-topic online experiment - call for participation
 Newsgroups: misc.consumers
 Date: 2003-07-31 09:11:31 PST


In article <7c10afbc.0307301043.615b5@posting.google.com> in
misc.consumers, Maria Buckley <bucklem@tcd.ie> wrote:
>I can assure you of at least two things:
>1) your email address will not be recorded

And the check is in the mail.

--
xxx yyy, www Systems, ggg County, New York, USA
                                        http://zzz.com
Fortunately, I live in the United States of America, where we are
gradually coming to understand that nothing we do is ever our
fault, especially if it is really stupid.           --Dave Barry
```

A Data Mining Approach To Incident Report Retrieval

Doireann Cassidy, John Sheppard, Joe Carthy, Annne Drummond, John Dunnion [1]

Abstract. There is an empirical relationship of one accident to every three hundred incidents. The relative frequency of incidents as opposed to the relative infrequency of accidents, helps to ensure that there is a focus on safety issues. It is now standard practice in safety critical industries to record detailed reports of incidents and accidents. Many Incident Management Systems provide basic information such as statistics on the number and types of incidents. However the drawback of such systems is in correlating a new incident with all the existing incidents in the database the user must compose numerous queries in deciding what might be appropriate. Subtle or unexpected relationships often exist. This paper looks at the use of Case Based Reasoning and Information Retrieval techniques in the development of such systems.

1 INTRODUCTION

The goal of incident reporting and analysis is to prevent the recurrence of incidents and accidents. An incident may be defined as an unwanted disruption that has unfortunate and untoward consequences (Perrow, 1999). Incident reporting systems often receive little attention until after an accident has occurred. Following such an event, questions are asked such as “Has this happened before ?”. If there has been similar incidents which have brought attention to the underlying fault, then the question “why has this been allowed to happen again ?” becomes a major concern.

It can be very difficult to identify common causes among the thousands of reports that are received each year. In our research techniques were developed to detect the similarity between a new incident, and incidents that have already occurred at the time of collection of the new incident. Incident report forms were made comprised of a number of pre-specified fields or features in addition to several free text fields giving an insight into the incident, it’s cause(s), and any preventative measures taken against it.

In any organization dealing with incident reports, it is standard practice to compare all new incoming incident reports with those already stored, to check if similar incidents have previously occurred. Such a procedure allows an organization’s safety officer to establish whether a new incident is a ”once-off” or part of a pattern of incidents. A repetition of similar incidents may be due to some design or procedural problems that an organization needs to address. The subtle or unexpected relationships that may exist may not be evident to a human user (Johnson, 2000).

InRet is an Incident Management System. It can be queried just like a standard incident management system. However it also uses data mining techniques to find relationships between incidents. These techniques are based in the areas of Case-Based Reasoning (CBR)and Information Retrieval (IR).

[1] University College Dublin, Belfield, Dublin, Ireland

1.1 Introduction To Case-Based Reasoning

The investigation at hand focuses on the problem of comparing incoming incident reports with those already stored. Case-based reasoning is an Artificial Intelligence problem-solving technique that solves new problems by reusing existing problem solutions stored in the form of cases in a case-base (Leake, 1996). Because of the small corpus size used, we adopted a nearest neighbour approach:

1. Extract the features and store all incident reports in a case-base of these incidents; Take the n th incident (initially n=1) and treat it as a new incident, i.e. compare it to all of the remaining incidents;
2. List the related incidents;
3. Repeat steps 2 and 3 for all incidents in the case-base;
4. Evaluate the performance of InRet.

1.2 Introduction to Information Retrieval

The information retrieval technique used is Cosine Similarity. It is based on the vector space model. In the vector space model documents are represented as a vector of terms or keywords. The cosine similarity determines the similarity between documents by measuring the cosine of the angle between vectors of terms (Baeza-Yates & Ribeiro-Neto, 1999).

The system selects documents in response to a query, by identifying documents whose vector representations are most similar to that of the query vector. However, the keywords are stored in a textual keyword index as opposed to a case-base of features.

2 CORPUS

InRet uses a small corpus of aviation incident reports provided by the Air Accident Investigation Unit (AAIU) of Ireland, whose purpose is to investigate civil aviation accidents and incidents with the objective of preserving life and the avoidance of similar recurrences. This is a real world domain, and real data is used in the experiments.

Each incident report is comprised of a number of defined features and a free text description. The data is divided into two separate storage depositories: a case-base and a keyword index. As the AAIU reports do not follow a specific format the reports were read manually and the characteristics thought to best describe the incidents were selected for the defined features of the case base. This is illustrated in figure 1. Examples of these features include the aircraft manufacturer, registration, and when the incident occured.

The keyword index was selected manually from the textual descriptions of the reports. These terms were not weighted by term frequency. These keywords do not include the predefined features in the case base.

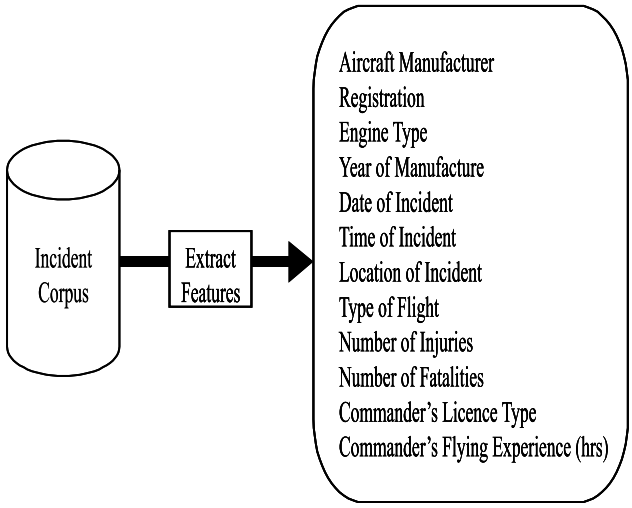


Figure 1. Feature Extraction from Aviation Corpus

3 INRET SYSTEM ARCHITECTURE

InRet comprises three layers as can be seen in figure 2:

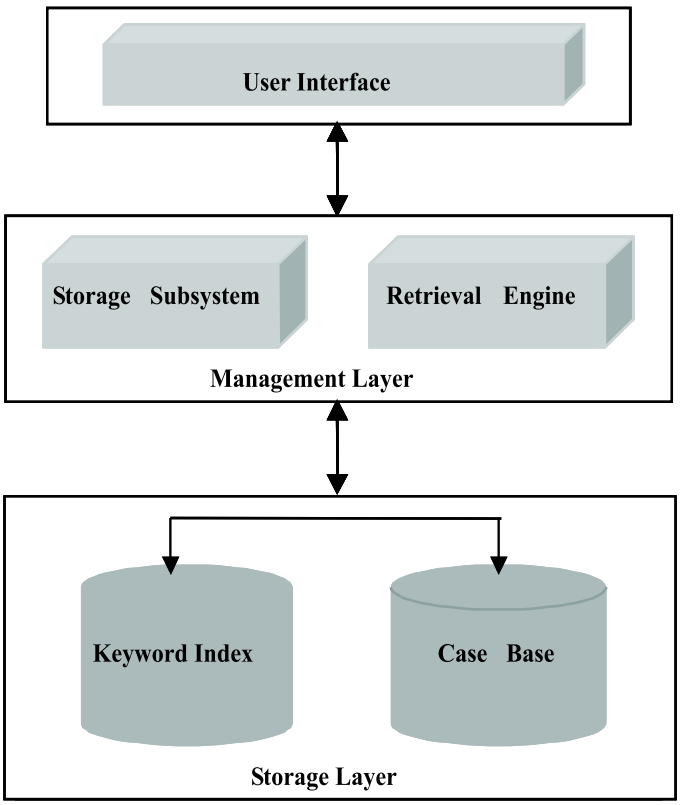


Figure 2. InRet System Architecture

1. User Interface: This is built upon the operations carried out in the management layer and provides the user with a means of interacting with the system.
2. Management Layer: This layer is responsible for the storage and retrieval operations. It acts as an intermediary between the user interface and the storage layer, using the low-level data in the storage layer to answer the user's queries. InRet uses programs written in C to carry out these operations.
3. Storage Layer: This is where the data is stored. InRet's storage layer is made up of a case-base and a textual keyword index. The case-base consists of a set of cases where each case corresponds to an incident. The keyword index contains words which are considered to be important in the description of the incident but which are not included in the defined features. These keywords are taken from the free text description of the incident.

The three layers while independent of each other, interact and communicate in order to provide a functional incident management system (Cassidy, 2002). Each layer's independence allows alterations to be made within the layers without affecting the overall functionality of the system.

4 METHODOLOGY

The system should match an incident presented by the user against each incident in the corpus, and retrieve the similar incidents. Three separate techniques are used for the retrieval of similar incidents. These are CBR, cosine similarity and a combination of these two techniques.

4.1 CBR Methodology

Two separate CBR methods are used for the retrieval of similar incident reports. These are unweighted and weighted forms of matching. According to unweighted matching, every feature is awarded an equal weight. Every feature in the incident is compared with the corresponding feature in each of the other incidents. If the features match, a score of 1 is awarded. If the features do not match, a score of 0 is awarded. A similarity score is calculated by:

1. Finding the sum of the matching features;
2. Dividing this sum by the number of features contained in the incident, as in the formula below where

$$fi \tag{1}$$

is a matching function for the

$$ith \tag{2}$$

feature of an incident:

$$Sim(T, S) = \frac{\Sigma_{i=1}^{n} fi(Ti, Si)}{n} \tag{3}$$

If the similarity score between two incidents exceeds a predetermined threshold, then they are considered to be similar. Several experiments were run in which the similarity threshold was varied in order to determine a suitable threshold value.

It may be agreed that some features are more important than others. For example, the type of aircraft may be considered to be of greater significance than the date on which the incident occurred. Subsequently, the user may decide to give more credence to the fact that the aircraft types match than to the fact that the two incidents occurred on the same day. This is provided for in the weighted matching

technique, where a user can assign a score to each feature ranking the importance of each feature intuitively.

A second weighted matching technique is also used in which sets of weights are randomly picked and assigned to the features. For illustration purposes, only three of these sets, identified as Random Weights 1, Random Weights 2 and Random Weights 3, are shown. These results are compared against the results of the intuitively weighted technique. As in the previous method, every incident is matched against each of the other incidents in the case-base. To compute the similarity between two incidents:

1. Sum the relevant scores together;
2. Divide this result by the sum of all the weights, as in the formula below:

$$Sim(T, S) = \frac{\Sigma fi(Ti, Si) * wi}{\Sigma_{i=1}^{n} wi} \quad (4)$$

4.2 Cosine Similarity

This technique, derived from vector theory, calculates the extent of the similarity between the incidents by measuring the cosine of the angle between vectors of terms, or keywords. The keyword index is processed in order to produce a file which allows the retrieval system to locate, and count, for any given incident, all keywords occurring in that incident. Every incident in the keyword index is matched against every other incident in the index, using the cosine similarity metric. As in the CBR methodology, several different similarity thresholds are used to determine the most similar incidents. The formula for calculating the cosine similarity is:

$$Cos(T, S) = \frac{\Sigma_{i=1}^{n} Ti, Si}{\sqrt{\Sigma_{i=1}^{n} Ti^2}\sqrt{\Sigma_{i=1}^{n} Si^2}} \quad (5)$$

It states that that the similarity of incident T and incident S is a ratio of the number of common features of the incidents to the square root of the product of the total number of features of incident T squared and the total number of features of incident S squared.

4.3 Combined Methodology

The final similarity metric used in the evaluation of InRet involved the amalgamation of the previous two methods. The results of the CBR methodology and those of the cosine similarity were awarded separate weights. For each incident, the similarity was computed by:

1. Getting the product of each result (CBR result and cosine similarity result) and it's weight;
2. Summing the products together and dividing this number by 2, as follows:

$$Sim(T, S) = \frac{(res\_CBR * w1) + (res\_Cos * w2)}{2} \quad (6)$$

A number of experiments were run in which the weights were varied to see if a hybrid system performed better than a system using a single methodology.

5 RESULTS/EVALUATIONS

An evaluation of the incidents was carried out manually by a single evaluator to establish a test corpus of similar incidents. This was a controlled evaluation against which we can assess the performance of the matching techniques. The s̈imilarïncidents obtained from InRet are compared against the evaluator's incidents to produce a set of results. These results are shown in Figures 3 to 6.

The metrics used were Precision and Recall.

1. *Precision* is the number of relevant incidents retrieved compared with the total number of incidents retrieved.
2. *Recall* is the number of relevant incidents retrieved compared with the total number of relevant incidents.

Both a high precision and a high recall result are desired. Generally however, as precision increases recall decreases and vice versa. This is a well known trade-off in information retrieval. Rather than focusing on each metric separately precision and recall can be combined into a single measure known as the Harmonic Mean. The harmonic mean F, of precision and recall is computed as follows:

$$F(j) = \frac{2}{\frac{1}{r(j)} + \frac{1}{p(j)}} \quad (7)$$

The function, F, assumes values in the interval [0,1], where 0 represents no similarity and 1 corresponds to exact similarity, or relevancy. Therefore, the higher the harmonic mean the more relevant a case is. A high harmonic mean is produced by a high precision and a high recall. Hence, determination of the maximum value of F can be interpreted as an attempt to find the best possible compromise between recall and precision. [Baeza-Yates & Ribeiro-Neto, 99]

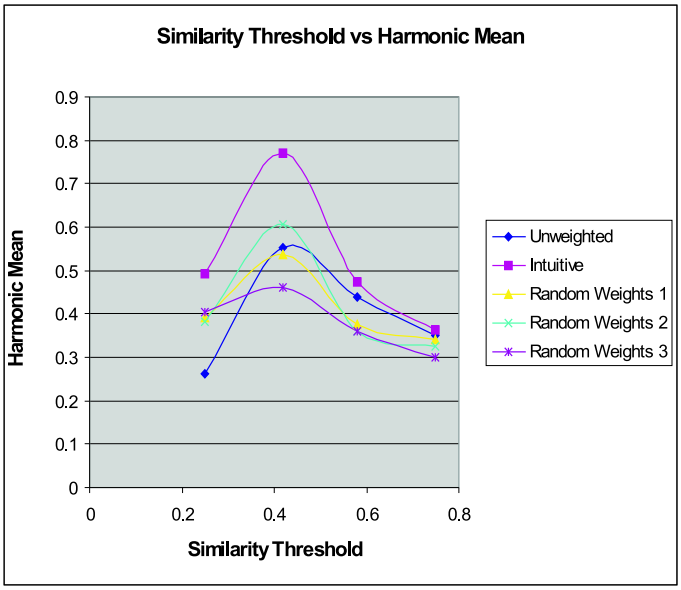


Figure 3. Similarity Threshold vs. Harmonic Mean (Case-Based Reasoning)

Figure 3 illustrates the performance of each of the CBR systems as a measure of the harmonic mean at various similarity thresholds. A system with a high harmonic mean performs better than a system with a low harmonic mean. The superiority of the intuitively weighted system is obvious. Random Weights 3 performs the worst.

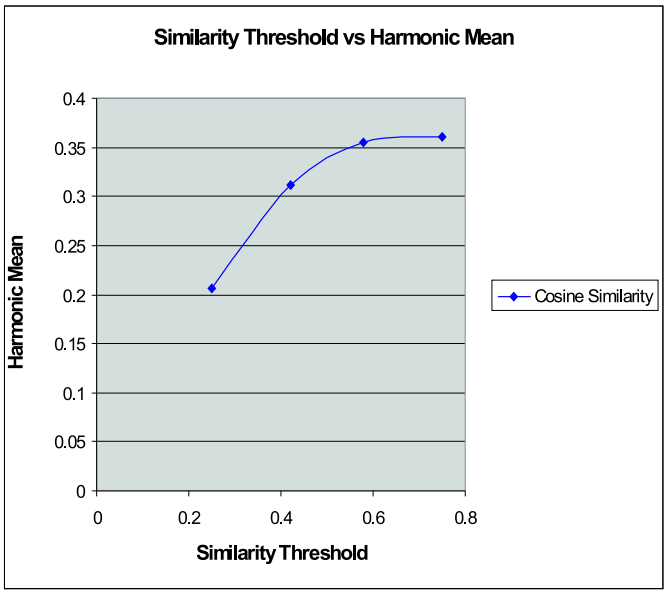


Figure 4. Similarity Threshold vs. Harmonic Mean (Cosine Similarity)

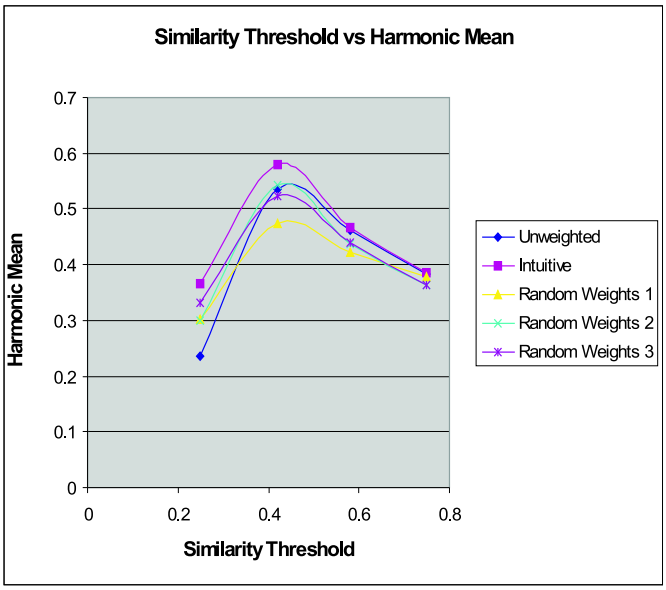


Figure 6. Similarity Threshold vs. Harmonic Mean (50% CBR - 50% Cosine Similarity)

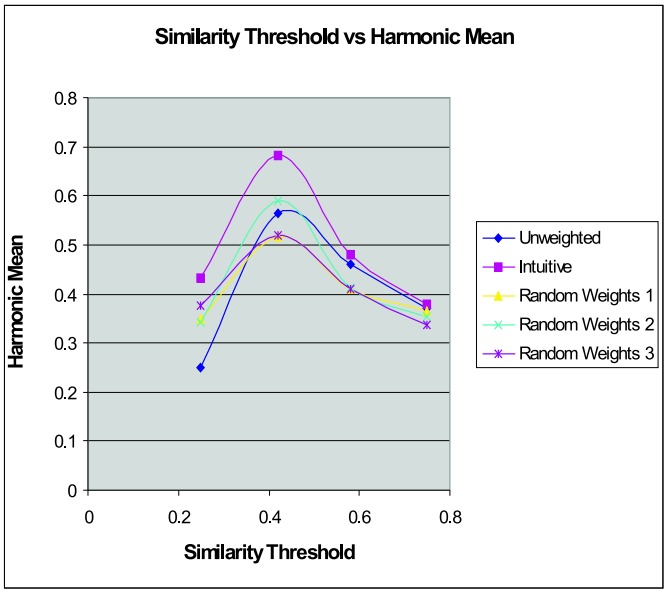


Figure 5. Similarity Threshold vs. Harmonic Mean (75% CBR - 25% Cosine Similarity)

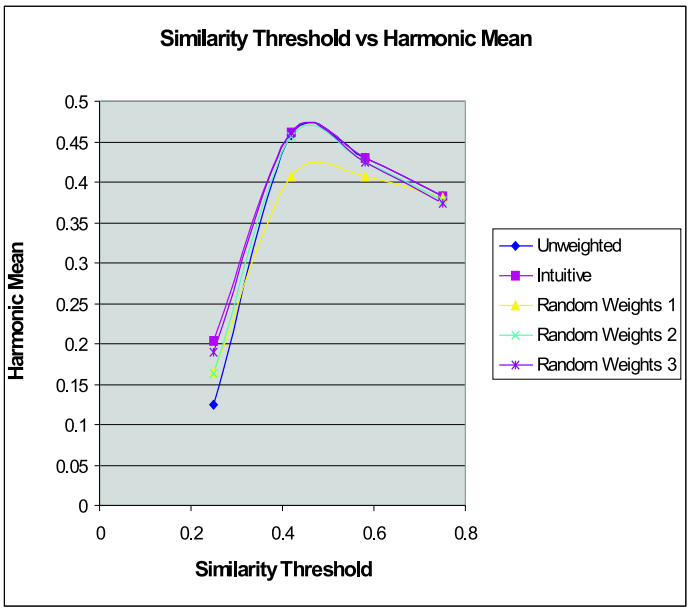


Figure 7. Similarity Threshold vs. Harmonic Mean (25% CBR - 75% Cosine Similarity)

Figure 4 shows the cosine similarity graphed as the similarity threshold against the harmonic mean. It provides the worst results by far. At a similarity threshold of 0.4 we can see the harmonic mean value is just above 0.3. Using the case-based reasoning technique and the intuitively weighted method the harmonic mean value was twice that of the cosine similarity value at almost 0.8.

Figures 5 through 7 show the results when the CBR method and cosine similarity methods are combined together. Again the intuitively weighted method outperforms the randomly weighted methods. The first combination of 75% CBR and 25% cosine can be seen in Figure 5. The results reflect a similar output with a lower harmonic mean at each similarity threshold. Figure 6 shows the two techniques evenly evenly weighted. The performance of each method decreases further with the intuitively weighted system being the one to suffer most. In Figure 7 the the combination was mostly dependant on the cosine method. Again the performance dropped. It can be seen clearly that as more weight is given to the cosine method perfromance decreases.

Comparing the CBR results with those of the combinated similarity that using CBR alone produces a better system than one using a combination of CBR and cosine. The results of the cosine have been impaired by the poor results produced by the cosine similarity.

6 CONCLUSIONS

This paper describes the design and implementation of InRet, a basic incident report retrieval system. The results of the prelimenary experiments outlined above are encouraging. The CBR results suggest a favourable set of weights will produce better results than either randomly asssigning weights or leaving the features unweighted. The evaluations also indicate a poor performance using the cosine similarity measure. However this is probably due to the total elimiination of keywords which were already defined as features. A more comprehensive set of experiments is currently underway.

7 LIMITATIONS OF CURRENT APPROACH

One obvious limitation of our research is the small corpus size and the fact that a single human evaluator carried out the evaluation. Unfortunately, no TREC-like or TDT-like corpus exists for incident retrieval research. We are currently approaching organizations such as airlines and health authorities with a view to gaining access to a larger corpus and expert evaluators. Weights can be assigned to the features either manually or automatically. Both methods have limitations. Without a specific query or user, it is not obvious as to how the weights should be delegated manually. An automatic weighting scheme would allow InRet to run through various sets of weights before deciding how it should allocate weights to achieve the best possible results. However, a user may wish to use different weights in relation to a specific incident. For ease of implementation, a limited number of defined features, and keywords, were used to describe each of the incidents. Other features have been excluded for the purpose of these preliminary experiments, but will be included in future work.

8 FUTURE WORK

The system is still undergoing development and we are investigating a number of possible enhancements such as:

1. Methods for automatically computing feature weights as well as allowing users to guide the weighting process;
2. Using all features available in the incident report;
3. Allowing the textual component to include terms already included in the defined features;
4. Utilizing lexical resources such as WordNet (Fell-baum, 1998) to handle matching synonyms, e.g. so that ”plane” matches ”airplane”.
5. Expanding the experimental corpus and improving the evaluation methodology so as to be able to generate more reliable results.

9 ACKNOWLEDGEMENTS

The support of the Basic Research Initiative of Enterprise Ireland is gratefully acknowledged.

10 REFERENCES

Baeza-Yates, R. & Ribeiro-Neto, B. (1999) Modern Infor-mation Retrieval, Addison-Wesley.

Cassidy, D., Carthy, J., Cullen, C., Hillary, P., McAdam, S., McGinty, L., Sheppard, J. (2002) The Design and Implementation of an Incident Report Retrieval Sys-tem, Ed. Vidal, T., and Liberatore, P., In Proceedings of the First STarting AI Researchers Symposium, (STAIRS-02) held in conjunction with ECAI-02, Vol.1, IOS Press, 165-167, Lyon, France.

Cunningham, P. (1998) CBR: Strengths and Weaknesses, In Proceedings of 11 th International Conference on Industrial and Engineering Applications of Artificial In-telligence and Expert Systems, Eds. A. P. del Pobil, J. Mira & M. Ali, Lecture Notes in Artificial Intelligence 1416, Vol. 2, pp517-523.

Fellbaum, C. (1998) Introduction in WordNet: An Electri-cal Lexical Database and some of its Applications, Ed. Fellbaum, C., MIT Press.

Johnson, C. (2000) Using Case-Based Reasoning to Support the Indexing and Retrieval of Incident Reports, Foresight and Precaution: Proceedings of European Safety and Reliability Conference ESRL, Netherlands, pp 1387-1394.

Leake, D.B. (1996) Case-Based Reasoning: Experiences, Lessons and future Directions, MIT Press.

Perrow, C. (1999) Normal Accidents: Living with High-Risk Technologies, Princeton University Press, New Jersey.

Reason, J.T. (1990) Human Error, Cambridge University Press.

Watson, I., Applying Case-Based Reasoning: Techniques for Enterprise Systems, Morgan Kaufmann Publishers Inc., 1997.

Implementing Faithfulness Constraints in a Finite State Model of Optimality Theory

Joan Chen-Main[1] and **Robert Frank**[1]

Abstract. Optimality Theory is a linguistic framework that characterizes surface forms in languages as the resolution of constraint conflict. Work in finite state phonology based on the rewrite rule framework that preceded Optimality Theory suggests phonological mappings fall in the class of rational relations. Previous work has shown that these two characterizations of phonological mappings are reconcilable, subject to certain restrictions. In previous models, the constraints are formalized as referring only to the output side of the mapping. This paper presents a model that optimizes over input-output pairs. This is important for modeling the family of faithfulness constraints, constraints which evaluate the relationship between input and output. The idea is based on the closure of same-length regular relations over intersection and the notion of "lenient intersection," a combination of ordinary intersection and priority union.

1 INTRODUCTION

1.1 Overview

For those who view the mind as a sort of computer, it is interesting to ask what kind of computer the mind is. To seek an answer to such an ambitious question, we might begin by tackling a smaller piece of the pie and asking the same question with respect to subcomponents of the mind. The particular subcomponent that this paper is interested in is the subcomponent that computes language. One means of investigating what kind of computer is necessary for computing language is to ask whether a correlation between natural languages and formal languages exists.

It has been suggested that a correlation indeed exists between the phonological mappings allowed in natural languages and rational relations, that is, the mappings allowed by finite state transducers. The basis of this conjecture is that the ordered sets of rewrite rules traditionally used to explain phonological mappings require only finite state power to model their effects (Johnson 1972). Many phonological rule systems were successfully modeled using finite state means and this was taken as further indication that phonological mappings most likely fall within the class of rational relations.

In response to weaknesses of the rule-based theory, attention in the phonology field shifted to constraints on forms. The adoption of a new framework, Optimality Theory (OT) (Prince and Smolensky 1993), brought about work on how the mappings allowed by OT could be modeled using finite state means. This paper considers a means for integrating faithfulness constraints, a key part of the Optimality Theory (OT) framework, into a previously proposed model of OT that did not explicitly allow for faithfulness constraints.

[1] Department of Cognitive Science, Johns Hopkins University, Baltimore, MD, USA

1.2 Optimality Theory

The central idea of OT is that surface forms in natural languages are the outcome of competing constraints. In OT, constraints have two major properties: They are ranked in importance and they are violable. The form with the violations that are least serious according to the constraint ranking is the most harmonic, i.e. optimal, form. This means that an optimal form will only violate a constraint if that violation allows the form to satisfy a higher ranked constraint.

The constraints fall into two overarching families, well-formedness constraints and faithfulness constraints. Well-formedness constraints state what a well-formed surface form should be like. These are also called markedness constraints, because they penalize marked forms, forms which are cross-linguistically avoided and used by grammars only to create contrast. For example, a constraint banning voiced obstruents (*b*, *d*, *g*) in coda position (i.e. the final position of a syllable) is a markedness constraint. Faithfulness constraints require that information in the input be realized in the surface form. An example of a typical a faithfulness constraint is the requirement that the output must preserve all segments in the input. While well-formedness constraints do compete with each other, perhaps the primary competition is characterized as well-formedness constraints standing in conflict with faithfulness constraints. Note that if an input form includes a voiced obstruent in coda position, then it is impossible to satisfy both our example faithfulness constraint and our example markedness constraint. The grammar must specify which constraint is more important to satisfy.

We can think of an OT grammar as having three components: a LEXICON, a GENERATOR (GEN), and an EVALUATOR (EVAL). The LEXICON is comprised of all underlying forms. These underlying forms are the input to the GENERATOR. GEN maps each input form to a set of candidate output forms. It is then EVAL's job to evaluate the input and candidate output pairs and select the optimal output for each input. Thus, the responsibility for accounting for regularities in surface forms rests primarily on EVAL. EVAL must include the ranking of the constraints, a means of assessing degree of violation of each input and candidate output pair with respect to each constraint, and a means of selecting the most harmonic pairs. How EVAL chooses among the possible pairs is the key device of the OT machinery and this paper focuses on using finite state operations to model EVAL's choices.

It is generally assumed that GEN generates an infinite set of candidate outputs for each input, causing some to question the psychological plausibility of OT. First, we note that an infinite set of candidate answers does not imply that a problem is logically unsolvable. Kager (1999) gives the equation $3n^2 - 3 = 45$ as an example. In this analogy, the set of candidate solutions is the set of integers, an infinite set. However, an algorithm exists that returns the unique solution. Second, some types of computation allow for

the evaluation and elimination of sets of candidates. It is not necessary to evaluate the candidates one at a time. Lastly, though certainly not proof positive of psychological plausibility, we note that the OT computation can be carried out by neural networks, a mechanism inspired by the behavior of neurons.

For a more substantial introduction, the interested reader is directed to Kager (1999) or Prince and Smolensky (1993).

1.3 Finite State Optimality Theory

Within the tradition of modeling GEN and the constraints as weighted finite-state machines (Ellison 1994, Eisner 1997 – as cited by Stabler, 1999), faithfulness constraints can be straightforwardly treated in the same manner as markedness constraints, because evaluation of a candidate against a particular constraint is coded into the system by hand. As pointed out by Eisner (2000), however, this implementation of OT can only generate the set of surface forms for a single underlying form.

In contrast, Frank and Satta (1998) and Karttunen (1998) conceive of the optimality system as a system that maps any and every input string to its optimal output form(s). Both papers demonstrate that the OT mapping may be computed with a finite state transducer as long as the system only distinguishes between a bounded number of constraint violations. Their formalizations are both done completely within a finite state calculus and both rely on composition of relations.[1] The system first maps each input to a set of outputs, creating candidate input-output pairs. The job of each constraint is to consider each set of pairs that share the same input and eliminate pairs that are not optimal. That is, among pairs with the same input, the constraint will allow only pairs that minimally violate it to pass onto the next constraint evaluation unless no pairs satisfy the constraint at hand. Thus, when no candidate pairs for a particular input satisfy the constraint at hand, all candidates with the minimal number of violations will be preserved for the next evaluation. When all the constraints have applied, only the optimal mappings should be left.

At first glace, it might seem that a variation on intersection would be the appropriate operation for this conditional elimination, but Frank and Satta and Karttunen are forced to use composition because rational relations are not closed under intersection. Because of this, their construction only allows the constraints to evaluate the output side of the mappings. Reference to only the output side is sufficient for markedness constraints, but faithfulness constraints require simultaneous reference to the underlying material, the input side of the transduction. One way to handle this would be to include all the information from the input and the output in the string that undergoes evaluation. For example, the candidate pair (*dɪgs*, *dɪgz*) could be represented as (*ddɪɪggsz*), where an underlined segment represents the underlying form. It is (*ddɪɪggsz*) that will undergo evaluation. This has been suggested explicitly in a paper by Eisner (2002), though the idea may have also appeared in Eisner's earlier work. An alternative version of this approach is to formalize constraints such that both the input and output are considered during evaluation. We may do so using a finite state calculus by requiring the OT mapping to be a same-length rational relation. Because same-length rational relations are closed under intersection, this allows us to use a variation on the intersection operation.

Phonological processes, however, include deletion and epenthesis (insertion) of segments, suggesting that phonological mappings are not same length. We will also provide a solution for this in this paper.

Section 2 both lays out the proposed model and provides some background work on which this work is based. Section 3 describes how the model works and addresses deletion and epenthesis. Section 4 contains concluding remarks.

2 A MODEL OF OT AND OPERATIONS FOR CONSTRAINT EVALUATION: OLD IDEAS AND VARIATIONS ON OLD IDEAS

This section lays out the pieces that we will use.

2.1 A model of OT – Variations on Frank and Satta (1998)

The definition below is a modification of Frank and Satta's formalization of an OT system:

Definition
An optimality system (OS) is a four-tuple $G = (\Sigma, \Pi, \text{GEN}, C)$, where
Σ is a finite alphabet,
Π is the set of pairs $(x{:}y)$ where $(x{:}y) \subseteq [\Sigma\times\Sigma]$
GEN = Π^*, and
$C = \langle c_1, \ldots c_p\rangle$, where $p \geq 1$, is an ordered sequence of total functions from Π^* to **N**.

Definition of an optimality system

As in Frank and Satta's model, we may think of members of the domain of GEN as input strings, and members of the range of GEN as candidate outputs. GEN may map each input string to multiple candidate outputs.

Unlike Frank and Satta, however, this definition treats each transduction pair of one input symbol and the output symbol it maps to as a unit. Π is the set of permitted pairs and GEN is the result of concatenating any number of permitted pairs. Thus, GEN is $[\Sigma\times\Sigma]^*$, which is a subset of $\Sigma^* \times \Sigma^*$. The strings in the former are automatically "same length" for input and output, but this is not so for the latter. This distinction is important in later discussion.

With faithfulness constraints in mind, we would like to allow constraints to refer to both the input and output strings. In the definition above, each function c_i in C represents a constraint. When c_i is applied to a member of Π^*, the function assigns the pair of strings a non-negative integer. This number is the "degree of violation," i.e the number of violations of the ith constraint that the particular string pair incurs. The co-domain of c_i is the set of possible numbers of violations c_i can assign. In typical OT research, the co-domain of c_i has an unbounded number of members. In an OS that incorporates the bounded evaluation proposal (Frank and Satta, 1998; Karttunen,1998) mentioned above, the co-domain of each constraint is finite.

This definition of constraints is slightly different from the definition used by Frank and Satta. In the definition above, constraints map string pairs to number of violations. In Frank and Satta's model, constraints map single strings to number of

[1] Perhaps it is more accurate to say that 1) Frank and Satta use a special type of relation-language intersection, right restriction, that refers only to the output side of a relation and that 2) the right restriction operation can be recast, as shown by Karttunen, as a variation on composition.

violations. In later comparison, it will be useful to be able to refer to Frank and Satta's conception of constraints, so we review it here.

$M = \langle \mathrm{m}_1, \ldots \mathrm{m}_p \rangle$, where $p \geq 1$, is an ordered sequence of total functions from Σ^* to **N**.

Frank and Satta's definition of constraints[2]

Of the candidates received for evaluation by m_i , the candidates assigned the lowest number by m_i, i.e. which minimally violate m_i, are the candidates that pass on for evaluation against the next constraint. OT^i will denote the relation that results after constraints 1 through i have been applied. In the case that every candidate in OT^{i-1} violates m_i to the same degree, then OT^i will be the same as $\mathrm{OT}^{(i-1)}$. To facilitate reference to the set that passes on, Frank and Satta use the definition of a set called $\mathrm{argmin}_\mathrm{c}\{S\}$, where S is another set and c is a function which assigns members of S a value. $\mathrm{argmin}_\mathrm{c}\{S\}$ is the subset of S that is assigned the lowest value by function c. Because the model has been modified, the set that passes on must be defined slightly differently. Below, we define $\mathrm{argmin}_\mathrm{c}\{F(x)\}$, where F is some function, $F(x)$ is a set, and c is a function which assigns values to pairs (x, y) with $y \in F(x)$.

$$\mathrm{argmin}_\mathrm{c}\{\mathrm{F}(x)\} = \{y \mid y \in \mathrm{F}(x), \quad \mathrm{c}(x, y) = \min\{\mathrm{c}(x, y') \mid y' \in \mathrm{F}(x)\}\}$$

Definition of $\mathrm{argmin}_\mathrm{c}\{\mathrm{F}(x)\}$

Frank and Satta also provide a definition for the map that an OS induces:

$$\mathrm{OT}_G^i(w) = \begin{cases} [\mathrm{GEN}](w) & \text{if } i = 0; \\ \mathrm{argmin}_{c_i}\{\mathrm{OT}_G^{i-1}(w)\} & \text{if } i \geq 1; \end{cases}$$

Definition of map induced by an optimality system

Function OT_G^p is called the optimality function associated with a particular triple G. We will follow Frank and Satta's convention of dropping the subscript when there is no ambiguity. This is the same definition we will use, but our mapping uses the modified definition of $\mathrm{argmin}_\mathrm{c}\{F(x)\}$.

2.2 Requirements for Rationality – from Frank and Satta (1998)

Frank and Satta's main result includes three requirements for a the mapping of an OS to be rational.

Let $G = (\Sigma, \mathrm{GEN}, M)$ be an OS such that GEN is a rational relation and each constraint in M is regular and has a finite co-domain. Then OT_G is a rational relation.

Frank and Satta's (1998) main result

First, GEN must be rational. Allowing GEN to be beyond the power of finite state transducers would ensure that OT^0 would not be rational. In the model presented here, we have a rational GEN by definition. Additionally, the constraints in M are assumed to be regular in that each m satisfies the following requirement: For each $\mathrm{q} \in \mathrm{N}$, the set $\{w \mid w \in \Sigma^*, \mathrm{m}_i(w) = \mathrm{q}\}$ is a regular language. For example, all strings in Σ^* that had exactly two violations of m would constitute a regular language. Let us call this the constraint language $L\mathrm{m}_{i,2}$. The i denotes the particular constraint and the 2 denotes number of violations. Frank and Satta note that nearly all constraints proposed in the OT literature are regular in this sense. Lastly, constraint evaluation must be bounded, as mentioned above. This is the condition that each constraint have a finite co-domain.

2.3 Lenient Composition – from Karttunen (1998)

Karttunen defines an operation called lenient composition and uses this to formalize OT. Against the backdrop of requirements for rationality, the idea is the same as that in Frank and Satta. Karttunen encapsulates constraint application as a finite state operation.

Lenient composition is a combination of ordinary composition and priority union, an operator introduced by Kaplan (1987). Given two relations, Q and R, the priority union of Q and R, denoted Q .P. R, is Q together with any pairs in R where the pair's input side is not an input in Q. Karttunen provides the following illustration:

$$Q = \{(a : x), (b : y)\}$$
$$R = \{(b : z), (c : w)\}$$
$$Q \,.\mathrm{P}.\, R = \{(a : x), (b : y), (c : w)\}$$

Example of priority union

Q maps a to x and b to y. R maps b to z and c to w. Q .P. R maps a to x, b to y, and c to w. Q .P. R does not map b to z, because b is an input in Q.

A definition of the priority union operator .P. is given below. $Dom(Q)$ is the language that includes all the inputs of Q and $Id(L)$ is the identity relation that carries every member of a regular language L into itself.[3]

$$Q \,.\mathrm{P}.\, R = Q \cup (Id(\overline{Dom(Q)}) \circ R)$$

Definition of priority union

When two relations R and C are leniently composed, we first compose R with C. We then take the resulting relation R∘C and priority union it with R. Lenient composition, denoted .O., is defined below.

$$R \,.\mathrm{O}.\, C = (R \circ C) \,.\mathrm{P}.\, R$$

Definition of lenient composition, compact version

$$R \,.\mathrm{O}.\, C = (R \circ C) \cup (Id(\overline{Dom(R \circ C)}) \circ R)$$

Definition of lenient composition, spelled out version

To see how lenient composition fits into Karttunen's model of OT, let us think of R as GEN and C as the identity relation on the language which satisfies the first constraint. Since GEN is rational and C is rational, the composition of GEN and C is also rational. GEN ∘ C will include all string pairs whose output satisfies constraint 1. The inputs in GEN that do not map to any output

[2] Frank and Satta called the set of constraints C and the members c_i. We call the set M, because all the members are markedness constraints. Also, we want to reserve C to refer to a set containing both faithfulness and markedness constraints.

[3] Karttunen gives his definition in the Xerox calculus. The definition here is the translation into standard notation.

which satisfies constraint 1 will not be mapped to anything by GEN ∘ *C*. These inputs, however, will be mapped to some output(s) by the relation on the right side of the union symbol. *Dom*(GEN ∘ *C*) refers to the domain of GEN ∘ *C*. We know *Dom*(GEN ∘ *C*) is a regular language because the domain of a rational relation is a regular language. We also know that $\overline{Dom(\text{GEN}\circ C)}$, i.e. the strings which do not get mapped to anything in GEN ∘ *C*, is also a regular language because regular languages are closed under complementation. Since an identity relation on a regular language yields a rational relation, $Id(\overline{Dom(\text{GEN}\circ C)})$ is rational. When we compose this relation with GEN, we will get all the string pairs in GEN whose inputs did not map to any output satisfying constraint 1. Because $Id(\overline{Dom(\text{GEN}\circ C)})$ and GEN are both rational, their composition yields a rational relation. Both relations on either side of the union symbol are rational and rational relations are closed under union. Therefore, the entire expression on the right of the equals sign is a rational relation. During the next step, we can think of *R* as the relation we get after applying constraint 1 to GEN and *C* as the identity relation on the language which satisfies the second constraint. As described here, this assumes that each constraint can be violated at most once. However, if we have bounded constraint evaluation, we can recast a constraint such as *a to be *(one a), *(two a's), *(three a's), etc.

When constraint evaluation is bounded, Frank and Satta's definition of the map an OS induces can be recast using Karttunen's lenient composition operation.

$$\text{OT}_G^i(w) = \begin{cases} [\text{GEN}](w) & \text{if } i = 0; \\ \text{OT}_G^{i-1} \,.\text{O}.\, Id(L\,\text{m}_{i,j}) & \text{if } i \geq 1; \\ \quad \text{where } 0 \leq j \leq k \end{cases}$$

Definition of map induced by an optimality system with constraint evaluation bounded at *k*

When two relations *R1* and *R2* are composed, it is the output of *R1* that becomes the input of *R2*. The input of *R1* associated with that output of *R1* does not play a role in determining whether or not the output of *R1* will map to anything in *R2*. In this sense, composition only allows reference to the output side of the transduction, which is well suited for markedness constraints.

3 OT AS A SAME LENGTH RATIONAL RELATION

3.1 What does Gen look like?

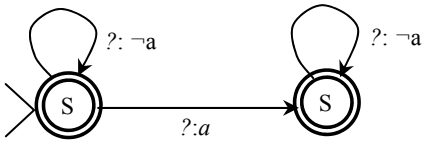
In the definition of an OS given above (section 2.1), GEN is defined as Π*. Recall that Π consists of pairs of symbols. The pairs in Π contain all the alphabet symbols, but they do not contain ε. Since pairs such as (*x*: ε) and (ε:*y*) are not members of Π, GEN is a same-length relation..[4] From the definition given for GEN, we also know GEN is a rational relation.

Same-length rational relations are the subset of the class of rational relations that consist of string pairs (*x*, *y*) such that the length of *x* is the same as the length of *y*. Kaplan and Kay (1994) discuss the properties of such relations. Because they are a subclass of rational relations, all properties of rational relations, such as closure under union, are also properties of same-length rational relations. Same-length rational relations, however, have properties that are not properties of rational relations in general. What is relevant for this paper is that while rational relations are not closed under intersection, same-length rational relations are.

We also know, however, that due to epenthesis and deletion, pairs of underlying and surface forms in phonology are not always the same length. To allow for epenthesis and deletion, we can use the symbol *0*, described by Kaplan and Kay (1994) as having been used in two-level systems which model derivational phonological rule systems. Like the empty string, ε, *0* has no pronunciation. Unlike ε, however, *0* is a bona fide alphabet symbol. Therefore, supposing that *a* is also in Σ, (*a*:*0*), (*0*:*a*), and (*0*:*0*) will all be members of Π. This means we will use a representation such as (*tɛnds*, *tɛn0z*) instead of (*tɛnds*, *tɛnz*).

3.2 What do the constraint relations look like?

In section 2.2, constraints in *M* were described as regular in that each m divided up Σ* into regular languages by number of violations. Since the constraints in *C* map string pairs to number of violations, we will have to modify our notion of regular constraints. Constraints in *C* are assumed to be regular in that each c satisfies the following requirement: For each q ∈ N, the set {(*u*,*w*) | (*u*, *w*) ∈ Π*, $c_i(u, w) = q$} is a regular relation. Just as all strings in Σ* that had exactly *j* violations of m would constitute a regular language, so all string pairs in Π* that had exactly *j* violations of c_i would constitute a regular relation. Let us call this the constraint relation $Rc_{i,j}$. The *i* denotes the particular constraint and the *j* denotes number of violations. For example, if the constraint c_1 penalizes *a*'s in the output, the finite state transducer corresponding to $Rc_{1,1}$,the relation whose output strings have at most one *a*, is drawn below. *?* means any alphabet symbol.



Finite state transducer corresponding to $Rc_{*(?:a),1}$

Because the constraint penalizes pairs, it is more accurate to think of c_1 as *(*?*, *a*) rather than **a*. Note that when c_i and m_i are markedness constraints penalizing the same forms, we can describe this transducer as accepting any string pair when the input is a member of Σ* and the output is a member of $Lm_{i,j}$. The relation it accepts is equivalent to the right restriction of Σ* x Σ* and $Lm_{i,j}$. That is,

$$Rc_{i,j} = \text{Rrst}(\Sigma^* \times \Sigma^*, Lm_{i,j}).$$

It follows that the range of the constraint relation for c_i is the same as the constraint language for the constraint m_i. That is,

$$\text{Range}(Rc_{i,j}) = Lm_{i,j}.$$

3.3 What operation do we use for constraint evaluation?

As its name suggests, the definition of lenient intersection is based on the definition of lenient composition. Lenient intersection is a combination of ordinary intersection and priority union. When two relations R and C are leniently intersected, we first intersect R with

[4] We could allow (ε:ε) to be a member of Π without changing the same-length property of GEN, though this does not have any linguistic relevance. Disallowing (*x*: ε) and (ε:*y*) as members of Π is the crucial condition.

C. We then take the resulting relation $(R \cap C)$ and priority union it with R. Lenient intersection, denoted .I., is defined below.

$$R \,.\mathrm{I}.\, C = (R \cap C) \,.\mathrm{P}.\, R$$

Definition of lenient intersection, compact version

$$R \,.\mathrm{I}.\, C = (R \cap C) \cup (Id(\overline{Dom(R \cap C)}) \circ R)$$

Definition of lenient intersection, spelled out version

Using the closure properties of same-length rational relations and regular languages, we can show that same-length rational relations are closed under lenient intersection. If *R* and *C* are same-length rational relations, then $R \cap C$ is a same-length rational relation as well. This means that $Dom(R \cap C)$ is a regular language. This in turn means that $\overline{Dom(R \cap C)}$ is a regular language. The identity relation on that language, $Id(\overline{Dom(R \cap C)})$, is a same-length rational relation. Composing two same-length rational relations, $Id(\overline{Dom(R \cap C)}) \circ R$, yields another same-length rational relation. Since both relations on either side of the union symbol are same-length rational relations and same-length rational relations are closed under union, the entire expression on the right of the equals sign is a same-length rational relation.

Lenient intersection will play a role that is very similar to the role played by lenient composition in the old system.

Having discussed the relationship between markedness constraints as previously formulated and markedness constraints as formulated here in section 3.2, let us check that the two formulations are consistent. That is, let us check that

$$\mathrm{OT} \,.\mathrm{O}.\, \mathrm{Id}(L\mathrm{m}_{i,j}) = \mathrm{OT} \,.\mathrm{I}.\, R\mathrm{c}_{i,j}.$$

Let us first describe the members of OT .O. Id(Lm$_{i,j}$).

From the definition of lenient composition, we can rewrite this relation.

$$\mathrm{OT} \,.\mathrm{O}.\, \mathrm{Id}(L\mathrm{m}_{i,j}) = [\mathrm{OT} \circ \mathrm{Id}(L\mathrm{m}_{i,j})] \,.\mathrm{P}.\, \mathrm{OT}$$

The relation on the left of the priority union operator is the composition of OT with Id(Lm$_{i,j}$). This composition will yield a subset of OT such that each member pair's output is a member of Lm$_{i,j}$. This is the same as the right restriction of OT to Lm$_{i,j}$.

$$\mathrm{OT} \,.\mathrm{O}.\, \mathrm{Id}(L\mathrm{m}_{i,j}) = [\mathrm{Rrst}(\mathrm{OT}, L\mathrm{m}_{i,j})] \,.\mathrm{P}.\, \mathrm{OT} \quad (\blacklozenge)$$

Let us now turn to the members of OT .I. $R\mathrm{c}_{i,j}$.

From the definition of lenient intersection, we can rewrite this relation.

$$\mathrm{OT} \,.\mathrm{I}.\, R\mathrm{c}_{i,j} = [\mathrm{OT} \cap R\mathrm{c}_{i,j}] \,.\mathrm{P}.\, \mathrm{OT}$$

From the discussion above, we have seen that Rc$_{i,j}$ can be written as the right restriction of Σ* x Σ* and Lm$_{i,j}$.

$$\mathrm{OT} \,.\mathrm{I}.\, R\mathrm{c}_{i,j} = [\mathrm{OT} \cap \mathrm{Rrst}(\Sigma^* \times \Sigma^*, L\mathrm{m}_{i,j})] \,.\mathrm{P}.\, \mathrm{OT}$$

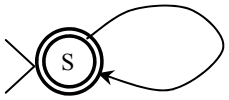
The relation on the left of the priority union operator is the intersection of OT and Rrst(Σ * x Σ*, Lm$_{i,j}$). The input side of OT, i.e. Dom(OT), is a subset of Σ* x Σ*. This means that the input side of the intersected relations will be a subset of the input side of OT. The output side of the intersected relations will be members of the output side of OT that are also members of Lm$_{i,j}$. Thus, the result of the intersection will be pairs such that the input is a member of Dom(OT) and the output is a member of Lm$_{i,j}$. This is the same as the right restriction of OT to Lm$_{i,j}$.

$$\mathrm{OT} \,.\mathrm{I}.\, R\mathrm{c}_{i,j} = [\mathrm{Rrst}(\mathrm{OT}, L\mathrm{m}_{i,j})] \,.\mathrm{P}.\, \mathrm{OT}$$

By transitivity of equality, using (♦),

$$\mathrm{OT} \,.\mathrm{O}.\, \mathrm{Id}(L\mathrm{m}_{i,j}) = \mathrm{OT} \,.\mathrm{I}.\, R\mathrm{c}_{i,j}.$$

As this paper has continually emphasized, constraints penalize pairs. This allows faithfulness constraints to be implemented in the same manner as markedness constraints. Suppose a constraint specifies that an underlying *a* must not be deleted: MAX-*a*-IO. We can think of this as *(*a*:*0*)



Finite state transducer corresponding to $R\mathrm{c}_{*(a:0),0}$

3.4 The EPEN Relation

Although we have a means, via using *0*, for allowing epenthesis and deletion and keeping the relation same-length, there is a remaining problem. A pair such as (*aab*, *aab*) must compete against pairs such as (*aa0b*, *aa0b*) or (*a00a0b*, *a00a0b*). Because *0* is treated as a real symbol, however, *aab*, *aa0b*, and *a00a0b* are all different inputs. This will disallow their outputs ought to compete against one another. We can avoid this problem by forcing all inputs to be of the same form in that there are the same number of *0*'s between each alphabet symbol corresponding to phonological content. We can think of these input *0*'s as empty input slots.

This move is justifiable by appealing to Tesar's (1995, p21) arguments that the constraints must ban cycles of epenthesis if optimality is to be well-defined. If it were the case that epenthesis eventually always increased the harmony of a form, then there would be no optimal output form, because one could always increase the harmony by epenthesizing one more segment. If epenthesis eventually had no effect on the harmony of a form, then an infinite number of optimal descriptions exist since a new optimal form could be created by taking any optimal form and adding epenthetic segments. At some point, then, epenthesis must begin to decrease the harmony of a form. I assume this is the effect of having constraints in the DEP-IO family and that an OS which includes DEP-IO constraints, there must be a bound on the number of consecutive epenthetic segments.

The number of *0*'s between the non-*0*'s, then, will be the bound on the number of consecutive epenthetic segments. Let us call the bound *q*. EPEN will pad strings made up of symbols corresponding to phonological segments (non-*0*'s) with *q* *0*'s after each non-*0*.

$$\mathrm{EPEN} = (x_1 x_2 \ldots x_n,\ 0^q x_1 0^q x_2 0^q \ldots x_n 0^q) \quad \text{where } x \neq 0 \text{ and } x \in \Sigma$$

Definition of EPEN

3.4 Putting the pieces together

The system as a whole will look like this:

$$\mathrm{EPEN} \circ ((\ldots((\ldots((\mathrm{GEN} \,.\mathrm{I}.\, R\mathrm{c}_{1,0}) \,.\mathrm{I}.\, R\mathrm{c}_{1,1}) \ldots) \,.\mathrm{I}.\, R\mathrm{c}_{1,k-1}) \ldots) \,.\mathrm{I}.\, R\mathrm{c}_{p,k-1}) \circ \mathrm{DELETE}0\text{'s}$$

where *k* is the bound on constraint evaluation and *p* is the number of constraints in *C*.

EPEN will pad every input with *0*'s to be of the form we want, $0^q(x0^q)^*$. GEN will overgenerate, but composing EPEN with GEN will leave only one input representation for each abstract underlying language form. Lastly, we can have a relation that maps each *0* to the empty string. After composing EPEN with OT^p and the resulting relation with DELETE*0*'s, the relation is no longer same-length, but it is still rational, keeping the mapping within finite state power.

That the final mapping is not same-length does not matter since we know that rational relations are closed under composition.

As an example, let us suppose the following about a toy system:

1) The bound *q* on consecutive epenthetic segments is 1. Therefore, the EPEN relation will map strings containing entirely symbols with phonological content
2) GEN only generates four same-length input-output pairs (*klæss*, *klæss*), (*klæs0s*, *klæsɪz*), (*k0l0æ0s0s0*, *k0l0æ0sɪs0*), and (*k0l0æ0s0s0*, *k0l0æ0sɪz0*)
3) c_1 bans consecutive strident consonants, a markedness constraint, and that the constraint "sees through" *0*'s – e.g. both *ss* and *s0s* are violations
   c_2 requires that every voiced segment in the output be followed by a voiced segment, also a markedness constraint
   c_3 requires output symbols to carry the same information as their corresponding input symbol, a faithfulness constraint
4) the bound *k* on the degree of violation distinguished by the system is 1 – that is, the system can only distinguish between pairs that satisfy a constraint and pairs that do not.

First, GEN is leniently intersected with $Rc_{1,0}$, all the pairs where the output does not have consecutive strident consonants. (GEN .I. $Rc_{1,0}$) yields the pairs (*klæs0s*, *klæsɪz*), (*k0l0æ0s0s0*, *k0l0æ0sɪs0*), and (*k0l0æ0s0s0*, *k0l0æ0sɪz0*). Since *k* is 1, we need not consider any other degrees of violation and move onto c_2. The relation is next leniently intersected with the pairs where every voiced segment in the output is followed by another voiced segment. ((GEN .I. $Rc_{1,0}$) .I. $Rc_{2,0}$) yields (*klæs0s*, *klæsɪz*) and (*k0l0æ0s0s0*, *k0l0æ0sɪz0*). Finally, these two pairs are leniently intersected with the relation $Rc_{3,0}$, where the input string is exactly the same as the output string. None of the string pairs in ((GEN .I. $Rc_{1,0}$) .I. $Rc_{2,0}$) are in $Rc_{3,0}$, but the priority union part of the lenient composition operator prevents (((GEN .I. $Rc_{1,0}$) .I. $Rc_{2,0}$) .I. $Rc_{3,0}$) from being the null set. Instead, the relation still includes the same two pairs, (*klæs0s*, *klæsɪz*) and (*k0l0æ0s0s0*, *k0l0æ0sɪz0*).

Having described the relation inside the parentheses, let us next consider EPEN. EPEN takes any string made up of segments carrying phonological content, such as *klæss*, and maps the string to an output made up with *q 0*'s after each input symbol. In this case, *q* = 1, so EPEN maps *klæss* to *k0l0æ0s0s0*. In our example, when we compose EPEN with (((GEN .I. $Rc_{1,0}$) .I. $Rc_{2,0}$) .I. $Rc_{3,0}$), this yields only the pair (*k0l0æ0s0s0*, *k0l0æ0sɪz0*). EPEN essentially ignores any pair in (((GEN .I. $Rc_{1,0}$) .I. $Rc_{2,0}$) .I. $Rc_{3,0}$) whose input is not of the form ?0?0?0 . . ., such as (*klæs0s*, *klæsɪz*). EPEN does not map anything to *klæs0s*. DELETE*0*'S maps *k0l0æ0sɪz0*, an input with *0*'s, to *klæsɪz*, an output without *0*'s. Thus, the end result of our toy system, EPEN ○ (((GEN .I. $Rc_{1,0}$) .I. $Rc_{2,0}$) .I. $Rc_{3,0}$) ○ DELETE*0*'S, is the pair (*klæss*, *klæsɪz*).

4 CONCLUDING REMARKS

This paper presents a model of a system that induces an Optimality Theory mapping that is a modification of the Frank and Satta (1998) model. Specifically, we have changed OT^i, the mapping after the constraints 1 through *i* have applied to GEN, defined constraints to evaluate string pairs, and added the EPEN relation to eliminate the problem created by multiple representations of the same input. When we subject the system to the same requirements for rationality put forth by Frank and Satta (1998), we can answer the question "Can we model the resolution of markedness and faithfulness constraint conflict using finite state means?" with a qualified "Yes." Because this answer is subject to certain restrictions, it cannot provide definitive evidence to bear on the larger questions posed in the introduction. The qualified "yes" does, however, provide some support for the conjecture that a correlation between natural languages and formal languages exists and the conjecture that the subcomponent of the mind that computes phonological mappings need not be more powerful than a finite state machine.

ACKNOWLEDGEMENTS

Valuable comments from Paul Smolensky in the development of this work are gratefully acknowledged. The first author is supported by a National Defense Science and Engineering Graduate Fellowship.

References

Eisner, Jason. 2000. Directional constraint evaluation in Optimality Theory. In *Proceedings of the 18th International Conference on Computational Linguistics*, 257-263.

Eisner, Jason. 2002. Comprehension and Compilation in Optimality Theory. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*.

Frank, Robert and Giorgio Satta. 1990. Optimality Theory and the generative complexity of constraint violability. *Computational Linguistics*, 24(2):307-315.

Kager, Rene. 1999. *Optimality Theory*. Cambridge University Press.

Kaplan, Ronald M. and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331-378. Written in 1980.

Karttunen, Lauri. 1998. The proper treatment of Optimality Theory in computational phonology. In *Finite-state Methods in Natural Language Processing*, 1-12, Ankara.

Prince, Alan and Paul Smolensky. 1993. Optimality Theory: Constraint interaction in generative grammar. Manuscript, Rutgers University and University of Colorado, Boulder.

Tesar, Bruce. 1995. Computational Optimality Theory. Doctoral dissertation, Department of Computer Science. University of Colorado, Boulder. Also available at the Rutgers Optimality Archive ROA-90, http://ruccs.rutgers.edu/roa.html.

As cited in Karttunen, Lauri. 1998.:
Johnson, C. Douglas. 1972. *Formal Aspects of Phonological Description*. Mouton, The Hague.

Kaplan, Ronald. 1987. Three seductions of computational psycholinguistics. In P. Whitelock, M. M. Wood, H. L. Somers, R. Johnson, and P. Bennett, editors, *Linguistic theory and Computer Applications*. Academic Press, New York. Reprinted in *Formal Issues in Lexical-Functional Grammar*. University of Chicago Press, 1996.

As cited in Stabler, E. 1999. Notes on computational phonology. Manuscript, UCLA.:
Eisner, Jason. 1997. Efficient generation in Primitive Optimality Theory. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*.

Ellison, Mark T. 1994. Phonological derivation in optimality theory. In *Proceedings of the 15th International Conference on Computational Linguistics*, 1007-1013. Also available at the Edinburgh Computational Phonology Archive.

Concept Discovery in Collaborative Recommender Systems

Patrick Clerkin[1] and **Pádraig Cunningham**[2] and **Conor Hayes**[3]

Abstract. There are two main types of recommender systems for e-commerce applications: content-based systems and automated collaborative filtering systems. We are interested in combining the best features of both approaches. In this paper, we investigate the possibility of using the k-means clustering algorithm as a basis for automatically generating content descriptions from the user transaction data that drives the collaborative filtering process. Using the the partitions of the asset space discovered by k-means, we develop a novel recommendation strategy for recommender systems. We present some encouraging results for two real world recommender systems. We conclude by outlining our approach to automatically generating descriptions of the clusters and report on an experiment designed to test concepts generated for the SmartRadio recommender system.

1 INTRODUCTION

A key role for intelligent systems in e-commerce is product recommendation [2]. Large e-commerce sites can have millions of products and customers. Since it is necessary to automatically match products to customers, recommender systems based on statistical, machine learning and knowledge discovery techniques have been developed to meet this need.

Broadly, there are two major approaches to the recommendation task, namely, content-based recommendation and automated collaborative filtering. The objective in this paper is to explore the mechanisms for taking the raw data on which collaborative recommendation is based and automatically eliciting the more semantically rich cases that can be used for content-based recommendation.

One problem with the collaborative approach is the bootstrap problem; there is no basis for making recommendations to new users who have not previously rated any assets (movies, songs, etc).

In this paper, we propose that the data that underpins the collaborative recommendation process can be mined to discover appropriate representations to underpin content-based recommendation. We show how cluster analysis can be used to generate high-level representations that can produce good quality recommendations. We also suggest that these representations are useful in overcoming the bootstrap problem.

2 RECOMMENDER SYSTEMS

As stated in the introduction, there are two approaches to recommendation on the Web. The recommendation process can be content based as represented by the upper path in Figure 1 where an appropriate representation of the assets and users requirements is determined at design time and recommendation is based on this representation. In the Case-Based Reasoning community this is referred to as case-based recommendation. The alternative lower path in the figure is automatic collaborative recommendation (ACF) which works with raw data on users ratings and behaviour and uses this data to produce recommendations. The focus of this paper is on how knowledge discovery techniques can be applied to this raw data to establish the appropriate representations for content-based recommendation. First, we will present brief descriptions of content-based and collaborative recommendation.

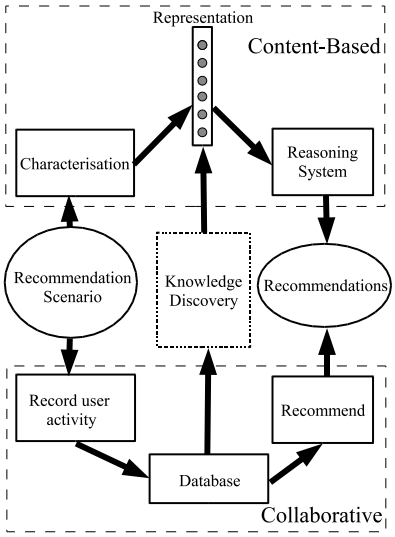


Figure 1. An overview of content-based and collaborative recommendation and the role for knowledge discovery in exploiting the benefits of both approaches

2.1 Content-based recommendation

Here we will describe a CBR-like content-based recommendation system that we can use for comparison purposes.

Table 1 shows a case-like description of a film (movie) and Table 2 shows the corresponding description of a user of the recommendation system. In this scenario recommendation is based on how well a film matches a users profile. In producing recommendations for a user, the matching score for each film in turn would be determined and the highest scoring films not already viewed would be recommended.

[1] Machine Learning Group, Department of Computer Science, University of Dublin, Trinity College, Dublin, Ireland, email: Patrick.Clerkin@cs.tcd.ie
[2] ditto, email: Padraig.Cunningham@cs.tcd.ie
[3] ditto, email: Conor.Hayes@cs.tcd.ie

This process has advantages over ACF in working well for assets of minority interest or for new assets and users.

Table 1. A case-like description of a film for content-based recommendation.

Four Weddings And A Funeral	
Title	Four Weddings and a Funeral
Year	1994
Genre	Comedy, Romance
Director	Mike Newell
Starring	Hugh Grant, Andie MacDowell
Runtime	116 mins
Country	UK
Language	English
Certification	USA:R (UK:15)

Table 2. A case-like description of user interests.

JB-7	
Name	Joe Bloggs
Preferred Era	1988
Genre	Thriller, Comedy, War, Romance
Director	S. Spielburg, F. F. Coppola.
Actors	S. Stone, S. Stallone, L. Neeson, A. MacDowell
Runtime	150 mins
Country	UK, US
Language	English
Certification	Any

2.2 Automated Collaborative Filtering

The basic idea of ACF can be shown using a simple example. If we have three users who have all shown an interest in assets as follows:

User 1: Asset 1, Asset 2, Asset 3

User 2: Asset 1, Asset 2, Asset 3, Asset 4, Asset 5, Asset 6

User 3: Asset 1, Asset 2, Asset 3, Asset 4, Asset 5

The high level of overlap indicates that these users have similar tastes. Further it seems a safe bet to recommend Asset 4 and Asset 5 to User 1 because they are endorsed by Users 2 and 3, who have similar interests to User 1.

The type of data typically encountered in ACF is illustrated by Table 3. Asset 1...5 are assets in a recommender systems, while User 1...4 are users who have rated these assets on a scale of one to five.

Table 3. Data for use in ACF where users have explicitly rated assets.

	Asset 1	Asset 2	Asset 3	Asset 4	Asset 5
User 1	3		1		5
User 2			5		
User 3	2				4
User 4		3	3		

One of the great strengths of ACF is that, if enough data is available, good quality recommendations can be produced without needing representations of the assets being recommended.

The basic structure of the recommendation process has two distinct phases. First the neighbourhood of users that will produce the recommendations must be determined. Then recommendations must be produced based on the behaviour or ratings of these users. See below (Section 4.1) for details on how this can be done.

In this paper we make reference to two recommender systems which employ ACF techniques.

MovieLens (http://www.movielens.umn.edu/) is an online film recommender system, which uses collaborative filtering to generate predictions. Users rate movies on a discrete scale and are recommended further movies on the basis of their ratings. The research team behind the system, GroupLens (http://www.cs.umn.edu/Research/GroupLens/), have made the data they have collected publicly available.

SmartRadio [4] is an experimental music recommender system deployed on the Intranet of the TCD Computer Science Department. The SmartRadio recommendation engine attempts to recommend playlists of songs to users based on their ratings of playlists to which they have listened in the past. Although the unit of recommendation is the playlist, users are asked to explicitly rate the individual tracks within the playlists on a scale of one to five.

3 CLUSTER ANALYSIS IN RECOMMENDER SYSTEMS

From the machine learning point of view, a clustering task has as its goal the unsupervised classification of a set of objects. Clustering is unsupervised in the sense that there are no *a priori* target classes used during training. In this section we outline how cluster analysis can be applied to raw user ratings data to uncover interesting patterns, the descriptions of which will constitute appropriate representations for content-based recommendation. Previously puplished work on the application of cluster analysis to recommender systems has covered the clustering of users [1, 3] and assets [7]. Other researchers have explored the benefits of clustering both users and assets simultaneously [6, 9, 10]. Our approach is to partition the assets in the database. In common with other researchers, we believe that the resulting partitioning can be used to make recommendations. However, our approach differs in the manner in which neighbourhoods of users are determined. Furthermore, we go on to use the clusters as the basis for concept formation in recommender systems.

3.1 Clustering assets

The first step is to partition the assets in the database. In the case of MovieLens the assets are the films, while in the case of SmartRadio they are the songs in the database. In both systems each asset will have been rated on a scale of one to five by a subset of the users. Thus, each asset can be represented as an object with as many attributes as there are users in the system. In recommender systems, the data sets are of very high dimension. Furthermore, the number of unknown attribute values for each asset is usually very high. This is because, in recommender systems, most users will have rated only a small fraction of the total number of assets available in the database. These problems need to be taken into account when applying clustering algorithms to recommender systems. Our implementation of the k-means clustering algorithm is discussed in Section 4.2.

3.2 Cluster-based recommendation

Once we have successfully partitioned a set of assets, the next step is to use the partitioning to form representations of users. The basic idea is to first compute for each user their *membership* of each cluster in the partitioning. The ordered series of their memberships yields their *membership vector* for the given partitioning. The membership of a user a of a cluster C is the sum of all ratings for assets in C rated by

a, divided by the sum of the ratings for all assets in the partitioning P rated by a. The membership, $M(a, C_m)$ of user a of a cluster C_m is yielded by the following formula:

$$M(a, C_m) = \frac{\sum_{i \in C_m \cap i \in A} r_{a,i}}{\sum_{j \in A} r_{a,j}} \quad (1)$$

where i ranges over all the assets in the cluster C_m, j ranges over all the assets in the partition P, and A is the set of all assets rated by the user a. To generate each term of the series to construct the membership vector, we let m range over the number of clusters in the partition. For example, consider a partitioning P consisting of five clusters and consider a user a for whom the membership of each partition has been computed. (See Figure 2).

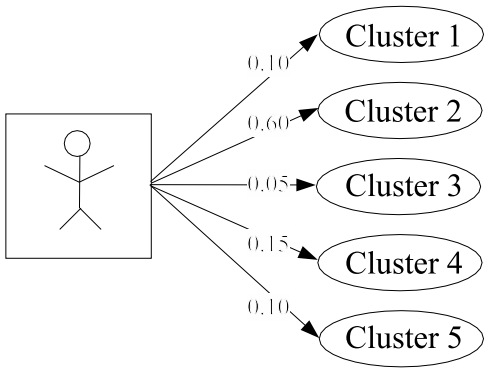


Figure 2. Each users membership of each cluster in a given partitioning is computed. The sum of all memberships for each user is 1.0

The membership vector for a might be represented as follows:

$$M(a, P) = \langle 0.1, 0.6, 0.05, 0.15, 0.1 \rangle$$

Intuitively, the membership vector for a user can be viewed as a compressed representation of that users ratings data. Thus, if we want to construct a neighbourhood of given size for a target user (the first step in the recommendation process), it is no longer necessary to compute correlations between users based on the raw ratings data; we need only compute correlations based on the (much lower-dimensional) membership vectors. Once neighbourhoods have been computed, recommendations can be made in the usual manner employed in ACF. Details of the correlation and recommendation process can be found in Section 4.1. Evaluation of the cluster-based recommendation technique is provided in the next section.

4 EVALUATION

In this section we describe our implementations of ACF and the k-means clustering algorithm. We then describe our experiments and provide results.

4.1 Our implementation of ACF

Our implementation of ACF is based on the published work of the GroupLens research group [5].

To form a neighbourhood of users for a target user the correlation between the target user and every other user in the system needs to be computed. In our ACF system, the Pearson correlation coefficient is used:

$$w_{a,u} = \frac{\sum_i (r_{a,i} - \overline{r_a})(r_{u,i} - \overline{r_u})}{\sqrt{\sum_i (r_{a,i} - \overline{r_a})^2 (r_{u,i} - \overline{r_u})^2}} \quad (2)$$

where the summations over i are over the assets which both users a and u have rated. This function yields values in the interval [-1, 1]. In forming a neighbourhood we can choose to consider only those users who are correlated above a certain threshold value. To predict a users rating for a given asset, his neighbours' ratings for that asset are aggregated, each rating being weighted according to the Pearson coefficient for that neighbour. In our implementation the neighbours ratings are also normalized using their average rating.

$$r_{a,i} = r_a + \frac{\sum_{u=1}^{k} (r_{u,i} - \overline{r_u}) w_{a,u}}{\sum_{u=1}^{k} w_{a,u}} \quad (3)$$

where the summations over u are over the k users in the neighbourhood of user a. In practice, it is also necessary to weight the Pearson correlation coefficient with a value representing the significance of the correlation. This is necessary because two users could be highly correlated on the basis of a very small number of co-rated assets. This can lead to poor predictions. We should put more confidence in less well-correlated users who have co-rated many assets. Thus, following Herlocker et al., for MovieLens, if two users have fewer than 50 assets in common, we multiply the correlation coefficient by $n/50$, where n is the number of co-rated assets. If there are 50 or more assets in common, we apply a significance weighting of 1. Let us refer to 50 as the significance cut-off for MovieLens. For SmartRadio, whose database is of lower dimensionality, we determined through trial and error that 5 serves well as a significance cut-off.

4.2 Our implementation of k-means

Our implementation of k-means is based on the descriptions of the k-means-based products on the Clustan website (http://www.clustan.com). We also programmed our version of the algorithm to accommodate unknown values in the absence of the possibility of pre-processing.

The *Euclidean Sum of Squares* (ESS) E_p for a cluster p is given by:

$$E_p = \Sigma x_{i \epsilon p} \Sigma_j (x_{ij} - \mu_{pj})^2 \quad (4)$$

where x_{ij} is the value of variable j in object i in cluster p and μ_{pj} is the mean of variable j for cluster p.

The total ESS for all clusters p is thus $E = \Sigma_p E_p$ and the increase in the Euclidean Sum of Squares $I_{p \cup q}$ at the union of two clusters p and q is:

$$I_{p \cup q} = E_{p \cup q} - E_p - E_q \quad (5)$$

While standard k-means programs relocate any object to the cluster with the nearest mean, we have implemented k-means to minimize the total Euclidean Sum of Squares E. This is preferable because while the standard approach may appear to minimize E, it does not necessarily converge quickly, or at all, because such relocations may not actually reduce E.

To minimize E we must only relocate an object i from cluster p to cluster q when $E_p + E_q > E_{p-i} + E_{q+i}$.

This is called the *exact relocation test* for minimum E. It is not the same as relocating object i to its nearest cluster mean, because any relocation from cluster p to cluster q causes consequential changes

to the means of p and q; and, in certain circumstances, these changes may actually increase E. Relocating an object i from cluster p to cluster q pulls the mean of q towards it and pushes the mean of p away from it. This can cause the distances from the mean of other cases in clusters p and q to increase, such that E is increased. With large data sets, an oscillation of boundary objects between two or more clusters can result in successive iterations. Indeed, this oscillatory behaviour was observed when the standard k-means algorithm was applied to SmartRadio.

Since E is a sum of squares, the relocation of only those objects which yield a reduction in E must result in convergence, because E cannot be indefinitely reduced. This guarantees that k-means analysis will converge if allowed enough iterations, since each iteration reduces the ESS. It also means that a relatively small number of iterations are required to reach a stable minimum ESS. This is important in the case of Smart Radio, since we need to run k-means numerous times for each value of k in order to determine the best clustering solution.

4.3 Experimental Methodology

We ran experiments on both the MovieLens and SmartRadio data sets. For every experiment we used five-fold cross-validation, using 80% of the data as a training set and reserving 20% as a test set on which predictions were made.

The first experiment was to use our implementation of ACF to generate predictions for each user-asset rating in the test sets. We calculated the absolute error of each prediction, defined as the absolute difference between the actual and predicted ratings. For each set of predictions on a test set, the *mean absolute error* (MAE) was then calculated as the sum of the absolute errors divided by the number of predictions made.

In the second experiment, we clustered each of the training sets, computed the membership vector for each user, and made predictions on the test set, as described above. We also randomly partitioned the data in each case, so as to be able to compare results generated using k-means against a random partitioning. For each fold, we generated five random partitionings and aggregated the MAE of the predictions made for that fold. In contrast, when using k-means we used a one-shot approach; that is to say, we ran k-means once for each training set for a given value of k.[4]

Note that in each experiment, we made predictions for several neighbourhood sizes. The neighbourhood sizes for SmartRadio are smaller than those for MovieLens, reflecting the relative sizes of the two data sets.

4.4 Results and conclusions

Fig. 3 is a plot of the MAE of predictions for MovieLens against neighbourhood size.

ACF performs as expected. Note, in particular, how the MAE begins relatively high, falls rapidly to a minimum, and then begins to increase again as the neighbourhood size is increased. This is characteristic beahaviour for ACF algorithms. When the neighbourhoods are too small, there is not enough information to make good predictions; when they are excessively large, there is too much irrelevant and misleading information; but when there is just the right number, the predictive capabilities of the ACF algorithm reach an optimum. The MAE for the cluster-based prediction method is not as low as for

[4] We were guided in our selection of a value for k by the silhouette technique[8]

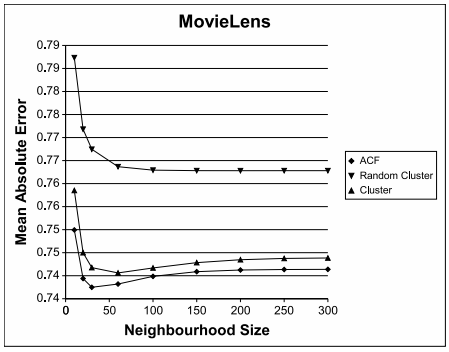


Figure 3. Plot of Mean Absolute Error against neighbourhood size for MovieLens data

'traditional' collaborative filtering methods. However, we can conclude that the k-means clustering algorithm has successfully identified interesting clusters, since a merely random partitioning yields inferior results.

Fig. 4 is a plot of the MAE of predictions for SmartRadio against neighbourhood size.

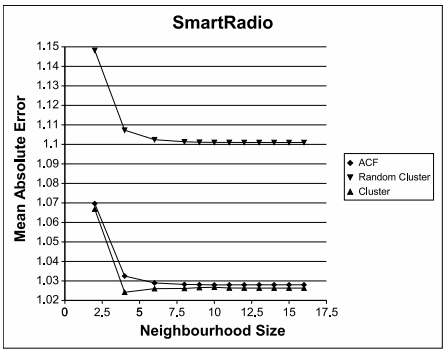


Figure 4. Plot of Mean Absolute Error against neighbourhood size for SmartRadio data

The first thing to notice is that the MAE for SmartRadio is generally higher than that for MovieLens. That given, it can be seen that the cluster-based method actually outperforms ACF for this data set, with random partitioning again being very poor. Now, SmartRadio does not exhibit the sort of behaviour discussed in relation to MovieLens; certainly, the MAE starts out high and falls to a low point with a neighbourhood size of about 4; but it does not rise again from this optimum. This is because the SmartRadio data set is not only much smaller than MovieLens but, also, much more sparse. SmartRadio is a relatively new system, deployed over a university department Intranet, while MovieLens is longer in existence and deployed over the World Wide Web. We must suppose therefore that the phenomenon we see in our experiments reflect these facts, and that SmartRadio is still in the *bootstrap* phase, where a new recommender system tries to make recommedations on the basis of the ratings of relatively few users. In this scenario, we conclude that, at least for the case of SmartRadio, our cluster-based method outperforms conventional ACF. This has a parallel in the work of Kohrs and Merialdo [6]: they discovered that a prediction method based on the simultaneous heierarchical clustering of users and assets did better than

conventional ACF in the bootstrap phase of recommender systems. At some time in the future we expect conventional ACF techniques to outperform cluster-based prediction in SmartRadio. This will be when SmartRadio acquires a critical mass of users and ratings and the MAE plots for SmartRadio start behaving similarly to those for MovieLens. Our work suggests that the cluster-based method could be used when such recommender systems are initially deployed and then, after bootstrapping is complete, conventional ACF could be introduced.

5 CONCEPT FORMATION

The experiments presented above indicate that the clusters discovered for SmartRadio are good for predictive purposes. In keeping with our objective of developing a high-level representation of raw user ratings data, the next phase is to generate descriptions of the clusters. To do this, we leverage existing knowledge.

5.1 Concept formation: describing the clusters

The main idea is to describe each cluster probabilistically according to the types of songs contained in each cluster. As a simplified example, imagine that we have a database that associates a list of descriptors with each artist in SmartRadio (Table 4).

Table 4. Artist Descriptors.

Artist	Descriptors
Bob Dylan	Rock 'n' Roll, Folk Rock, Singer/Songwriter
The Beatles	Rock 'n' Roll, Pop/Rock, Psychedelic
Bruce Springsteen	Rock 'n' Roll, Pop/Rock, Singer/Songwriter
The Prodigy	Electronica, Techno, Rave
Orbital	Electronica, Techno, Ambient Techno

Now suppose that we have just two clusters, the first with three songs, and the second with two, as in Table 5.

Table 5. Artist Descriptors.

Song	Artist	Cluster
Tangled Up In Blue	Bob Dylan	1
Yellow Submarine	The Beatles	1
Tunnel Of Love	Bruce Springsteen	1
Firestarter	The Prodigy	2
Chime	Orbital	2

In building a description of a cluster, we describe that cluster in terms of the descriptors associated with each of the songs in the cluster, maintaining a count of the number of songs in the cluster that fall under each of the decriptors. Thus, in our example, the first cluster could be described by:

Rock 'n' Roll = 3/3 (since three of the three songs in the first cluster are Rock 'n' Roll)

Folk Rock = 1/3

Singer/Songwriter = 2/3

Pop/Rock = 2/3

Psychedelic = 1/3

The description of the second cluster is as follows:

Electronica = 2/2

Techno = 2/2

Rave = 1/2

Ambient Techno = 1/2

We can interpret these descriptions probabilistically. Thus our descriptions of each cluster can be viewed as logical conjunctions of statements of the form $p = P(Dx|X \in C)$, where p is the probability that a song x is described by descriptor D, given that x is a member of cluster C.

Using the above strategy, we generated descriptions of the SmartRadio clusters. The clusters of music tracks were generated by k-means, with k = 7, on the same ratings data used in the collaborative filtering experiments. However, this time, instead of generating five sets of clusters in order to preform cross-validation, we ran the program only once on the whole dataset.

Once the clustering was performed and descriptors had been automatically compiled for each song, we calculated how many assets in each cluster fell under each descriptor. Because of the nature of these descriptors, many tracks shared descriptors, but, in some cases, only a small number of tracks fell under certain descriptors. For example, part of the output for the first cluster is as follows:

{'Adult Alternative': 12, 'Adult Alternative Pop/Rock': 30, 'Adult Contemporary': 8, 'Album Rock': 13, 'Alternative Dance': 7, 'Alternative Metal': 1, 'Alternative Pop/Rock': 31, 'Ambient': 2, 'Ambient Pop': 1, 'Ambient Techno': 3, 'American Underground': 1, ... }

You will notice that, in this example, there are indeed some descriptors which are counted as occurring only once or twice; these descriptors are surely relatively unimportant for describing the cluster, while those that have much higher counts are likely to be most important.

5.2 Experimental set-up

Since our objective was to have users of SmartRadio evaluate the concepts, it was essential to eliminate some descriptors in the final description of the clusters, as there were simply too many to expect users to examine them all. We used the following heuristic:

1. within each cluster, eliminate all descriptors which are counted less than ten times
2. across all clusters, eliminate all descriptors which occur in three or more clusters

The second step is to ensure that the descriptions of clusters do not contain descriptors which are incapable of discriminating the clusters for users. For example, an extreme case would be if a descriptor, say, *Rock* was present in the description of every cluster; in such a case, *Rock* will not help a user to differentiate between the clusters.

When this process was completed, one cluster was completely denuded of descriptors, so we did not include it in our online experiment. This left six clusters with descriptors which are recorded in Table 6.

Users were asked to examine and rate each of the six playlist descriptions in Table 8.1. They were instructed as follows: 'Imagine that you are listening to SmartRadio and are presented with playlists composed of songs falling under the genre descriptions presented here. On the basis of these descriptions, how do you think you would rate each playlist?' The experiment was conducted online and the ratings each user provided were recorded in a database for later, offline analysis.

5.3 Results and conclusions

There are sixty-two users in the SmartRadio dataset under consideration; eleven of those users participated in our experiment. For each

Table 6. Descriptions of six clusters presented to users in the SmartRadio experiment.

Cluster	Descriptors
1	World, Celtic, Adult Alternative, Ethnic Fusions, Contemporary Instrumental, Contemporary Celtic, Celtic New Age
2	House
3	Folk-Rock Britpop, Rock 'n' Roll
4	Britpop, Experimental Rock, House
5	Intelligent Dance Music, Ambient Techno, Experimental Techno, Electro-Techno, Techno, Trance, Experimental Jungle, Drill 'n' Bass, Experimental Rock, Acid Techno
6	Rock 'n Roll, Folk-Rock

user, we were able to calculate on the basis of their ratings of the six clusters in the experiment, their membership of each of those clusters. This yielded what we call a *perceived* membership vector, a term which captures the fact that, on the basis of high-level style descriptors, users perceive themselves to be aligned in a particular manner with the clusters. Now, we already had at our disposal the means to compute a user's membership of a cluster based on their ratings data. Let us call the result generated by these computations a *real* membership vector. Our question was: Do the perceived and real membership vectors match up? In other words: Are they highly correlated? If the answer to this question were affirmative, then we could conclude that the descriptions of the clusters are useful for quickly determing the preferences of users. The Pearson correlation coefficient computed between real and perceived membership vectors for the eleven users are presented in Table 8.2.

Table 7. The Pearson correlation coefficient between real and perceived membership vectors for SmartRadio users who participated in the experiment.

User	Pearson Correlation
Anon-1	0.35
Anon-2	0.59
Anon-3	0.86
Anon-4	0.76
Anon-5	0.33
Anon-6	0.04
Anon-7	0.54
Anon-8	0.53
Anon-9	0.92
Anon-10	0.89
Anon-11	0.31

The mean Pearson coefficient is calculated to be 0.56, and the standard deviation is 0.27. This indicates that the real and perceived membership vectors are highly correlated. Therefore, the descriptions used in the experiment are useful for determing the preferences of users. In particular, such cluster descriptions could be applied to the problem of making good quality recommendations to new users in the absence of any previous ratings. In SmartRadio, a new user might be asked to rate styles of music in order to determine their perceived membership of clusters, and then be recommended assets favoured by those other users whose real membership vectors closely correlate with the new user's perceived membership vector.

6 SUMMARY

We described a novel cluster-based strategy for predicting user preferences in recommender system. Empirical evaluation of our method suggested that it may work best in the context of bootstrapping a recommender system in its early stages, when there is insufficient data available for the optimal performance of conventional ACF techniques. We went on to show how a partitioning of system assets can be used as the basis for a system of concepts. Furthermore, we illustrated how the concept formation process can be automated in the case of SmartRadio by making use of pre-existing knowledge. Our experiments reveal that the resulting concepts capture users' understanding of the SmartRadio domain. This suggests that the concepts might be useful for bootstrapping new users of the SmartRadio system.

ACKNOWLEDGEMENTS

We would like to thank the GroupLens Research Group for making their MovieLens data publically available at http://www.cs.umn.edu/Research/GroupLens/data/

We would also like to thank the referees for their comments which helped improve this paper.

REFERENCES

[1] S.H.S. Chee, J. Han and K. Wang, 'RecTree: An Efficient Collaborative Filtering Method', *Lecture Notes in Computer Science*, **2114**, 141-151, (2001).

[2] P. Cunningham, R. Bergmann, S. Schmitt, R. Traphoner, S. Breen and B. Smyth, 'Websell: Intelligent Sales Assistants for the World Wide Web', *e-Business and e-Work 2001*.

[3] D. Fisher, K. Hildrum, J. Hong, M. Newman, M. Thomas, and R. Vuduc, 'Swami: A Framework for Collaborative Filtering Algorithm Development and Evaluation', *Research and Development in Information Retrieval*, (2000).

[4] C. Hayes, P. Cunningham, 'Smart Radio: Building Music Radio on the Fly', *Expert Systems*, Cambridge, UK, (2000).

[5] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl, 'An Algorithmic Framework for Performing Collaborative Filtering', *Proceedings of the 1999 Conference on Research and Development in Information Retrieval*.

[6] A. Kohrs and B. Merialdo, 'Clustering for Collaborative Filtering Applications, *Computational Intelligence for Modelling, Control and Automation (CIMCA'99)*.

[7] M. O'Connor and J. Herlocker, 'Clustering Items for Collaborative Filtering', *ACM SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation*, University of California, Berkeley, USA.

[8] P. J. Rousseeuw, 'Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis', *Journal of Computational and Applied Mathematics*, **20**, 53–56, North-Holland, (1987).

[9] L. H. Ungar and D. P. Foster, 'A Formal Statistical Approach to Collaborative Filtering', *Proc. CONALD'98*.

[10] L. H. Ungar and D. P. Foster, 'Clustering Methods for Collaborative Filtering. *Workshop on Recommendation Systems at the Fifteenth National Conference on AificiaI Intelligence*, (1998).

The Knowledge-Fitting Theory of Plausibility

Louise Connell and Mark T. Keane [1]

Abstract. Plausibility is judged on a daily basis in a wide range of cognitive phenomena, yet the study of plausibility in its own right has been long neglected in cognitive science. In this paper, we present the Knowledge-Fitting Theory of Plausibility that incorporates both concept-coherence (i.e. the conceptual structure and relatedness of a scenario) and word-coherence (i.e. the distributional properties of the individual words used to describe a scenario) in plausibility judgement. We also present the Plausibility Analysis Model (PAM), which is an implementation of this theory and the first computational mode to specifically address the issue of human plausibility judgements.

1 INTRODUCTION

Plausibility is an ineluctable phenomenon of everyday life – engaged in everything from assessing the quality of a movie plot to considering a child's excuse for a broken dish. It is perhaps this very ubiquity that has led to its neglect in cognitive science. Typically, in cognitive science literature, plausibility is merely operationalised (as ratings on a scale), rather than explained. This literature has shown the cognitive processes that utilise plausibility are many and diverse. People often use plausibility judgements in place of costly retrieval from long-term memory, especially when verbatim memory has faded [1][2][3][4]. Plausibility is also used as a kind of cognitive shortcut in reading, to speed parsing and resolve ambiguities [5][6][7]. In everyday thinking, plausible reasoning that uses prior knowledge appears to be commonplace [8], and can even aid people in making inductive inferences about familiar topics [9]. It has also been argued that plausibility plays a fundamental role in understanding novel word combinations by helping to constrain the interpretations produced [10][11]. In this way, the empirical literature leaves us with a sense that plausibility is important but without a good indication of what is actually involved. From this overview it is clear that plausibility is in need of a thorough theoretical, computational and empirical treatment.

In the rest of this paper, we outline the evidence for the effect of two factors on plausibility – concept-coherence and word-coherence. We then detail our Knowledge-Fitting Theory of Plausibility, which proposes an innovative approach to plausibility that incorporates both concept-coherence (i.e. the conceptual relatedness of a described scenario with prior knowledge) and word-coherence (i.e. the distributional information of the individual words used to describe the scenario). In addition, we describe the implementation of the theory in the form of PAM, the Plausibility Analysis Model, and discuss its performance in modelling human data [12][13].

[1] Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland. Email louise.connell@ucd.ie

2 EVIDENCE OF EFFECTS

2.1 Concept-Coherence

Although few researchers have expounded a theory of plausibility, there is a shared view that plausibility has something to do with *concept-coherence* – i.e. that "something is plausible if it is conceptually supported by prior knowledge" [8]. For example, it has been demonstrated the importance of concept-coherence to perceived plausibility by disrupting the causal sequence of events in short stories [14]. People's ability to recall these stories, and the plausibility ratings they gave, was sensitive to the degree to which the overall concept-coherence of the story had been altered. Indeed, these re-ordered stories often appeared to be just incomprehensible.

A concept-coherence view of plausibility suggests that when people make a plausibility judgement they relate a target description to their prior experience, and in some way assess whether the current scenario fits in with what they have experienced in the past. If a person read the statement "*The bottle rolled off the shelf and smashed on the floor*" they might make the bridging inference that the bottle falling *caused* it to smash on the floor. This might lead them to judge this scenario as being highly plausible because prior experience tells them that fragile things often break when they fall on hard surfaces. Put simply, the description has a certain conceptual coherence. In contrast, if the target description was "*The bottle rolled off the shelf and melted on the floor*", the person might consider it less plausible because their past experience has few examples of falling fragile objects melting on contact with floors – though a scenario could be construed where this could occur, such as if the room was made of metal and heated up like an oven). In other words, this description lacks a certain conceptual coherence.

It has been shown that manipulating the concept-coherence of a scenario (i.e. by inviting different bridging inferences) affects its perceived plausibility [12][15][16]. For example, sentence pairs linked by causal inferences (*causal pairs* such as the bottle-smashing scenario) were judged as being more plausible than sentences that fail to invite obvious causal inferences (*unrelated pairs* such as the bottle-melting scenario). Furthermore, causal pairs were also found to be more plausible than sentence pairs that invited simple attributal inferences (i.e. where some attribute of the object is mentioned in the second sentence, such as "*The bottle was pretty*"), which in turn were judged to be more plausible than inferences of temporal succession (e.g. "*The bottle sparkled*"). It has also shown that concept-coherence affects the time it takes people to make a binary (yes / no) plausibility judgement [13]. People took significantly longer to make a yes / no decision of plausibility for causal sentence pairs than

attributal sentence pairs. These studies provide specific concrete evidence that plausibility is influenced by the conceptual coherence of a situation, as shaped by the type of inferences involved.

2.2 Word-Coherence

While concept-coherence has been seen as an overarching view of what is involved in many aspects of cognition, including plausibility (e.g. [17]), very little consideration has been given to other factors that may influence plausibility judgements. More recently, evidence has been provided for the role of word-coherence in determining plausibility [18]. This view suggests that plausibility judgements are sensitive to the distributional information of the individual words used to describe a scenario. In other words, the distinctive relationship between words, as represented in distributional knowledge, can make certain scenarios seem more plausible simply by virtue of the particular words used.

Distributional knowledge of a language is gleaned through statistical analysis of large corpora that determine how each word in the language is used in relation to every other word. By moving through the corpus and counting the frequency with which a given word appears with other words in its surrounding context, a picture of the distribution of a language is formed. In this fashion, a word can be summarised as a vector – or point in high-dimensional space – showing the frequency with which is it associated with other lexemes in the corpus. Similarly, a sentence may be represented as a single point in distributional space by merging the points of individual words; for example, the Latent Semantic Analysis model (LSA; [19]) uses the weighted sum of constituent word vectors to denote tracts of text. In this way, two sentences containing words that occur in similar linguistic contexts (i.e., that are distributionally similar) will be positioned closer together in this space than two sentences containing words that do not share as much distributional information.

Manipulating the word-coherence of a description has been shown to affect the time needed to decide if a situation is plausible [13]. For example, consider these sentence pairs:

(i) *The pack saw the fox. The hounds snarled.*

(ii) *The pack saw the fox. The hounds growled.*

While (i) and (ii) essentially have the same meaning (i.e. they both invite the same inference), the different distributional properties of *snarled* and *growled* mean that the sentences of pair (i) are further apart distributionally than the sentences of pair (ii) (see Figure 1). People are faster to judge as plausible sentences of type (i) that are distributionally far apart, than sentences of type (ii) that are distributionally close together. So, word-coherence has an effect on plausibility, albeit weaker than that of concept-coherence.

3 THE KNOWLEDGE-FITTING THEORY OF PLAUSIBILITY

3.1 Making a plausibility judgement

In essence, we see plausibility as being about making what one is told fit what one knows about the world. When we ask people to make a plausibility judgement, we are essentially asking them to estimate the goodness of this fit. It is this process that we capture in our Knowledge-Fitting theory of plausibility, which is characterised by two main stages: the comprehension and assessment stages. The *comprehension* stage constructs a representation of a scenario with reference to distributional and prior knowledge. The *assessment* stage analyses this representation to ascertain its fit to prior knowledge, hence assessing its plausibility (see Figure 2). We shall now look at each of these stages in turn.

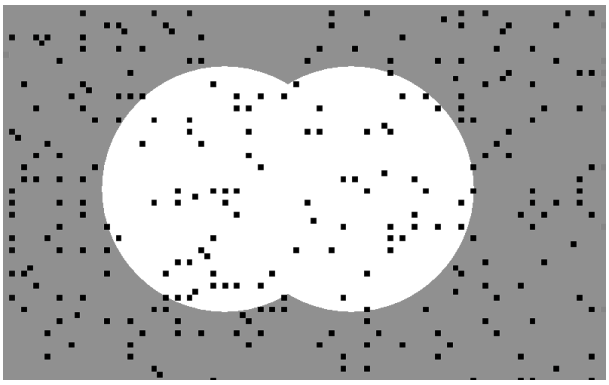


Figure 1. Illustration of the distributional spotlight; sentences that are close togher have overlapping spotlights and give smaller coverage than sentences that are far apart

3.1.1 Comprehension

Our view of the comprehension stage is not much different to that currently held in the cognitive psychology literature. Namely, that the comprehension of a scenario involves constructing a mental representation of the described situation, which is aided by cues from the linguistic input [20][21][22][23][24].

For example, in comprehending the sentences "*The bottle fell off the shelf. The bottle smashed*", the main cognitive steps could be described as follows. First, the specific words in the first sentence reference distributional knowledge and cause a spotlight to fall on a neighbourhood of related terms in a high-dimensional distributional space [19][25][26]. Each of these spotlit terms then helps to prime relevant prior knowledge in long-term memory [27]. For example, knowledge relevant to the first sentence's situation may include that bottles are often fragile, that shelves are located at a height, that fragile things break when they hit the ground, etc. When the next sentence is read, the same procedure of distributional spotlighting and knowledge priming takes place. If any of the primed knowledge is used by inferences to connect the two sentences, then it will remain primed in case it is useful again; if the primed knowledge is not used, then it will be suppressed as irrelevant [28][29].

However, the amount of knowledge primed by a sentence pair depends on the size of the sentences' distributional spotlights. If sentences are close together in distributional space, then their spotlights will overlap almost completely and the overall coverage will be small. However, if the sentences are far apart, then their spotlights will fall on separate areas the overall coverage will be large (see Figure 1). The further apart the sentences are in distributional space, the greater the spotlight coverage and the more prior knowledge is primed. This means that distributionally distant sentence pairs have an advantage over distributionally close sentence pairs, because there is a greater chance that the knowledge required by the inference will already be primed. By retrieving prior knowledge (e.g. bottles are often fragile) and making the

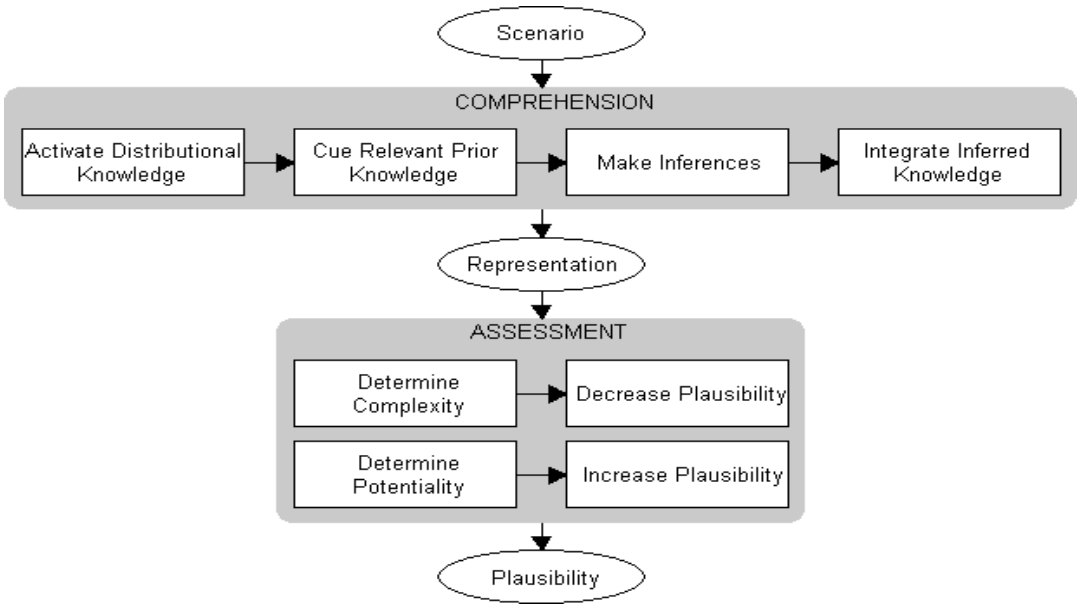


Figure 2. Diagram of processes in the Knowledge-Fitting Theory of Plausibility

relevant inferences (e.g. falling onto the floor *caused* the bottle to smash), the conceptual representation of the scenario is constructed and the sentence is said to be comprehended.

3.1.2 Assessment

Once a scenario has been comprehended, it must then undergo assessment to ascertain its plausibility. Assessment of a scenario involves examining the representation that has been produced by comprehension.

Plausibility assessment is dependent on two aspects of the representation; its *complexity* (how many inferences had to be made to connect events), and its *potentiality* (how much knowledge remains primed). For an example of the complexity aspect, with the sentences "*The bottle fell off the shelf. The bottle melted*" it might be possible to construct a representation where the bottle fell from the shelf onto the floor, which was made of metal, and which had somehow heated up enough to cause the bottle to melt. Clearly, this is not a very plausible scenario and it takes a lot of work to make it fit our knowledge about the world. The more inferences that need to be made to connect events in a scenario, and the more complex the representation grows, the less plausible the scenario becomes.

For an example of the potentiality aspect, with the sentence "*The bottle fell off the shelf. The bottle smashed*", the bottle smashed because it was fragile and hit the floor. Alternatively, the bottle may have smashed because it struck another shelf during its fall, or a table, or some other hard surface. Essentially, there are far more ways that a bottle can *smash* rather than *melt* – there are more ways that the scenario could have come about. When an individual is judging the plausibility of a scenario, it is unlikely that he or she will consider all of the possible versions. However, the possibility that more versions could potentially be constructed is reflected by the amount of knowledge that still remains primed. In both scenarios, the distributional spotlight of the first sentence ("*The bottle fell off the shelf*") primed knowledge relevant to the situation, such as that bottles are often fragile, that fragile things break when they hit the ground, etc. The melting scenario did not use this primed knowledge, and so it is suppressed. In assessing the scenario, the lack of primed knowledge suggests that there are no further versions of this scenario to be constructed and that there are no other ways in which we can make it fit our knowledge about the world. In contrast, the *smashing* scenario did make use of these examples of primed knowledge. The large amount of primed knowledge remaining suggests there are a further number of possible versions of the scenario that could be fleshed out if so desired, and suggests that there are many more ways that we could make this scenario fit our knowledge about the world. In short, the more knowledge that remains primed after the representation is built, the more potential versions of the scenario that could be constructed and the more plausible the scenario becomes.

3.2 Types of plausibility judgement

We have described how plausibility judgements are generally carried out, but we must also be aware that there are different ways in which one can judge plausibility. Recent work in [13][15][16] shows two different types of plausibility judgement task, which we shall discuss in turn.

The first type of plausibility judgement can be described as a binary plausibility decision, namely a decision of whether a scenario is plausible or not. To do this, one need only examine the complexity aspect of the representation during assessment (i.e. the number of inferences that had to be made to connect events). If the events could not be connected (e.g. no inferences could be made because of missing or contradictory information in prior knowledge), then the representation of the scenario is incomplete and the scenario will not be plausible. On the other hand, where a number of inferences can be seen to successfully connect the events, then the representation is complete and the scenario is plausible. For example, the statement that "*the balloon landed on a pin and burst*" is plausible because prior knowledge gives us the information that sharp things (the pin) can cause balloons to burst. However, saying that "*the balloon landed on a pin and melted*" is not plausible because there is little in our prior

knowledge to suggest how a balloon could melt as the result of contact with a pin.

The second type of judgement is more involved than the first, requiring someone to ascertain exactly *how* plausible a scenario is. In order to do this, one must examine both aspects of the representation during assessment – complexity (how many inferences had to be made using prior knowledge) and potentiality (how much knowledge remains primed). As we noted earlier, the example where the bottle falls from the shelf and melts is not very plausible, because of the high complexity of several inferences being needed (such as the floor bring metal, and something heating it up to a high enough temperature). On the other hand, the example where the bottle falls from the shelf and smashes is quite plausible, because of the high potentiality of how primed knowledge could lead to other explanations (such as the bottle hitting the floor, or a table, or another shelf).

3.3 Location of effects

The two stages involved in making a plausibility judgement are quite distinct, and are supported by the empirical evidence.

Word-coherence affects only the comprehension phase (when words spotlight distributional knowledge). However, concept-coherence affects both the comprehension phase (when the representation of the scenario is built with reference to prior knowledge) and the assessment phase (when the structure of the scenario's representation is examined). As discussed above, the type of plausibility judgement task determines what aspects of the representation are assessed. In the case of *plausibility decision* times [13], only the complexity aspect of the representation need be examined, so there is not too much for the assessment phase to do. This makes the plausibility decision process more dependent on the comprehension phase, which is why both word- and concept-coherence effects are evident in decision times. In contrast, the task of *plausibility rating* [15][16] means that both the complexity and potentiality aspects of the representation must be examined, so there is a lot for the assessment phase to do. This makes the plausibility rating process more dependent on the assessment phase, which is why only the concept-coherence effect is evident in ratings, as it outweighs that of word-coherence.

4 PAM: PLAUSIBILITY ANALYSIS MODEL

PAM is a computational implementation of the Knowledge-Fitting theory of plausibility. PAM implements both the comprehension and assessment phases, incorporating knowledge of word-coherence and concept-coherence to provide judgements of plausibility that reflect those made by people [12][13][16]. The model takes sentence inputs and outputs an estimated plausibility decision time (in milliseconds) and a plausibility rating (from 0-10) for the scenario described in the sentences. PAM judges plausibility using a combination of commonsense reasoning (for concept-coherence) and distributional analysis (for word-coherence).

4.1 Comprehension phase

When a sentence is first read, each word helps to spotlight a certain area of distributional knowledge. PAM models this process by the use of a model of linguistic distributional knowledge, Latent Semantic Analysis (LSA: [19]). LSA (in the form used by PAM) is a statistical model of the distributional patterns of English words, which works by passing a window over a large corpus that represents the cumulative lifetime readings of an American first-year university student[1]. PAM uses LSA to calculate the 50 words that are the nearest neighbours of each sentence (i.e. the distributional spotlights), and then unifies the two sets of words. The number of unique words covered by the spotlights depends on how far apart the sentences are in LSA's distributional space. If the sentences are very close together then their spotlights will overlap completely, giving a count of 50 unique words. However, if the sentences are very far apart then their spotlights will be completely separate, giving a count of 100 unique words. This *distributional word count* represents the word-coherence factor, and is later used in the Assessment phase as a scaling variable in estimating the plausibility decision time and rating for the presented scenario. The higher the distributional word count, the more knowledge is primed, which means faster decision times and higher perceived plausibility.

As we have said, distributional information alone is not enough to form the basis of a judgement of plausibility; we also need a conceptual representation of the scenario. To do this, PAM breaks down each sentence into propositional form and makes the inferences between the sentences by fitting their propositions to information in the knowledge base.

For example, the sentence "*the pack saw the fox*" is represented as *see(pack, fox*). PAM must therefore check the predicate *see* in the knowledge base and determine if the arguments meet the conditions specified. The *see* predicate requires that its first argument is an animal (i.e. something must be an animal in order to see), and since the definition of pack shows that it contains dogs, and the type hierarchy for dog shows that it is an animal, the first condition of the see predicate is met. Also, the see predicate requires that its second argument is a non-abstract entity (i.e., something must be non-abstract in order to be seen). Since the type hierarchy of fox shows that it is an animal and not an abstract entity, the second condition of the see predicate is met. The way in which each condition is met is listed, and if all conditions are fulfilled, PAM returns this list as a path

When the first sentence has been represented, PAM moves onto processing the second sentence. The sentence "*the hounds growled*" is broken down into propositional form as *growl(hounds)* and PAM than searches for ways to meet the conditions of the *growl* predicate. The first condition is that the argument (hounds) must be an animal, which is easily met. However, there are other conditions as to *why* the hounds are growling, such as because they are generally aggressive, or because they are fighting amongst themselves. Some of these conditions lead to other predicates which have their own conditions attached, such as *hunt(hounds)* which requires that hounds must be predators and that the *fox* of the first sentence must be prey. It is likely that there are many different paths in the knowledge base that could be followed to fulfil the conditions of the *growl* predicate, and PAM will record them all. In this respect, PAM models group behaviour in plausibility judgement; rather than limit the representation to a single path that one individual may consider, PAM represents the set of paths that a group may consider and averages out the differences.

---

[1] In LSA parlance, the analysis was done in the 'General Reading up to 1st Year College' semantic space, with pseudodoc comparison at maximum factors. In order to exclude misspellings and other very low frequency words, any words with a frequency in the corpus of less than 5 were excluded.

4.2 Assessment phase

When the comprehension phase is completed, it is the role of the assessment phase to analyse the structure of the path representation in order to estimate the plausibility decision time and to calculate the plausibility of the scenario.

PAM extracts three important variables from the representation:

1. Total Number of Paths *P* (the number of different ways the sentence conditions can be met in the knowledge base)
2. Mean Path Length *L* (the average count of how many different conditions must be met per path)
3. Proportion of "Hypothetical" Paths *H* (proportion of all paths that contain a condition that was only met by assuming the existence of something not explicitly mentioned – e.g. [*The bottle fell off the shelf. The bottle melted.*] is considered a plausible path if we assume a hypothetical furnace for the bottle may to fall into)

4.2.1 Plausibility decision times

In estimating the time needed to decide if a scenario is plausible or not, PAM uses the above variable *L* to calculate the concept-coherence of the scenario. A high path length (*L*) means a longer decision time, because more elaborate requirements must be met to verify the sentence. In addition, the comprehension time of a sentence is affected by syllable length, and to a lesser extent orthographic length, so PAM increases the decision time estimate as each of these increase.

Word-coherence has a strong effect on plausibility decision times, and PAM uses the distributional word count (calculated in the Comprehension phase) to model this. If the two sentences in the pair are very far apart, then they will have the maximum distributional word count of 100. However, if the two sentences are very close to each other, then they will have the minimum distributional word count of 50. The higher the distributional word count, the more knowledge that the sentences prime, and the faster the plausibility decision time becomes. PAM therefore uses the distributional word count to scale down the estimated response time.

It has been demonstrated that using this approach allows PAM to produce estimates of plausibility decision times that correlate highly with human responses (r=0.633, r^2=0.401, p<0.0001, N=60) [12]. The human data modelled was taken from [13], and is graphed against PAM's output in Figure 3.

4.2.2 Plausibility ratings

PAM uses the three variables above to calculate the concept-coherence of the scenario, and return a rating between 0 (not plausible) and 10 (completely plausible). In short, a high number of paths (*P*) means higher plausibility, because there are more possible ways that the scenario can be represented. A high mean path length (*L*) means lower plausibility, because elaborate requirements must be met to verify the sentence. Finally, a high proportion of hypothetical paths (*H*) means lower plausibility, because it assumes the existence of entities that may not be present.

When the path rating has been computed, PAM applies word-coherence (the distributional word count calculated in the comprehension phase) as a scaling variable. The magnitude of this scaling is less than that of other variables, but still has a perceptible effect. In this way, PAM models the small difference in plausibility ratings found between versions of sentence pairs that vary in their distributional distance but are conceptually identical (see pairs i and ii).

Table 1. Mean Plausibility ratings per inference type from human raters and from PAM.

Inference Type	Human Rating	PAM Rating
Causal	7.8	7.9
Attributal	5.5	5.7
Temporal	4.2	5.0
Unrelated	2.0	0.9

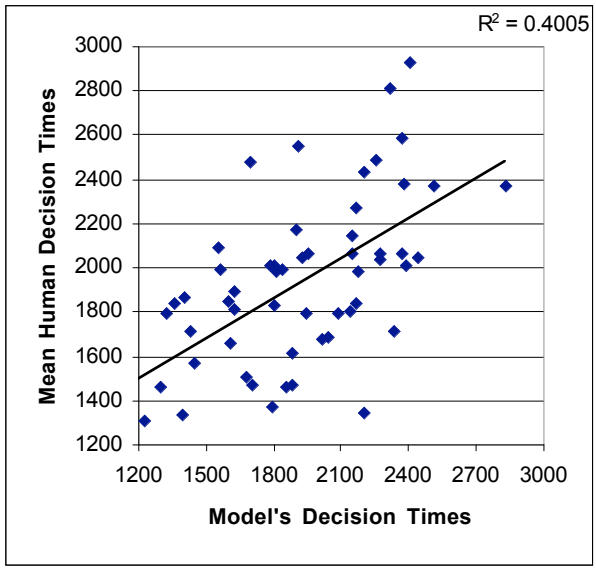


Figure 3. PAM's performance against human responses in modelling plausibility decision times

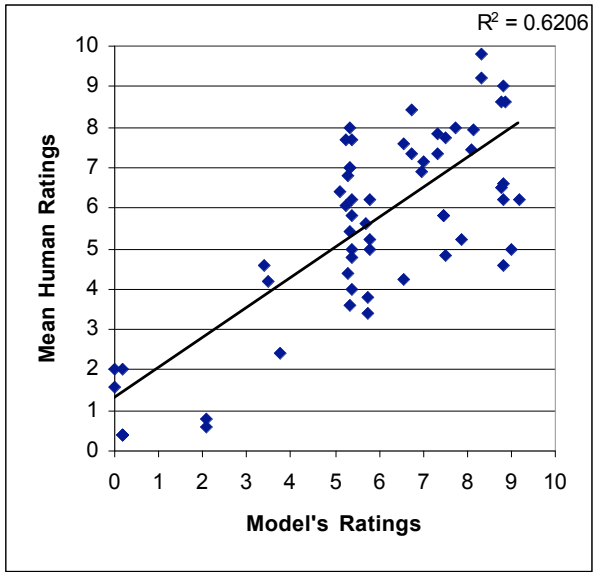


Figure 4. PAM's performance against human responses in modelling plausibility ratings

It has been demonstrated that using this approach, PAM can produce plausibility ratings that correlate highly with human plausibility judgements (r=0.788, r^2=0.621, p<0.0001, N=60) [12]. The human data modelled was taken from [16], and is graphed against PAM's output in Figure 4. Additionally, Table 1 illustrates mean ratings for scenarios that invite different types of bridging inference, comparing those produced by people to those ratings produced by PAM.

5 DISCUSSION

In this paper we have presented the new Knowledge-Fitting Theory of Plausibility and the Plausibility Analysis Model (PAM), which is the first account to specifically and accurately address human plausibility judgements. Our theory of plausibility can explain the some counterintuitive empirical findings regarding word- and concept-coherence, and provides simulations for testing in further studies.

PAM is the computational implementation of the Knowledge-Fitting theory, and as such is the first cognitive model of human plausibility judgements. The importance of word- and concept-coherence in people's plausibility judgements is clear, and in a novel paradigm, we have integrated both these factors in our model. Future work in the field of plausibility must also take account of both distributional knowledge and conceptual prior knowledge, as well as the interactions between them.

ACKNOWLEDGMENTS

This work has been part-funded by a grant from the Irish Research Council for Science, Engineering and Technology.

REFERENCES

[1] P. Lemaire and M. Fayol. When plausibility judgments supersede fact retrieval: The example of the odd-even rule effect in simple arithmetic. *Memory and Cognition, 23*, 34-48, (1995).

[2] L. M. Reder. Plausibility judgments vs. fact retrieval: Alternative strategies for sentence verification. *Psychological Review, 89*(3), 250-280, (1982).

[3] L. M. Reder and B. H. Ross. Integrated knowledge in different tasks: The role of retrieval strategy on fan effects. *Journal of Experimental Psychology: Learning, Memory and Cognition, 9*, 55-72, (1983).

[4] L. M. Reder, C. Wible and J. Martin. Differential memory changes with age: Exact retrieval versus plausible inference. *Journal of Experimental Psychology: Learning, Memory and Cognition, 12*(1), 72-81, (1986).

[5] M. J. Pickering and M. J. Traxler. Plausibility and recovery from garden paths: An eye-tracking study. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 24*, 940-961, (1998).

[6] S. R. Speer and C. Clifton. Plausibility and argument structure in sentence comprehension. *Memory and Cognition, 26*(5), 965-978, (1998).

[7] M. J. Traxler and M. J. Pickering. Plausibility and the processing of unbounded dependencies: An eye-tracking study. *Journal of Memory and Language, 35*, 454 - 475, (1996)

[8] A. Collins and R. Michalski. The logic of plausible reasoning: A core theory. *Cognitive Science, 13,* 1-49, (1989).

[9] E. E. Smith, E. Shafir and D. Osherson. Similarity, plausibility, and judgments of probability. *Cognition, 49*, 67-96, (1993).

[10] F. Costello and M. T. Keane. Efficient Creativity: Constraints on conceptual combination. *Cognitive Science, 24,* 299-349, (2000).

[11] F. Costello and M. T. Keane. Alignment versus diagnosticity in the comprehension and production of combined concepts. *Journal of Experimental Psychology: Learning, Memory & Cognition, 27*, 255-271, (2001).

[12] L. Connell and M. T. Keane. PAM: A Cognitive Model of Plausibility. *Proceedings of the Twenty-Fifth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum, (2003).

[13] L. Connell and M. T. Keane. The knowledge-fitting theory of plausibility. *Manuscript in preparation.*

[14] A. Black, P. Freeman and P. N. Johnson-Laird. Plausibility and the comprehension of text. *British Journal of Psychology, 77*, 51-60, (1986).

[15] L. Connell and M. T. Keane. The roots of plausibility: The role of coherence and distributional knowledge in plausibility judgements. *Proceedings of the Twenty-Fourth Annual Conference of the Cognitive Science Society* (p. 998). Hillsdale, NJ: Erlbaum, (2002).

[16] L. Connell and M. T. Keane. What plausibly affects plausibility? Concept-coherence and distributional word-coherence as factors influencing plausibility judgements. *Manuscript in submission.*

[17] P. Thagard. Coherence in thought and action. Cambridge, MA: MIT Press, 2000.

[18] M. Lapata, S. McDonald and F. Keller. Determinants of adjective-noun plausibility. *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics*, (pp. 30-36). San Mateo, CA: Morgan Kaufmann, (1999).

[19] T. K. Landauer and S. T. Dumais. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review, 104*, 211-240, (1997).

[20] M. A. Gernsbacher. Language comprehension as structure building. Hillsdale, NJ: Erlbaum, (1990).

[21] M. Singer, A. C. Graesser and T. Trabasso. Minimal or global inferences during reading. *Journal of Memory and Language*, 33, 421-441, (1994).

[22] P. N. Johnson-Laird. *Mental Models*. Cambridge: Cambridge University Press, 1983.

[23] G. McKoon and R. Ratcliff. Inference during reading. Psychological Review, 99, 440-466, (1992).

[24] R. A. Zwaan and G. A. Radvansky. Situation models in language comprehension and memory. *Psychological Bulletin*, 123, 162-185, (1998).

[25] C. Burgess, K. Livesay and K. Lund. Explorations in context space: Words, sentences, discourse. *Discourse Processes, 25*, 211-257, (1998).

[26] K. Lund, C. Burgess and R. A. Atchley. Semantic and associative priming in high-dimensional semantic space. *Proceedings of the 17th Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Erlbaum, (1995).

[27] M. Halldorson and M. Singer. Inference processes: integrating relevant knowledge and text information. *Discourse Processes, 34*, 145-161, (2002).

[28] M. A. Gernsbacher and M. Faust. The role of suppression in sentence comprehension. In G.B. Simpson (Ed.), *Understanding word and sentence.* Amsterdam: North Holland, 1991.

[29] M. A. Gernsbacher and R. W. Robertson. Reading skill and suppression revisited. *Psychological Science, 6*, 165-169, (1995).

Lifetime Learning in Multi–Agent Systems: Robustness in Changing Environments

Dara Curran and Colm O'Riordan
Dept. of Information Technology,
National University of Ireland, Galway.

Abstract. Dynamic behaviour in response to a dynamic environment is a highly desirable quality in a multi–agent system. This paper outlines a genetic algorithm and neural network approach to the simulation of both population and lifetime learning. Experiments are carried out on populations of agents that are placed in environments which may change suddenly or gradually. The effects of population and lifetime learning are presented and discussed.

1 INTRODUCTION

The two primary evolutionary forces in nature can be described as population learning and lifetime learning. The first considers the effect of genetic inheritance and its role in the behaviour of living things. Populations of creatures evolve through the processes of genetic recombination and mutation. Over long periods, genetic encodings emerge which produce phenotypic traits suitable for a particular environment, such as webbed feet or enhanced vision, giving individuals competitive advantage over others. As generations progress, useful traits will be passed down successive populations resulting in an overall fitness improvement.

Lifetime learning represents each individual's ability to interact and learn from its environment. Creatures which are capable of correctly interpreting novel situations will be more likely to survive the un–predictability of many habitats. Furthermore, creatures which are capable of memory will recall previous errors in similar situations, greatly reducing the risk of repeated damaging behaviour. This learning mechanism is also a driving force of evolution: as a result of fitness gains brought by lifetime learning, the number of creatures capable of lifetime learning increases and the population improves its overall performance.

The ability for organisms to withstand changes and adapt to their environment is a fundamental aspect of the evolutionary process. Both population and lifetime learning contribute to a species' survival by providing different mechanisms for adaption. Of the two evolutionary forces, lifetime learning is intuitively more suited to adaptation and plasticity – a creature is more likely to survive if it is capable of learning from its environment in its own lifetime rather than rely on an encoding of suitable behaviour to occur at a genetic level. From an agent point of view, a population capable of dynamic behaviour is of great benefit, particularly in problem domains where environments are not discrete or predictable.

These evolutionary forces can be simulated using genetic algorithms and neural networks respectively. Genetic algorithms represent potential problem solutions as genetic codes which are then evaluated for fitness. Pairs of codes are selected in proportion to their fitness and are combined together to produce offspring. These offspring become part of the next generation and the process is repeated. Genetic algorithms have been shown to be useful in a vast variety of problem domains [1, 2].

Neural networks are simplified mathematical models of nervous systems, inspired by the neurons and synapses of living creatures. Neural networks function by reading input patterns from specified input neurons, feeding these pattern values through a succession of weighted synapses linking other neurons, and finally displaying output values at specified output neurons. By examining a network's output pattern and the desired output pattern for a given input, a measure of error can be obtained. This error is then used to alter the weighting value of synapses connecting neurons in the network. One algorithm for performing this weighting adjustment is known as error back propagation. Through a series of training iterations, the overall error of a network is reduced, improving its performance.

The combination of genetic algorithms and neural networks provides a framework for evolutionary experimentation popular in much research including language evolution [3], neural network design optimization [4, 5, 6] and games [7, 8]. To combine the two approaches, the structure of the neural network must be converted to a format which is suitable for the genetic algorithm. For these experiments, an artificial life simulator developed in our previous work [9, 10] was used which employs a process known as marker based encoding [7] to generate genetic encodings of neural network structures and allows both lifetime and population based learning.

The purpose of this paper is to examine the effects of dynamic environments on a population of agents employing population and lifetime learning. Experiments using both sudden and gradual changes in environment are presented. Section 2 briefly outlines the structure and functionality of the artificial life simulator. Section 3 describes the experimental setup used for each experiment. Section 4 shows the results obtained and finally Section 5 presents a conclusion.

2 SIMULATOR

The architecture of the artificial life simulator can be seen as a hierarchical structure. At the top level of the simulator is a command interpreter which allows users to define an experiment's variables including the number of networks, the number of generations to run the experiment, mutation and crossover rates and the actual problem set which the population will be attempting to solve.

The neural network layer takes the set of variables using the command interpreter and initializes a given number of neural networks. It then performs training and testing of the networks according to the parameters of the experiment. These network memory structures are then passed to the encoding layer which transforms them into genetic code structures for use in the genetic algorithm.

The genetic algorithm layer uses the genetic codes and the data retrieved from the neural network layer's testing of the networks to perform its genetic operators on the population. A new population is produced in the form of genetic codes. These are passed to the decoding layer which transforms each code into a new neural network structure. These structures are then passed up to the neural network layer for a new experimental iteration. Once the required number of generations has been reached the experiment finishes.

3 EXPERIMENTAL SETUP

A population of 200 agents is randomly initialized at the beginning of each experiment. The population is then allowed to forage for food items, and each agent is allocated fitness according to how well it distinguishes between food and poison bit patterns. The food and poison bit patterns used in this set of experiments are the 16 bit patterns of the 4–bit parity problem thus generating an equal number of food and poison elements in the population's environment. Two experiments are presented here - the first examining the population's performance in the face of gradual changes in food and poison patterns and the second observing the population's behaviour in an environment which suddenly changes completely. Each experiment is divided in two parts – one using lifetime learning and the other does not. In order to obtain reliable results, each experiment is carried out 20 times to derive an average population performance. The crossover rate for each experiment was set at 0.6 and the mutation rate at 0.02. Each experiment allowed populations to evolve for 300 generations.

3.1 Gradual Environment Change

In order to simulate a gradual change in the environment, the bit patterns representing food and poison must be altered over time. The approach taken for this experiment is to change one of the sixteen food or poison values every 19 generations. Thus, patterns which once represented food items become poison and the makeup of the environment eventually reverses with respect to its initial parameters. At the end of the experiment run, all patterns which represented food now represent poison and vice–versa. The goal of this experiment is to ascertain to what degree the population is capable of recovering from each iterative change in its environment.

3.2 Sudden Environment Change

The second experiment examines the behaviour of the population in the face of a dramatic change in food and poison patterns. After a period of 150 generations, each of the sixteen patterns is altered simultaneously to represent its opposite, meaning that all food becomes poison and vice-versa - a potentially catastrophic event for the population. The goal of this experiment is to observe whether such a change will force a population into virtual extinction or whether the population is robust enough to recover.

4 RESULTS

This section presents the results for the experiments described in Section 3, divided into gradual and sudden environmental change.

4.1 Gradual Environment Change

The results of the first experiment, gradual change without lifetime learning, are illustrated in Figure 1. Each change iteration can be clearly seen in the graph as a sudden plunge in fitness at the change generation. This plunge is immediately followed by quite steep fitness gains suggesting that the population does not take long to recover from such events. The level of fitness peaks at around generation 160 before a gradual descent in fitness. This descent is characterized again by sharp falls in fitness but with subsequent climbs slowing down, resulting in an overall loss.

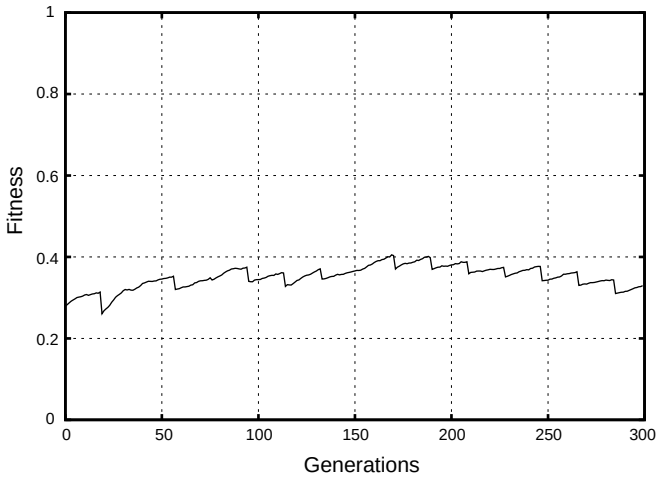


Figure 1. Gradual Change - No Lifetime Learning

It is interesting to note that the population peaks at around the half–way mark in the experiment. At this point 8 of the 16 food and poison patterns have been reversed. Therefore, the first eight patterns represent an inverse of the XOR problem, while the second half represents the original XOR problem. It is possible that such a problem set is easier to solve than either the full XOR or the full inverse XOR problems, thereby resulting in higher population fitness. This would explain the gradual drop in fitness following the half–way mark.

As the problem begins to resemble the full inverse XOR problem more and more, the population is increasingly unable to solve the problem.

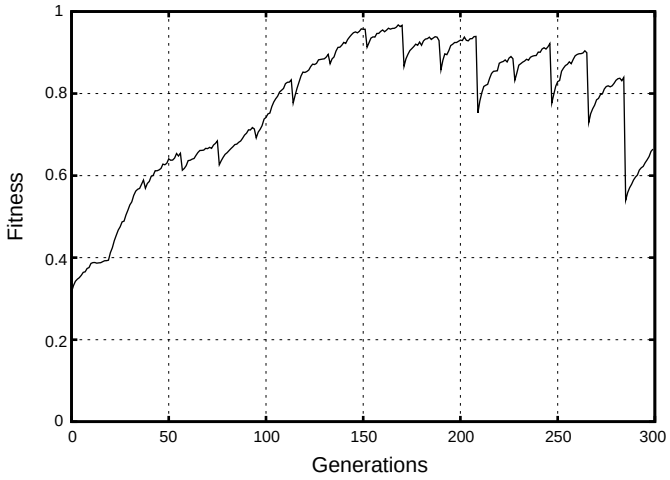


Figure 2. Gradual Change - Lifetime Learning

The second experiment immediately shows the gains produced by applied lifetime learning (Figure 2). The population achieves much higher levels of fitness than in the previous experiment. The first half of the experiment does not show the same sudden drops in fitness (although smaller ones still occur) that were present in the previous experiment results implying that lifetime learning is softening the effects of these changes. Once again, the population fitness peaks at around the half–way mark, obtaining near optimum levels. Past this level, the drops in fitness become more and more acute, as the problem becomes more difficult for the networks to solve. The gains in fitness after each of these drops is significant however, suggesting that given more time between changes, the population would be able to recover.

4.2 Sudden Change

The first sudden change experiment results are illustrated in Figure 3. The population seems capable of achieving moderate levels of fitness without lifetime learning, but the fitness increase slows down considerably as the experiment nears the half-way mark. The sudden change in environment at generation 150 causes a disastrous drop in fitness, down to around 0.15 – much lower than even the randomly generated initial population at generation 1. This low level indicates that the networks are completely incapable of reacting to a change of this magnitude. The population then begins quite a sharp recovery which slows down considerably after generation 200 and stagnates to a level which is considerably lower than the one previously achieved.

The second experiment again shows the benefit of introducing lifetime learning in a population faced with changing environments. The graph in Figure 4 shows a steady rise in fitness up to generation 150 with a fitness level already surpassing that of the previous experiment. While the drop in fitness at generation 150 is significant, it is not nearly as disastrous as

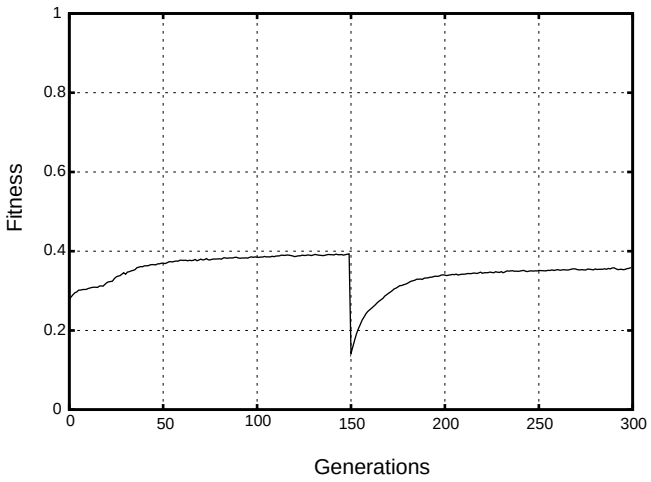


Figure 3. Sudden Change - No Lifetime Learning

the one seen previously. Compared to a drop of nearly 0.3 in the last experiment, the population's fitness falls a mere 0.07 – showing that the population has been able to adjust to the new problem without the need for any genetic change. The fitness levels quickly rise again and surpass the level obtained prior to the drop, continuing to increase steadily.

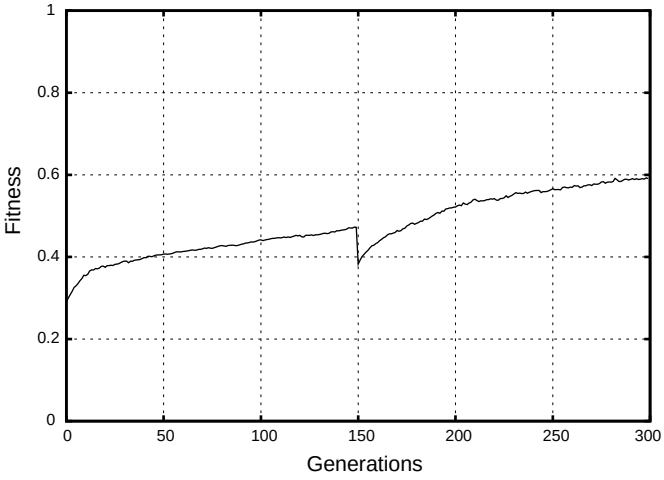


Figure 4. Sudden Change - Lifetime Learning

5 CONCLUSION

These results clearly outline the robustness of a population endowed with the ability to learn from its environment. In each experiment, populations capable of learning were able to sustain their fitness growth. Perhaps the more realistic gradually changing environment proved more of a challenge to the populations. As the experiments near their centre point, both learning and non—learning populations achieved their fitness peaks. This might imply that the evolved network architectures happen to be suitable for the particular problem at this

point. In such an environment, selection would ensure that networks which exhibit the ability to solve similar problems to a reasonable level would propagate through the population. However, once the problem begins to become more difficult, these networks become less valuable as their ability to perform well is impaired. The absence of networks capable of solving the new problem causes the decrease in fitness witnessed in both experiments. The sudden change experiments highlight the superiority of the lifetime learning approach. Not only is the drop in fitness far less severe in the lifetime learning experiment, but the fitness continues to rise steadily after this point. Perhaps it is easier for evolutionary neural network populations to start afresh with a completely different problem set rather than a continually shifting one. Clearly, an environment which is constantly changing presents far more difficulties to the evolutionary process than one which changes, even dramatically, only rarely. However, it is clear from both experiment sets that the ability to learn during one's lifetime is beneficial to the population as a whole in both types of environment.

ACKNOWLEDGEMENTS

This research is funded by the Irish Research Council for Science, Engineering and Technology.

REFERENCES

[1] J. H. Holland. *Adaptation in Natural and Artificial Systems.* Ann Arbor MI: The University of Michigan Press, 1975.

[2] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, MA, Addison-Wesley, 1989.

[3] B. MacLennan. Synthetic ethology: An approach to the study of communication. In *Artificial Life II: The Second Workshop on the Synthesis and Simulation of Living Systems, Santa Fe Institute Studies in the Sciences of Complexity*, pages 631–635, 1992.

[4] D. Whitley, T. Starkweather, and C. Bogart. Genetic algorithms and neural networks - optimizing connections and connectivity. *Parallel Computing*, 14(3):347–361, Aug 1990.

[5] R. K. Belew, J. McInerney, and N. N. Schraudolph. Evolving networks: Using the genetic algorithm with connectionist learning. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 511–547. Addison-Wesley, Redwood City, CA, 1992.

[6] X. Yao. Evolving artificial neural networks. *PIEEE: Proceedings of the IEEE*, 87:1423 – 1447, 1999.

[7] D. Moriarty and R. Miikkulainen. Discovering complex othello strategies through evolutionary neural networks. *Connection Science*, 7(3–4):195–209, 1995.

[8] Norman Richards, David E. Moriarty, Paul McQuesten, and Risto Miikkulainen. Evolving neural networks to play go. In Thomas Bäck, editor, *Proc. of the Seventh Int. Conf. on Genetic Algorithms*, pages 768–775, San Francisco, CA, 1997. Morgan Kaufmann.

[9] D. Curran and C. O'Riordan. On the design of an artificial life simulator. In R.J.Howlett V.Palade and L.C.Jain, editors, *Proceedings of the Seventh International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES 2003)*, University of Oxford, United Kingdom, 2003.

[10] D. Curran and C. O'Riordan. Artificial life simulation using marker based encoding. In *Proceedings of the 2003 International Conference on Artificial Intelligence (IC-AI'03)*, volume II, pages 665–668, Las Vegas, Nevada, USA, 2003.

Evolving Crossover, Mutation and Training rates in a Population of Neural Networks

Dara Curran and Colm O'Riordan
Dept. of Information Technology,
National University of Ireland, Galway.

Abstract. This paper describes a method of determining the rates of crossover, mutation and training employed in the evolution of a population of neural networks. The genetic codes of the population are modified to include rate data which evolves with the population to attain optimum levels. We compare these results to experiments performed to determine optimum rate values by trial and error.

1 INTRODUCTION

The combination of genetic algorithms and neural networks has been shown to be successful in a variety of problem domains [1, 2, 3, 4]. The evolutionary approach of genetic algorithms has been shown to guide the neural network's data processing capability to high levels of performance. In a typical implementation, the genetic algorithm is employed to generate a population of random neural network architectures, each of which is trained and tested to evaluate its performance. The genetic codes of successful networks are then combined to create the next generation. As the genetic algorithm can be designed to select particular network characteristics (such as small number of nodes and links), the approach yields very efficient neural network architectures with little or no human intervention.

The genetic algorithm uses several operators including crossover and mutation. Crossover allows offspring to inherit portions of each parent's genetic code, thus probing new search spaces while the mutation operator modifies the offspring's genetic code in order to both open up un-explored search spaces and slow the process of stagnation.

Typically, the behaviour of both these operators is determined by a rate, usually expressed as a percentage or probability. A crossover rate of 0.75, for instance, means that when any two agents mate, their genetic codes will undergo the process of crossover with a probability of 75%. Similarly, a mutation rate of 0.5 indicates that each bit in the offspring's genetic code will be modified with probability 50%. These rates seriously affect a population's performance. For instance, a setting of 0.00 crossover, would create a stagnant population of cloned networks, while a population evolved using a mutation rate of 0.8 would exhibit very erratic behaviour as a result of constant genetic changes.

The third rate examined in this paper is that of training. Each network is given the opportunity to reduce its error via learning using error back propagation. Generally speaking, the more a network is trained, the better it will perform its task — although, over–training is possible.

It can be difficult to determine values for any of these rates in advance of a given experiment. Usually several trial and error attempts must be made before an appropriate setting is found. This paper presents a method of evolving optimal rate values for crossover, mutation and training for a population of neural networks. All experiments assume that rates are independant and that optimum values can be determined for each individually. Future work will concentrate on experiments attempting to evolve all rates simulatenously.

The next section of the paper describes related research which has been performed. Section 3 briefly describes the artificial life simulator used in this set of experiments. Section 4 presents results for the first set of experiments, where rates are determined in an iterative manner. Section 5 describes the method employed for the evolutionary rate determination experiments as well as their results. Finally, Section 6 ends with a conclusion and an indication of future work.

2 RELATED WORK

The combination of neural networks and genetic algorithms originally stemmed from the desire to generate neural network architectures in an automated fashion using genetic algorithms [1, 2]. The advantage of this approach is that neural networks can be selected according to a variety of criteria such as number of nodes, links and overall accuracy. The neural network component, on the other hand, provides computational functionality at an individual level within the genetic algorithm's population. The combination of genetic algorithms and neural networks has since been proven successful in a variety of problem domains ranging from the study of language evolution [3] to games [4].

Some research has also been performed on the evolution of crossover in a genetic algorithm, notably by Spears [5]. Experiments were performed to allow populations to determine the crossover type to employ for a given problem. The populations were allowed to choose between 1–point and 2–point crossover. This was achieved by appending extra bits to each genetic code in the population to represent either 1 or 2–point crossover. As the population evolved, individuals carrying the most successful crossover type emerged as dominant. Thus, at the end of each experiment it was possible to observe the evolved preference to each crossover method.

3 ARTIFICIAL LIFE SIMULATOR

The experiments outlined in this paper were performed using a previously developed artificial life simulator [6, 7, 8]. The simulator allows populations of neural networks to evolve using a genetic algorithm and each network can also be trained during each generation of an experiment to simulate life–time learning.

The mapping of neural network to genetic code required for the genetic algorithm is achieved using a modified version of marker based encoding. This allows networks to develop any number of nodes and interconnecting links, giving a large number of possible neural network architecture permutations.

Marker based encoding represents neural network elements (nodes and links) in a binary string. Each element is separated by a marker to allow the decoding mechanism to distinguish between the different types of element and therefore deduce interconnections [9, 10].

In this implementation, a marker is given for every node in a network. Following the node marker, the node's details are stored in sequential order on the bit string. This includes the node's label and its threshold value. Immediately following the node's details, is another marker which indicates the start of one or more node–weight pairs. Each of these pairs indicates a back connection from the node to other nodes in the network along with the connection's weight value. Once the last connection has been encoded, the scheme places an end marker to indicate the end of the node's encoding.

The networks undergo various stages throughout their lifetime. First, the gene codes are decoded to create their neural network structure. Training is then performed using error back–propagation for a given number of iterations (training cycles). Each network is tested to determine its fitness and the population is ranked using linear based fitness ranking. Roulette wheel selection is employed to generate the intermediate population. Crossover and mutation operators are then applied to create the next generation.

Since the goal of this paper is to demonstrate rate evolution, not complex problem solving, the problem set chosen for these experiments is 3–bit parity, a problem of modest complexity but which allows numerous experiments to be undertaken in a short space of time.

4 EXPERIMENTAL RATE DETERMINATION

These experiments were designed to give an approximation of the optimal rates of crossover, mutation and training. Each of the rates was modified for each experiment to observe to what degree each rate affects the population's performance. The experiments use 500 networks which evolve for 500 generations. The rates for each experiment were set as follows:

- Crossover Experiment — Mutation set at 0.02, training cycles set at 10.
- Mutation Experiment — Crossover rate set at 0.5, training cycles set at 10.
- Training Experiment — Crossover rate set at 0.5, mutation set at 0.02.

Each experiment was carried out 20 times to ensure an accurate estimation of each of the rates.

4.1 Crossover

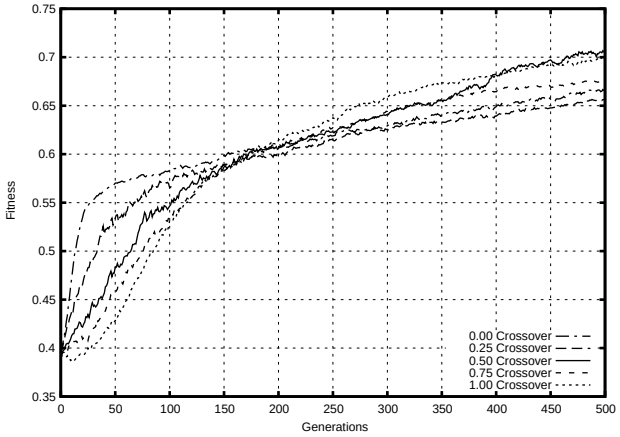


Figure 1. Crossover Rate Determination

The chosen encoding scheme does not permit crossover to operate at the bit level as this could result in the generation of invalid gene codes. Therefore the crossover points are restricted to specific intervals — only whole node or link values may be crossed over.

These results show that at certain levels, the crossover operator begins to negatively affect the population's performance (fig.1). At low levels of crossover this can be interpreted easily: network offspring are often merely clones of their parents thereby impeding the progress of the genetic algorithm. It is interesting to note that the highest level of fitness achieved occurs when crossover is set at 0.5, narrowly out–performing the 1.0 crossover rate. The fact that the fitness drops considerably when crossover is set at 0.75 would seem to indicate that the optimal level is somewhere between 0.5 and 0.75. Thus, allowing crossover to occur too often has negative effects on the population's fitness. This may be because valuable individuals are sometimes lost in the process of crossover and the population is unable to achieve high levels of fitness as a result.

4.2 Mutation

As a result of the encoding scheme employed, the mutation operator in the artificial life simulator may only operate at the weight encoding level and bit–level mutation is not possible as it could result in the generation of invalid gene codes. Instead, weight encodings are modified by a percentage in the range -200% to +200%.

Since mutation is essentially a disruptive element in the genetic algorithm, one would expect that high levels of mutation would cause a decrease in the population's fitness. This expectation appears to be justified by the results presented in fig. 2. The 0.02 mutation rate appears to be the best available option, as it performs better than no mutation and pushed to higher levels, the population's fitness begins to fall. However, 0.02 mutation may not be necessarily the optimum level. It is

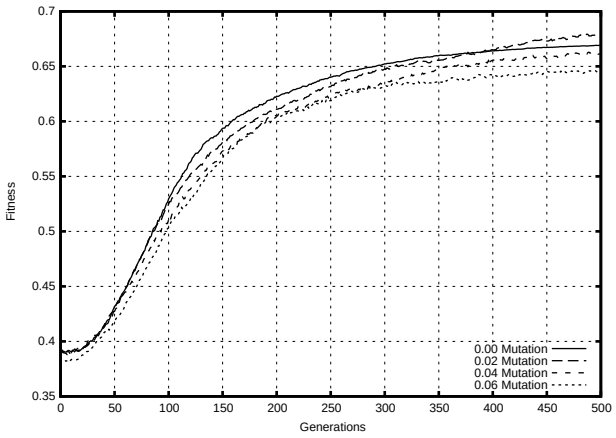


Figure 2. Mutation Rate Determination

possible that a better performance could be obtained by setting the mutation level at between 0.00 and 0.02 or between 0.02 and 0.04.

4.3 Training Cycles

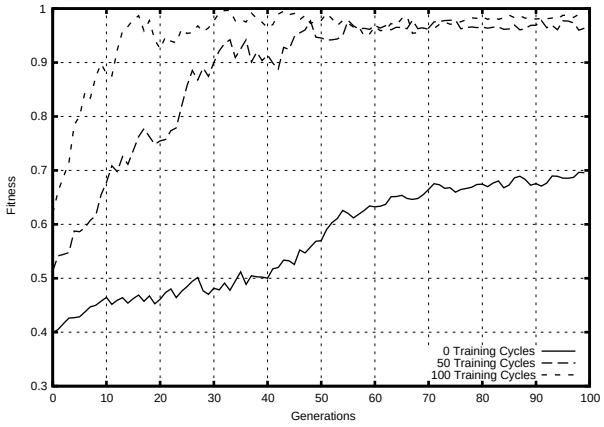


Figure 3. Training Rate Determination

The training cycle rate determines how often error back propagation is performed on a neural network. As is evident from fig. 3, the population performs better as more training is applied. It would seem that the more training is applied to the population, the faster the initial fitness jump. In the case of 50 training cycles the population's fitness reaches 0.9 in 30 generations. When the training cycles are increased to 100, the population reaches the same level in a mere 12 generations. It is interesting to note, that despite this increased acceleration, the population's fitness converges at generation 60 for both these rates. Only after generation 75 does the additional training finally overtake the 50 training cycle graph. It would appear from these results that 50 training cycles should be ample to give the population a high level of fitness.

5 EVOLUTIONARY RATE DETERMINATION

This set of experiments was designed to allow populations to adapt the rates of crossover, mutation and training to determine the optimal level of each. In order to achieve this goal, changes were made to the structure of the genetic codes of each member of the population to allow the incorporation of rate data.

At the beginning of an experiment, randomly determined rate values are appended to each individual's genetic code. These values are integers in the range $0 - 100$ and are encoded into 6–bit strings. The encoded rate values undergo separate crossover and mutation processes following those of the main genetic string and using the same probability values.

The behaviour of each rate value varies according to its operator. When two networks are selected as parents, the crossover rate values for each is examined and an average is taken to determine the crossover rate to be used. The parent gene codes will then undergo crossover according to this probability value.

The mutation operator is used on the offspring gene codes generated following the crossover process. The operator uses the inherited rate value of an individual offspring to determine its mutation rate. Each weight encoding in the offspring's gene code will be modified according to this probability value.

The training rate is also inherited and represents the number of training cycles that a neural network receives. At each cycle, the network's error is reduced using error back propagation, so in general, the more cycles a network receives, the better it performs.

Enough randomly generated networks must be produced in order to create a sufficiently large spread of values to allow potentially beneficial values to propagate across generations. The population size must also be small enough to allow the experiments to be undertaken in a reasonable length of time. After several trial runs, it was determined that the optimum population size to set the experiments was 500.

As the population evolves, individuals are selected according to their ability to solve the given problem. At the same time, rate values which they have inherited from past generations also evolve. Values that cause networks to perform poorly (such as excessively high mutation rates) should become extinct from the population as the badly performing individual is less likely to be selected for mating. Similarly, values which create the least disruption and the most benefit to an individual's performance will become more prominent in the population.

5.1 Experiment setup

500 networks are randomly generated at the start of each experiment to allow a suitably larger spread of rate values to be present in the initial population. The networks are then allowed to evolve to solve the 3–bit parity problem for 100 generations. The crossover rate was set to 0.5, the mutation rate to 0.02 and training at 10 training cycles, where these rates where not the focus of the experiment. Each experiment was carried out 20 times.

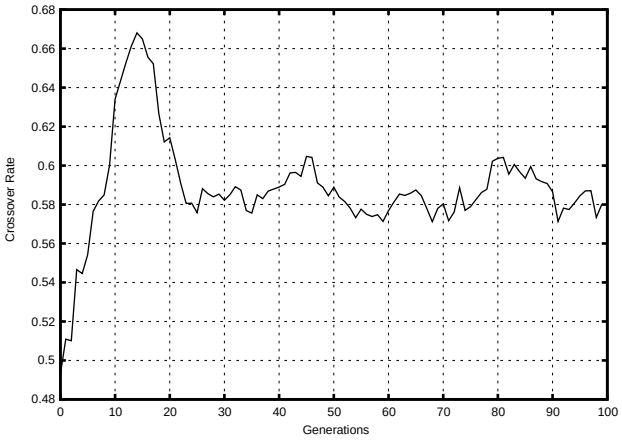


Figure 4. Crossover Rate Evolution

5.2 Evolution of Crossover Rates

The results for the evolution of the crossover rate are illustrated in fig. 4. The population begins with an average crossover rate of 0.48 which quickly ascends to over 0.66. However, the rate begins to fall sharply before stabilising somewhat at around 0.58—0.59. The initial sharp rise is most likely a result the initial population's poor fitness — a high crossover rate is useful in such situations because it allows more comprehensive probing of the search space to occur. In response to an increase in population fitness, the crossover rate then drops as exhaustive probing is no longer desirable. The final crossover rate evolved by the population seems to fit with the predicted rate of crossover (between 0.5 and 0.75) determined in the previous experiment set.

5.3 Evolution of Mutation Rates

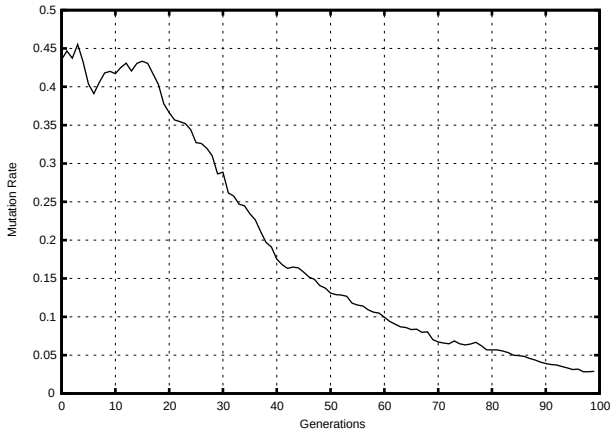


Figure 5. Mutation Rate Evolution

Fig. 5 shows the evolution of the population's mutation rate. This experiment begins with an average mutation rate of 0.45 – one which is large enough to have disastrous consequences on any offspring subjected to it. This is reflected by the results, which show that the population rejects this value and forces it to plummet down to below 0.05. At this point, the drop slows down considerably indicating that the population has found a near optimum level of mutation. Following the generation 90, the mutation rate levels to around 0.025. This value lies in the range 0.02 – 0.04, one of the ranges predicted from the previous experiment set.

5.4 Evolution of Training Cycles

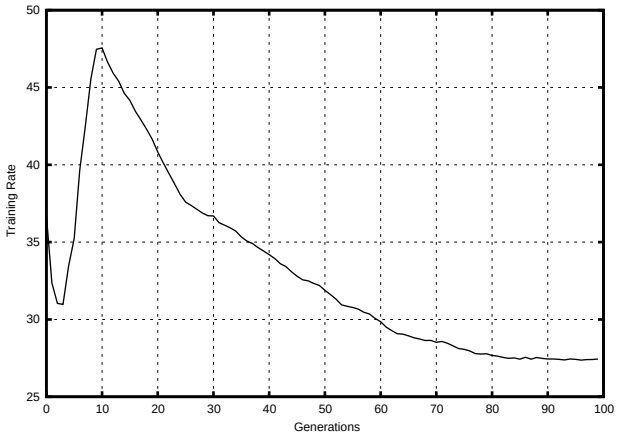


Figure 6. Training Rate Evolution

The evolution of the training rate was treated slightly differently from the others to favour networks able to acheive high fitness without the need for excessive training. Since the training process is very time consuming, it would be useful to adapt the training rate to give the maximum fitness improvement for the least effort. To encourage the evolution of such a training rate, the fitness function was altered to reward networks with lower training rate values.

The results illustrated in fig. 6 show an initial sharp increase in training rate, followed by a gradual fall to only 27 training cycles. It should be noted that the population's fitness during this gradual fall remained above 0.95 at all times. The population spends the initial 10 generations undergoing intensive training and once the networks have acheived a sufficiently high fitness, the population's reliance on training appears to subside.

5.5 Comparison of Performance of Evolved Rates to that of Designed Rates

A final experiment was performed to observe the performance of the evolved rates versus the designed rates. The performance of two populations are illustrated in fig. 7. The first population uses the crossover, mutation and training rates arrived at from iterative experimentation in section 4, that is crossover at 0.625, mutation at 0.02 and training at 100 cycles.

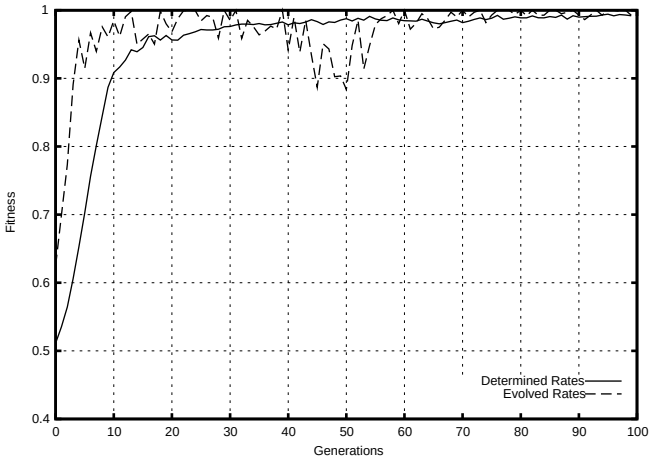


Figure 7. Comparison of Performance of Evolved Rates to that of Designed Rates

The second population uses the rates evolved for crossover (0.59) and mutation (0.025) but is allowed to evolve and adapt the training rate as in section 5.4. This is because the evolution of training rate does not appear to result in an optimum value, rather it is the process of adaptation that results in improved performance. Both populations were allowed to evolve for 100 generations and the experiment was repeated 20 times.

The results show that the evolved rates slightly outperform the designed rates. However, the designed rates population is using a training rate of 100 cycles for the entire experiment, thus inhibiting its performance in terms of time, while the evolved rates population is adapting its training rate as each generation passes, allowing it to reduce the training rate considerably and thus greatly out–performing the determined rates population in terms of execution time.

6 CONCLUSION

The results of the evolutionary rate determination experiments show that it is possible to determine the optimal setting for crossover and mutation for a given problem set. This effectively eliminates the requirement for trial and error estimation of these rates. The values achieved are very close to the values predicted by the first set of experiments, where the rates were modified in order to find the optimum by trial and error.

In addition, the results show that allowing a population to determine its own training rate dynamically can produce a better performance both in terms of fitness and execution time. Future work will examine the relationship between each rate and attempt to evolve all rates simulatenously in both static and changing environments.

ACKNOWLEDGEMENTS

This research is funded by the Irish Research Council for Science, Engineering and Technology.

REFERENCES

[1] P. J. Angeline, G. M. Saunders, and J. P. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks*, 5(1):54–65, January 1994.

[2] R. K. Belew, J. McInerney, and N. N. Schraudolph. Evolving networks: Using the genetic algorithm with connectionist learning. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 511–547. Addison-Wesley, Redwood City, CA, 1992.

[3] B. MacLennan. Synthetic ethology: An approach to the study of communication. In *Artificial Life II: The Second Workshop on the Synthesis and Simulation of Living Systems, Santa Fe Institute Studies in the Sciences of Complexity*, pages 631–635, 1992.

[4] D. Moriarty and R. Miikkulainen. Discovering complex othello strategies through evolutionary neural networks. *Connection Science*, 7(3–4):195–209, 1995.

[5] W. M. Spears. Adapting crossover in evolutionary algorithms. In J. R. McDonnell, R. G. Reynolds, and D. B. Fogel, editors, *Proc. of the Fourth Annual Conference on Evolutionary Programming*, pages 367–384, Cambridge, MA, 1995. MIT Press.

[6] D. Curran and C. O'Riordan. Learning in artificial life societies. In *Report number nuig-it-220202. Technical Report, Dept. of IT.* NUI, Galway, 2002.

[7] D. Curran and C. O'Riordan. On the design of an artificial life simulator. In R.J.Howlett V.Palade and L.C.Jain, editors, *Proceedings of the Seventh International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES 2003)*, University of Oxford, United Kingdom, 2003.

[8] D. Curran and C. O'Riordan. Artificial life simulation using marker based encoding. In *Proceedings of the 2003 International Conference on Artificial Intelligence (IC-AI'03)*, volume II, pages 665–668, Las Vegas, Nevada, USA, 2003.

[9] H. Kitano. Designing neural networks using genetic algorithm with graph generation system. In *Complex Systems, 4, 461-476*, 1990.

[10] P. M. Todd G. F. Miller and S. U. Hedge. Designing neural networks using genetic algorithms. In *Proceedings of the Third International Conference on Genetic Algorithms and Their Applications*, pages 379–384, 1989.

Parsing ill-formed text using an error grammar

Jennifer Foster[1]

Abstract. This paper presents a robust parsing approach which is designed to address the issue of syntactic errors in text. The approach is based on the concept of an error grammar which is a grammar of ungrammatical sentences. An error grammar is derived from a conventional grammar on the basis of an analysis of a corpus of observed ill-formed sentences. A robust parsing algorithm is presented which is applied after a conventional bottom-up parsing algorithm has failed. This algorithm combines a rule from the error grammar with rules from the normal grammar to arrive at a parse for an ungrammatical sentence. This algorithm is applied to 50 test sentences, with encouraging results.

1 Introduction

A traditional rule-based parser, when faced with a sentence which is not described by its grammar, will fail to return any information. The aim of a robust parser is to behave sensibly when confronted with input which does not conform to its ideas about a particular language. One of the ways in which a parser's expectations can be confounded is if the input contains a syntactic error. The ideal way for a robust parser to behave when confronted with an ungrammatical sentence is to recognize that the sentence is ungrammatical, to suggest possible error diagnoses, and for each diagnosis, to produce appropriate parses.

This paper proposes a robust text parsing approach which is capable of handling a large class of syntactic errors. An overview of the idea is given in section 2 and comparisons are made to previous attempts to tackle this issue. A description of the data which forms the basis of this approach is given in section 3, section 4 describes the robust parsing algorithm and section 5 details a preliminary evaluation of the approach and provides suggestions for improvement.

2 Overview

A collection of authentic ungrammatical sentences was analysed. On the basis of this analysis, a set of transformations was applied to a context-free phrase structure grammar yielding an *error grammar* or a grammar of likely ungrammatical sequences. This error grammar contains a sequence of *error rules*, where each error rule corresponds to a particular error which might be expected to occur. When a bottom-up chart parser fails to find a parse for a sentence, a recovery parsing process is un dertaken. This recovery phase uses the edges found during the normal parsing phase together with edges arising from an error rule in its attempt to arrive at a complete parse. In order to keep the search space within a reasonable limit, the interaction between error rules is kept to a minimum. The benefits of this approach to robust parsing are as follows:

[1] Computational Linguistics Research Group, Department of Computer Science,Trinity College, Dublin. Email: jfoster@tcd.ie

1. **A uniform framework for handling different classes of errors**: A popular approach to robust parsing, the *constraint relaxation* approach, proceeds by repeatedly relaxing constraints in the grammar until a parse for an ungrammatical sentence can be found. [4, 5, 12, 3]. However, such an approach does not address the problem of errors arising from the omission or insertion of a word within a sentence – two frequent error types according to the analysis carried out as part of this research (see section 3). Ad-hoc parsing techniques (see for example [9]) are the usual proposed solution to the problem of missing or extra word errors. The approach outlined in this paper does not need to rely on such ad-hoc methods since all errors are dealt with within the uniform framework of an error grammar.

2. **A clear model of ungrammaticality**: Probabilistic parsers, such as those described in [2], are by their very nature robust since they make no distinction between the grammatical and the ungrammatical. According to [10], this is a good thing since a clear line cannot always be drawn between the two. However, the fact that language errors do undoubtedly occur, means that the concept of ungrammaticality cannot be dismissed. The use of an error grammar has the advantage of providing a linguistic model of ungrammaticality – this means that ill-formed sentences can be diagnosed as such, instead of being viewed merely as sentences occurring with a low frequency. The use of an error grammar is, however, compatible with a probabilistic view of language processing since each individual error rule can be augmented with a probability (derived from a corpus of ill-formed sentences such as a larger version of the one described in section 3), thus allowing one robust parse to be preferred over another.

3. **Limiting the number of robust parses**: Constraint relaxation attempts to solve the problem of ill-formed input quickly become intractable unless strictly controlled. This is because any constraint which fails can potentially be relaxed, leading to nonsensical parses for an ungrammatical sentence. Take, for example, the ungrammatical sentence

   `Want to saving money?`

   A constraint relaxation approach has the potential to suggest the following as a parse for this sentence:

   [s [pro `want`] [vp [verb `to`] [np [det `saving`] [noun `money`]]]]

   One solution to this problem has been proposed by [5] - all the potential parses are generated and then ranked on the basis of a general notion of "information loss". It is not clear how well this works in practice. [3] limit the number of parses by stating in advance what constraints may be relaxed. The approach outlined in this paper is closer in spirit to the latter approach. Error rules are

derived on the basis of empirical linguistic data, and are introduced into the parse in a controlled fashion so that nonsensical parses are never proposed as a solution.

The concept of an error rule is not new. [13], for example, describe an ATN parser which contains *meta-rules* corresponding to patterns of ill-formedness. Each meta-rule corresponds to a conventional rule and it is invoked during a parse when a conventional rule fails. This approach is very similar in spirit to mine but there are differences. The meta-rules are integrated into the main parsing process, whereas in my approach the error-rules are only applied after a normal parse has failed. The integration of the meta-rules into the main parsing phase is a consequence of the fact that the parser is a top-down ATN rather than a chart parser. This means that there is less control over when the meta-rules are invoked, since they can be applied even before sections of the input have been encountered. Another consequence of the ATN-based approach is that the meta-rules are more procedural than declarative — it is difficult to view them in isolation from the actual parsing algorithm and so they cannot be used as a model of ungrammaticality in the same way that the error rules described in this paper can.

Mal-rules or *error productions* are used within the field of applied linguistics to describe the errors typically made by the learners of a language. [11] describe a parsing system (the ICICLE system) which identifies syntactic errors in the writing of native speakers of American Sign Language who are learning English as a second language. This system uses mal-rules to model the errors which are expected to be made by this community. Schneider and McCoy's approach differs from mine in two fundamental ways:

1. Their attention is restricted to second language errors and they explicitly attempt to model the second language learning process. The class of errors handled by the approach outlined in this paper is more general - this class includes any type of syntactic error, be it language learning errors or performance errors.[2]
2. The parsing process employed by [11] is not a two-stage one. This means that when a sentence is being parsed, the set of mal-rules are available along with the set of normal rules from the outset, with the unfortunate consequence that grammatical sentences can be parsed with mal-rules and hence flagged as ungrammatical. Since there are no restrictions on when the mal-rules can be applied and on how many can be applied at any one time, the problem of spiralling spurious ambiguity is also an issue here.

3 Error Data

This section describes how a small corpus of syntactic errors was compiled and then used to generate an error grammar from a conventional grammar.

3.1 Error Corpus Collection

A complementary project to this robust parsing study is the compilation of a corpus of ungrammatical language. The corpus currently contains 6,650 words. Every time a sentence containing a syntactic error is noted, the following steps are carried out.

1. The sentence is added to the corpus.
2. A note is made of where the sentence occurred.
3. The error in the sentence is diagnosed and based on this diagnosis, the sentence is corrected.
4. The corrected sentence is added to a parallel corpus of well-formed sentences.
5. A note is made of what was done to correct the sentence. For example, to correct the sentence

   ```
   Are people really capable to understand-
   ing them?
   ```

   the infinitival marker `to` is replaced by the preposition `of`.

Not only does the error corpus provide us with invaluable information on what kind of errors people tend to make when writing or typing, it also provides us with a set of test data which can be used to test any robust parser. Storing the corrected version of the sentence along with the ill-formed sentence means that the results produced by a robust parser can be easily evaluated. If one of the solutions proposed by the parser matches the corrected sentence, then the parser has successfully parsed the ill-formed sentence. This evaluation procedure is carried out for the robust parsing strategy described here (see Section 5), and it is envisaged that the same procedure could be used to compare the ability of several parsers to understand ungrammatical sentences, irrespective of their approach to parsing.

An important aspect of this corpus compilation is that the context in which the error occurs is always available. This means that the sentences can be corrected without ambiguity. This will not be the case for some ill-formed sentences taken out of context, e.g. in the sentence

```
Where did these woman learn such arro-
gance?
```

taken from the British National Corpus[3], there is no way to tell whether the sentence should be corrected by changing the determiner *these* or by changing the noun *woman*.

The errors in the corpus come from the following sources: academic papers and theses, newspapers, magazines, novels, textbooks, websites, lecture notes, student assignments, pamphlets, emails and technical manuals. This is quite a broad source of material, especially when compared to other attempts to create a repository of errors: [6] only collect errors in date expressions in student assignments and newspapers; [1] use an online discussion forum as their only error source.

3.2 Analysing the Errors

Before the errors were analyzed, 50 sentences were randomly extracted from the corpus for the purposes of testing (see section 5). The remaining 306 sentences were analyzed.

Every time a sentence is added to the corpus, a note is made of the operation that is needed to correct the sentence. It is this information which is analyzed. Fig. 1 indicates the correction operations which were applied to the sentences in the corpus. The frequency of each correction operation is provided along with two examples from the error corpus and their corrections from the parallel corrected corpus.

Another possible correction operation is the *move* operation which moves a word from the sentence into another position within the sentence. However, since such an operation occurs relatively infrequently (2%, according to this corpus), the decision was taken to define it in terms of the add and delete operations, and to treat errors

[2] [7] distinguishes between errors which occur as a result of lack of knowledge of the language, i.e. language learning errors, and those which occur as an oversight. The former are known as *errors* and the latter *mistakes*. In this paper the word "error" is used with its more general meaning.

[3] Accessed via *http://sara.natcorp.ox.ac.uk/lookup.html*, April 2000

Correction Operation	Examples
Replace a word (47%)	• the theory **in** empirical→the theory **is** empirical • staff **was** allowed to return→staff **were** allowed to return
Add a word (27%)	• Will be declaring their undying love for each other?→Will **they** be declaring their undying love for each other? • we must assume the validity this induction principle→we must assume the validity **of** this induction principle
Delete a word (18%)	• Why **is** do they appear→Why do they appear • Such databases can be to some extent **be** improved→Such databases can be to some extent improved
More than one of above (8%)	• This means **to allow** structure-sharing→This means structure-sharing **is allowed** • What does a single line **yellow** mean?→What does a single **yellow** line mean?

Figure 1. Corpus analysis results

which could be corrected in this way as *composite errors* or errors which can be corrected by applying more than one correction operation. It is also possible to define the replacement operation in terms of the add and delete operations, but since replacement errors occur frequently, this operation is seen as a valid correction operation in its own right.

Another interesting point to note is that 92% of the errors that occur can be corrected by applying just *one* correction operation. It is these errors which will be handled by the robust parsing approach described here. Composite errors are not handled, although they are not incompatible with this approach.

3.3 Generating the Error Grammar

A grammar of well-formed language is needed in order to generate a grammar of ill-formed. The only constraint on the format of the grammar is that it must be parsable using a chart parser. To test this error grammar approach, a context-free phrase structure grammar containing 1,106 rules was used. The non-terminal symbols of the grammar are augmented with agreement features where appropriate. For each error type, the process of generating error rules for this type is described. The error rules were generated manually but there is no reason why this generation procedure could not be automated.

3.3.1 Replacement errors

In a sentence containing a replacement error, the erroneous word must be replaced by a word that is similar in some way to it. The ways in which two words can be similar correspond to particular types of replacement errors found in the error corpus. These are as follows:

1. A word can be replaced by a word similar to it in spelling. This is determined as follows: a word X is similar in spelling to a word Y if X can be transformed into Y by changing or deleting one letter in X, or by adding a letter to X, e.g. `nor` with `not`
2. If the word is a noun, verb or determiner, it can be replaced by the same word with a different value for an agreement feature, e.g. first person `am` with third person `is`, singular `man` with plural `men`
3. If the word is a verb, it can be replaced by the same verb with a different form, e.g. infinitival `tell` with present participle `telling`
4. If the word is a preposition, it can be replaced by any other preposition, e.g. `for` with `of`
5. A word can be replaced by a word with the same lexical root but with a different part of speech category, e.g. adjective `syntactic` with adverb `syntactically`

So, for each rule in the grammar which expands a pre-terminal symbol (i.e. a part-of-speech category), an error rule is generated for all the words which are similar to the right-hand side of this rule, according to the similarity criteria just described. For the rule

```
verb(sing,third) --> [is]
```

the following error rules are generated

```
verb(sing,third) spellop [in]
verb(sing,third) spellop [it]
verb(sing,third) spellop [if]
verb(sing,third) agreeop [are]
verb(sing,third) agreeop [am]
```

```
verb(sing,third) vformop [be]
verb(sing,third) vformop [being]
verb(sing,third) vformop [been]
```

To distinguish error rules from conventional grammar rules, the $--> $ connective is replaced by a connective which describes the error, e.g. the connective `spellop` refers to replacement errors which result when the correct and incorrect word are similar in spelling, the connective `agreeop` refers to replacement errors where the correct and incorrect word have conflicting values for an agreement feature, and the connective `vformop` refers to replacement errors where the two words (or verbs) have conflicting verb form values. In all other respects these connectives mean the same thing as the conventional rule connective $--> $.

3.3.2 Missing word errors

For each rule in the grammar (excluding rules expanding pre-terminal symbols), an error rule is generated which has the same right-hand side as the original rule except that a pre-terminal symbol on the right-hand side is removed.[4] So, for example, for the rule

```
np(Num,Per) --> det(Num,Per),nbar(Num,Per)
```

the following two error rules will be generated:

```
np(Num,Per) missingop nbar(Num,Per)
np(Num,Per) missingop det(Num,Per)
```

For unary rules such as

```
np(Num,Per) --> pro(Num,Per)
```

no error rules with an empty right-hand side are generated. Instead, for each rule which has the category `np(Num,Per)` on its right-hand side, a corresponding error rule is generated which omits this category. An error rule isn't generated if its right-hand side is the same as the right-hand side of a conventional rule, e.g. for the rule

```
nbar(Num,Per) --> adj, n(Num,Per)
```

the error rule

```
nbar(Num,Per) missingop n(Num,Per)
```

isn't generated since this already exists in the grammar as a conventional rule.

3.3.3 Extra word errors

For each rule in the grammar (excluding rules expanding pre-terminal symbols), an error rule is generated which has the same right-hand side as the original rule except that a symbol is added at some position in the rule. This symbol must be capable of matching with any pre-terminal symbol in the grammar: in a typed system it could be the most general word type; in the system described here, a Prolog variable is used, resulting in the added bonus that these error rules can cope with extra constituents as well as extra words. Given, for example, the rule

```
vp_pastp --> v_pastp, pp.
```

the following two error rules are generated:

```
vp_pastp extraop v_pastp, X, pp.
vp_pastp extraop v_pastp,pp,X.
```

[4] It might be interesting to investigate to what extent missing word error rules provide a treatment for the grammatical phenomenon of ellipsis.

4 Robust Recovery Algorithm

For all *appropriate* error rules

1. Add an active edge corresponding to that rule to the chart agenda
2. Reinvoke the bottom up chart parser - stop when all solutions have been found or when there are no more edges on the agenda
3. If there are solutions record them
4. Remove all edges in the chart which have arisen from the error rule

Figure 2. Recovery Algorithm

The robust recovery parsing algorithm is given in Fig 2. The algorithm takes as input a sentence which has failed to parse using a bottom-up chart parser. It is important that a bottom-up parsing strategy as opposed to a top-down one is used during the normal parsing phase. A bottom-up parser is driven by the words in the input sentence and will be guaranteed to find all partial parses in the ungrammatical sentence. A top-down parser, on the other hand, is driven by the grammar rules and will run out of steam, leaving sections of the input sentence untouched. All the chart edges built during the normal parsing phase, active and inactive, are available to the recovery algorithm. After an error rule has been tried (whether successfully or unsuccessfully), step 4 wipes the slate clean for the next error rule. It ensures that there is no interaction between error rules.

Fig. 3 details the algorithm which is used to select appropriate error rules. The $errorop$ connective corresponds to any connective which appears in an error rule, e.g. $missingop$ or $spellop$ (see section 3.3).

Step 1 is used to choose error rules corresponding to replacement errors. For this kind of error, appropriate error rules are selected on the basis of the actual words in the input sentence, in a process similar to the chart initialization process.[5]

Step 2 is used to choose error rules corresponding to missing and extra word errors. The chart edge notation

$$W --> \alpha.[start, end]$$

is explained as follows: a category W consisting of the category sequence α has been found between the positions $start$ and end in the input sentence. The selection process is a combination of the bottom-up rule of chart parsing[6](step 2a) and a left and right scan of the chart (step 2(a)i).

5 Evaluation

The robust recovery algorithm described in Fig. 2 and Fig. 3 was applied to the 50 test sentences which were randomly extracted from the error corpus. The shortest sentence in the set of test sentences has 4 words, the longest has 35 words and the average sentence length is 20 words.

[5] When a chart is initialized, each word in the input sentence is examined and an active edge is added to the chart before this word for each grammar rule which has this word as its right hand side.

[6] The bottom-up rule of chart parsing states that if there is an inactive edge of a particular syntactic category, X, spanning a particular section of the chart, then any rule which has X as the first category on its right-hand side is added as an active edge at the beginning of this chart section.

1. For all words, w, in the input sentence
   (a) Add all error rules of the form

   $Z\ errorop\ w$

   to the chart in the form of an active edge before w
2. For all inactive edges

   $W - - > \alpha.[start, end]$

   found during the normal parsing phase
   (a) For all error rules of the form

   $Z\ errorop\ W\ \beta$

   i. If there is an active edge in the chart ending at position $start$ looking for Z or if position $start$ is at the start of the sentence, and if there is a sequence of inactive edges in the chart starting at position end for the category sequence β, then the rule is appropriate.
   ii. Otherwise the error rule is discarded.

Figure 3. Algorithm to choose appropriate error rules

5.1 Accuracy

The recovery process was deemed to have worked if one of the corrections it proposed for an input sentence matched the corrected version of the sentence. According to this measure, the robust recovery procedure achieved an accuracy rate of 84%. The 8 failed attempts can be explained as follows:

1. **More than one error in the sentence**: 2 sentences in the test data contained two separate errors, e.g. the sentence

   ```
   From all of the above considerations,
   the following roadmap for the has been
   derived derived for this presentation
   ```

   which contains a missing word error along with an extra word error. Step 4 in the recovery algorithm means that only one error rule is ever considered during any one parse attempt, with the result that this algorithm will not handle sentences containing two or more errors. The next step in this research is to modify the algorithm so that it can consider more than one error rule under certain controlled circumstances.

2. **Composite errors**: 3 sentences in the test data contained a composite error, e.g. the sentence

   ```
   But not one of them is capable to deal
   with robustness as a whole
   ```

   which can be corrected by replacing `to` with `of` and by replacing `deal` with `dealing`. Sentences containing a composite error will not be dealt with under this approach for the same reason that sentences containing more than one error won't: only one error rule can be considered at any one time. To deal with composite errors, the recovery algorithm will need to be modified so that it has the potential to use more than one error rule at a time.

3. **Failure to recognize a sentence as ungrammatical**: 3 sentences in the corpus were not recognized as ill-formed by the normal parser, e.g. the sentence

   ```
   Compared to syntactic valid structures,
   the set of syntactically incorrect
   sentences can be considered almost
   infinite
   ```

   which can be corrected by replacing the adjective `syntactic` with the adverb `syntactically`. If the recovery algorithm was applied to these sentences the appropriate robust parse would be suggested, but since they are not ungrammatical according to the test grammar, the recovery process will never be applied. Short of extending the test grammar so that its grammatical constraints make use of sophisticated semantic and contextual information (in itself a formidable task), these kinds of cases are unavoidable.

5.2 Efficiency

	Average Parse No.	Average Cycle No.
Corrected	2.1	820.4
Ill-formed	4.1	5225.2

Figure 4. Some performance results

Fig.4 compares the ill-formed sentences which could be parsed correctly to their corrected counterparts, in terms of average number of parses and average number of parse cycles. The notion of a parse cycle was proposed by [9] as an implementation-independent measure of a chart parser's efficiency. In one parse cycle, an edge is taken from the agenda, added to the chart and used (via the fundamental and bottom-up rules of chart parsing) to propose more edges.

The increase in number of parses for the set of ill-formed sentences is expected, since the robust recovery algorithm is essentially proposing ways of correcting an ill-formed sentence, and each correction will have a certain number of parses associated with it. Consider, for example, the ill-formed sentence

```
Will be declaring their undying love for
each other
```

The corrected version of the above sentence is the sentence

```
Will they be declaring their undying love
for each other
```

and this receives 4 parses. The robust parsing algorithm finds these 4 parses, along with another 4 associated with the other possible correction, i.e.

```
They will be declaring their undying love
for each other.
```

The parse cycle number for an ill-formed sentence includes the parse cycle number for the original parse of the sentence (when no parses are found) plus the parse cycle number for the recovery parse. The large figure given in Fig.4 is also not a mystery given the large number of error rules. The reduction in the parse cycle number for the ill-formed sentences is a goal for further research. The following are possible ways in which improvements could be made:

1. **Just one correction** The number of parse cycles will be significantly reduced if the recovery phase parsing stops after one error rule has parsed successfully. The problem with this is that, for

some sentences, there is more than one way to correct the sentence and the first correction obtained does not always correspond to the correct one. A probabilistic ordering of the error rules (along the lines proposed by [8]) suggests itself as a possible way around this problem. However, in order for realistic probabilities to be computed, more errors will need to be collected.

2. **Choosing error rules** Another way to reduce the number of parse cycles is to reduce the number of inactive edges and/or words in the sentence which are used to choose the appropriate error rules (see Fig. 3). This could be done by guessing where in the input sentence the error is most likely to have occurred. [9] suggests that running a top-down parse on an ungrammatical sentence, directly after running a bottom-up parse, can be used to pinpoint the location of an error. This technique needs to be investigated within the context of the approach described here.

6 Conclusions

The robust parsing technique described in this paper is based on the concept of an error grammar, which provides a set of error rules describing ungrammatical sentences in the same way that a conventional grammar provides rules which describe grammatical sentences. The rules in an error grammar are distinguished by means of a rule connective from rules in the normal grammar but they can be parsed using normal chart parsing techniques. The error rules themselves are realistic since they are derived from normal grammar rules on the basis of actual error data. The recovery algorithm described in this paper is designed to use one error rule together with the results produced by a bottom-up chart parser in order to find a parse for an ungrammatical sentence. The decision to use just one error rule at a time has also been justified by empirical data. The recovery parsing algorithm has achieved promising results in its first evaluation, results which, it is expected, will be improved when the suggestions for further work have been carried out.

ACKNOWLEDGEMENTS

I would like to thank Carl Vogel and the anonymous referees of this paper for their helpful comments and suggestions.

REFERENCES

[1] Markus Becker, Andrew Bredenkamp, Berthold Crysmann, and Judith Klein, 'Annotation of error types for german news corpus', in *Proceedings of the ATALA workshop on Treebanks*, Paris, (1999).
[2] Charniak, 'A maximum-entropy-inspired parser', in *Proceedings of the NAACL-2000*, (2000).
[3] Shona Douglas and R. Dale, 'Towards robust patr', in *COLING '92*, pp. 468–474, (1992).
[4] Frederik Fouvry, 'Robust unification for linguistics', in *ROMAND 2000 1st workshop on Robust Methods in Analysis of Natural language Data*, Lausanne, (October 2000).
[5] Frederik Fouvry, 'Constraint relaxation with weighted feature structures', in *Proceedings of the 8th International Workshop on Parsing Technologies*, Nancy, France, (23-25 April 2003).
[6] Koldo Gojenola and Maite Oronoz, 'Corpus-based syntactic error detection using syntactic patterns', in *NAACL-ANLP00, Student Research Workshop*, Seattle, (April 2000).
[7] Carl James, *Errors in Language Learning and Use: Exploring Error Analysis*, Addison Wesley Longman, 1998.
[8] David M. Magerman and Carl Weir, 'Efficiency, robustness and accuracy in picky chart parsing', in *Proceedings of the 30th ACL*, (1992).
[9] Chris S. Mellish, 'Some chart-based techniques for parsing ill-formed input', in *Proceedings of the 27th ACL*, pp. 102–109, (1989).
[10] Geoffrey Sampson, 'Evidence against the grammatical/ungrammatical distinction', in *Empirical Linguistics*, chapter 10, Continuum, New York, (2001).
[11] David Schneider and Kathleen McCoy, 'Recognizing syntactic errors in the writing of second language learners', in *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and the Seventeenth International Conference on Computational Linguistics (COLING-ACL)*, volume 2, Montreal, Canada, (August 1998).
[12] Carl Vogel and Robin Cooper, 'Robust chart parsing with mildly inconsistent feature structures', in *Nonclassical Feature Systems*, eds., Andreas Schöter and Carl Vogel, Centre for Cognitive Science, University of Edinburgh, Working Papers in Cognitive Science, (January 1995). Volume 10.
[13] Ralph M. Weischedel and Norman K. Sondheimer, 'Meta-rules as a basis for processing ill-formed input', *American Journal of Computational Linguistics*, **9**(3–4), 161–177, (1983).

An Empirical Investigation of the Association between Musical and Linguistic Aptitude

Lorraine Frances Gilleece[1]

Abstract. The purpose of this paper is to present details of an empirical investigation carried out by the author into the association between musical and linguistic aptitude. Results show a moderate correlation between results on musical aptitude and linguistic aptitude tests. These findings add further support to the existence of a relationship between musical ability and linguistic ability. They also contribute to the debate on how different intelligences may interact and may influence each other.

1 INTRODUCTION

In 1983, Gardner [11] first proposed his theory of multiple intelligences, according to which there are seven distinct areas which constitute intelligences, i.e. musical intelligence, bodily-kinesthetic, logical-mathematical, linguistic, spatial, interpersonal, and intrapersonal. It is worth noting that in 1993 [12], he stated that there is no theoretical reason why two or more intelligences could not overlap or correlate with one another more highly than with others.

Recent research points towards the likelihood of a relationship between musical activity and many other cognitive processes, e.g. spatial reasoning (see [15] and [16]). Musical activity has also been linked to increased creativity (see [21]) and to linguistic intelligence (see [8]). Some studies have shown a certain overlap in the processing of music and language in the brain, e.g. Broca's area is shown to be involved in the processing of musical syntax [13]. This provides evidence of a strong relationship between the processing of language and music and might at least partly account for influences of musical training on verbal abilities, e.g. as shown in [6] and [8].

Areas in which there exists evidence of common processing in music and language include grouping[2], metre and contour[3], (see [14] and [10]). Grouping and metre are both closely related to rhythm - a term which is often considered synonymous with metre. However, when a distinction is made between metre and rhythm, the former refers to the division of notes into evenly distributed note values and is indicated at the beginning of a composition by a metre signature, e.g. 3/4, 6/8, 4/4. Therefore in 3/4 time there are 3 crotchet (quarter note) beats in one bar, while in 6/8 time there are 6 quaver (eighth note) beats in one bar. Rhythm, on the other hand refers to stressing the notes according to the musical sense of the phrase, (for further discussion see [7]). The difference between rhythm and metre can be seen clearly in the traditional Irish reel and hornpipe. Both may be written in 4/4 time, i.e. the metre is 4/4, yet they have distinctive rhythms. The rhythm is defined in terms of the underlying structure of the metre, e.g. it would be very difficult to have a waltz rhythm written in 4/4 time. Rhythm is defined in [14] as "the temporal and accentual patterning of sound" and tempo, grouping and metre are considered to be subsidiary concepts of rhythm.

It has been pointed out ([14]) that studying music and language in parallel offers a chance to understand human auditory communication and cognition in a broader perspective than is possible by studying either domain alone. It may be the case that "Music" and "Language" are not entirely independent mental faculties, but rather that these are labels for complex sets of processes some of which are shared and some of which are not.

This paper presents preliminary results of an investigation into the association between musical aptitude and foreign language aptitude. I will first outline the tests which were carried out. I will then present a preliminary analysis of the results. Discussion of the results will follow.

2 METHOD

A series of tests was devised to analyse the musical and linguistic aptitude of Irish secondary school students. Subjects ranged in age from twelve years and one month to sixteen years and six months. Subjects were drawn from a reasonably wide geographical area, i.e. urban and rural areas in Ireland, and from different socio-economic backgrounds.

2.1 Bentley measures of musical aptitude

The musical aptitude test used was the Bentley Measures of Musical Abilities, [1]. This is a standardised music aptitude test designed in the 1960s to measure basic music skills in children, i.e. those skills which children have inherited or acquired incidentally and not through specific musical training. The statistical reliability of the test given in [2] is 0.84. The test consists of four components, i.e. pitch, tonal memory, chords and rhythmic memory.

The pitch test tests the ability to distinguish between different pitch sounds. It involves listening to twenty pairs of pitch sounds. Answers are either 'S' for same, if the second sound of the pair is the same as the first; 'U' for up, if the second sound moves up, i.e. higher in pitch than the first; or 'D' for down, if the second sound moves down. Some of the intervals used are those normally used in Occidental music, i.e. semitones, whole-tones, thirds, fifths, etc. Others are micro-intervals, i.e. pitch differences smaller than a semitone.

The test of tonal memory is based on the assumption that appreciation of a melody is impossible without the ability to recall, in detail,

[1] Centre for Language and Communication Studies, University of Dublin, Trinity College, Dublin 2, Ireland.

[2] Grouping refers to the tendency for individual elements to be grouped into larger units, e.g. phrases in music. Grouping also occurs in language as it has been shown that boundaries between spoken words are marked by pre-boundary lengthening.

[3] Melodic pattern.

sounds that have already been heard. Ten items of paired comparisons are heard, each half of each item being a five-note tune. In the second half of each item, one note is changed by either a whole tone or a semitone. Subjects must say which note is changed. All notes are equal in length and there are no dynamic accents.

The test of chords consists of ten two-note chords, eight three-note chords and two four-note chords. The subject must say how many notes are heard at once. It is acknowledged in [2] that the ability to analyse chords is not fundamental to melody but it is argued that it is a highly desirable ability.

The rhythmic memory test consists of ten items of paired comparisons, each half of each item being a four-pulse rhythmic figure. In the second half of each item, one pulse may be changed. The subject answers 'S' for same if no such change occurs. If there is a change, the subject must note the pulse of the note which is changed. Rhythmic memory is measured separately from tonal memory as various researchers have shown that there is only weak a correlation between the ability to appreciate relations of pitch and relations of rhythm. It is fair to say that this test is a test of rhythm rather than metre as only 4/4 time is used, i.e. subjects are not asked to distinguish between different metres.

2.2 Language tests

The language test also consists of four parts, two of which are based on a version of the Modern Language Aptitude Test (MLAT) [3], modified in Trinity College Centre for Language and Communication Studies. The MLAT was developed and validated in the late 1950s. It is based on the four factors identified in [4] as those making up language aptitude. These are:

- Phonetic Coding. This is the ability to "code" auditory phonetic material in such a way that this material can be recognized, identified, and remembered over something longer than a few seconds.
- Grammatical Sensitivity. This is the ability to handle "grammar", i.e. the forms of language and their arrangements in natural utterances.
- Rote Memory for Foreign Language Materials. This is the capacity to learn a large number of associations in a relatively short time. This ability is independent of phonetic coding ability as even those who have the phonetic coding ability may still not be able to hear and remember new relationships.
- Inductive Language Learning Ability. This is the ability to infer linguistic forms, rules and patterns from new linguistic content itself with a minimum of supervision or guidance.

In 1990, Carroll [5] proposed another category of abilities which he believes is important for language learning, namely auditory abilities. Auditory ability refers not only to issues such as hearing loss but also to the ability to understand speech which is unclear or under conditions of masking. Auditory abilities identified in [19] include Speech Perception Under Distraction, Temporal Tracking (an ability to recognize and remember the order in which particular sounds occur), Maintaining and Judging Rhythm and Discrimination among Sound Patterns.

Carroll also decided that some factor of memory ability should be included. He mentions free recall learning, memory for episodic events, verbal discrimination learning, memory span and tests of delayed memory as areas worthy of further research in relation to language aptitude. The MLAT has not been updated to reflect these new ideas on the components of language aptitude.

The full MLAT was not available for this study. The language test used here consists of four tests, i.e. Numerical, Sounds, Chinese and Grammar. The Numerical test and Grammar test are similar to two parts of the MLAT, i.e. Number Learning and Grammatical Sensitivity. Number Learning is a measure of the auditory and memory abilities associated with sound-meaning relationships. The Grammatical Sensitivity test demonstrates the subject's awareness of the syntactical patterning of sentences in his native language and of the grammatical functions of individual elements in a sentence, according to [5].

In the Number Learning test which I administered, subjects are asked to learn a set of numbers in Czech through aural input and then to discriminate different combinations of these numbers. A native speaker of Czech gives number pairs, i.e. English followed by Czech for the digits 2 to 6. The numbers 20, 30, 40, 50, 60 are then given along with their English counterparts. Finally the subject hears numbers 200, 300, 400, 500, 600. The test involves the foreign numbers being jumbled up. The subject must write the digits of the number he hears on the tape.

The Sounds test involves the subject deciding which of two words is repeated, given aural input. Sixteen items are recorded by a native speaker, each of which consists of three words. The third word is a repetition of either the first or second word. The subject ticks Box A if the first word is repeated. Otherwise, if the second word is repeated, Box B is ticked. This tests the subject's ability to discriminate between sounds which are not phonemic contrasts in English.

The Grammatical Sensitivity test aims to examine grammatical sensitivity in the native language, i.e. English. Subjects are given a key word in one sentence. They are then asked to read a second sentence and select the word in that sentence that has the same function as the key word in the first. There are twenty such items, e.g.

Given sentences:

- **Mary** likes to go to school.
- **(A) He** likes to go **(B) fishing** in **(C) Maine**.

the subject must decide which of the words in bold in sentence 2 fulfills the same grammatical function as the word in bold in sentence 1. In this case 'He' has the same grammatical function as 'Mary' in sentence 1 as both act as the subject of the sentence.

The Chinese test was included in the battery as a result of the author's hypothesis that the capacity to discriminate words which differed only in pitch might be a useful ability in language learning. A native speaker of Mandarin Chinese recorded four words differing only in tone along with their English equivalents. Subjects listened to these four pairs twice, before hearing ten Chinese sentences, each containing one of the learned words. Subjects were asked to write the English equivalent of the Chinese word in the sentence.

2.3 Raven standard progressive matrices test

The Standard Progressive Matrices test was constructed to measure the eductive component of **g** as defined in Spearman's theory of cognitive ability. Eductive ability is defined in [17] as the ability to forge new insights, the ability to discern meaning in confusion, the ability to perceive, and the ability to identify new relationships. The essential feature of eductive ability is the ability to generate new, largely non-verbal, concepts which make it possible to think clearly. The Raven test was included in the test battery in order to investigate the possibility of both music and language being related to non-verbal intelligence.

The Standard Progressive Matrices Test consists of five sets of 60 problems, divided into sections A, B, C, D, E, each of which contains 12 problems. The puzzles exhibit serial change in two dimensions simultaneously. Each puzzle has a part missing, which the subject has to find among the options provided.

2.4 Testing conditions

Testing took place in seven different class groups from five different schools. Three classes were from one school. Class groups were as follows - four first year classes, one second year class, one third year class and one transition year class. In the majority of classes (5/7), testing took place on two different days, with music and language tests on one day and the Raven Progressive Matrices test on another. In the other two classes, students took all three tests on one day, with a short break between the second and third test. In total 149 tests were fully completed and available for analysis.

2.5 Scoring

Music tests were scored according to the key for marking provided in [1]. In the Language tests, subjects were awarded one point for a correct answer, zero for an incorrect answer. In the Chinese test I noted where an answer was the correct digit, but an incorrect multiple, e.g. 2 instead of 20. These observations have not as yet been included in the analysis. The Raven Standard Progressive Matrices test was scored in accordance with smoothed British Norms for the Group Test of Children, given in [17].

3 RESULTS AND DISCUSSION

The data consists of variables *Pitch, Tunes, Chords, Rhythm, Sounds, Chinese, Numerical, Grammar* and *Raven*. At this stage only basic statistical analysis has taken place.

A preliminary examination of the full data set revealed a significant correlation between *Total Music* and *Total Language* ($r=0.37$, $p<0.01$). *Rhythm* and *Language* showed a slightly higher correlation, ($r=0.47$, $p<0.01$). Higher correlations were found between individual components of a test, e.g. *Grammar* and the whole test (i.e. *Language*). However this is to be expected if the component makes a significant contribution to the test battery as a whole. For the purposes of this paper, the more interesting correlations are those between components of the music test and components of the language test as these are not to be expected in the same way as the correlations between the language test components themselves or the music test components themselves. Indeed it could be argued that there is no a priori reason for a strong correlation between music and language, particularly if these relate to independent mental faculties or modules, see [9].

With respect to nonverbal intelligence, there is a significant correlation between *Total Language* and *Raven* ($r=0.39$, $p<0.01$), while *Total Music* also shows a similar correlation with *Raven* ($r=0.34$, $p<0.01$).

3.1 Closer Inspection

Table 1 provides a summary of the results found in each school. The correlations between *Total Music* and *Total Language* are given, as well as those between *Rhythm* and *Total Language*.

It can be seen from this table that School C shows a remarkable linear relationship between *Total Music* and *Total Language* ($r=0.85$, $p<0.01$). This is clearly evident in the scatter-plot in Figure 1.

Table 1. Results for Individual Schools

School	A1 N=26	A2 N=23	A3 N=26
Total Music and Total Language	r=0.47 p<0.05	r=0.22	r=0.40 p<0.05
Rhythm and Total Language	r=0.61 p<0.01	r=0.53 p<0.01	r=0.25

School	B N=27	C N=14	D N=15	E N=18
Total Music and Total Language	r=0.33	r=0.85 p<0.01	r=0.49	r=0.41
Rhythm and Total Language	r=0.20	r=0.63 p<0.05	r=0.28	r=0.17

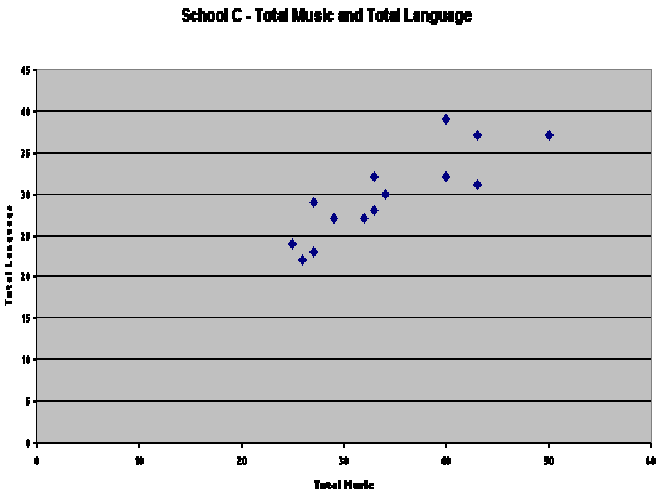


Figure 1. Scatterplot of Total Music and Total Language Scores in School C

However this relationship was not found to the same extent in the data-set as a whole. This is seen in Figure 2. It is noteworthy that many of the outliers come from the same school (School B). There is no apparent explanation for this phenomenon at present, but it receives further discussion in Section 3.6.

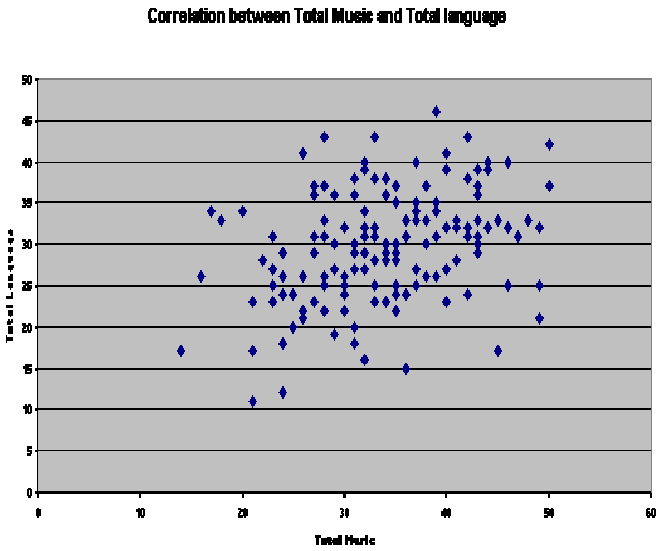


Figure 2. Scatterplot of Total Music and Total Language Scores

Table 1 also reveals that the correlations between *Rhythm* and *Total Language* are generally higher than those between *Total Music* and *Total Language* and also that they have a greater level of significance. Groups A1, A2 and C show significant correlations for *Rhythm* and *Total Language*, while A1, A3 and C show significant correlations for *Total Music* and *Total Language*. Schools A1 and C show significant correlations in both *Total Language* and *Total Music* and in *Rhythm* and *Total Language*. It is interesting to note that School B shows almost no correlation between Total Music and Total Language (r=0.03). This contrasts dramatically with School C mentioned above where r=0.85, $p<0.01$.

The total possible score in the *Rhythm* test was 10, while the maximum possible score for *Total Language* was 61. The mean score for *Rhythm* was 6.7 and the mean score for *Total Language* was 29.7. It is worth noting that the *Rhythm* has a slightly negatively skewed distribution. This means that students found the *Rhythm* test somewhat easier than other tests. This is in line with the findings in [2] where it was found that musically unselected children of 7 to 14 years found the rhythmic memory test easier than the tonal memory test. It is taken there as an indication that memory for rhythmic patterns is likely to develop earlier than any other aspect of musical ability.

Out of the total 149, there is only one case of *Rhythm* score ≥ 8 and *Total Language*<20, (*Rhythm*=9, *Total Language*=17) and there are eight cases of *Rhythm*≥ 8 and *Total Language*≤ 25. Of the latter, in four cases *Rhythm*=8 and *Total Language*=25. From this it can be seen that when *Rhythm*≥ 8, the *Total Language* score is generally near to 30, i.e. above average *Rhythm* usually results in average or above in *Total Language*.

It is interesting to note that these findings are in line with those found previously in a similar study [20], where a strong correlation was found between a rhythm test and a word test which involved subjects repeating foreign language words. In that study, rhythm was found to have a greater correlation with word than the melody test had with word. Rhythm and metre were not discussed separately although the tests focussed mainly on rhythm rather than on metre. Similarly here, a correlation of 0.28 was found between *Tunes* and *Total Language* while a correlation of 0.47 ($p<0.01$) was found between *Rhythm* and *Total Language*.

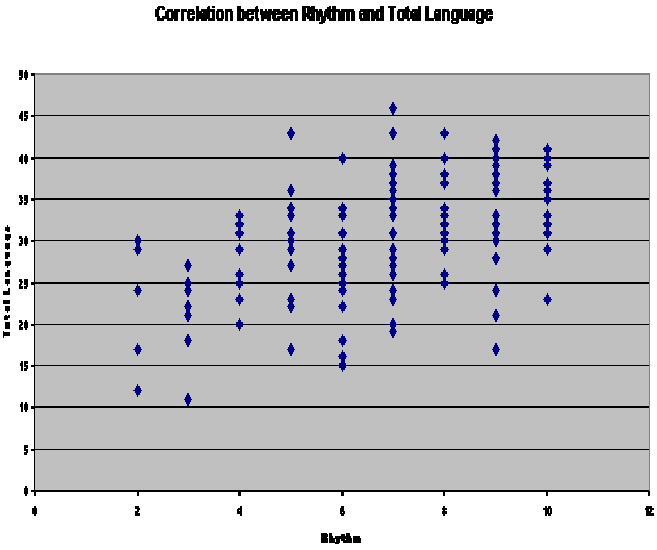


Figure 3. Scatterplot of Rhythm and Total Language Scores

Also as mentioned above (in Section 1), rhythm (as a general term for tempo, grouping, metre) is one of the areas in which it is believed there may be common processing mechanisms for language and music, see [14] and [10]. This would seem to be supported by the results of the current study showing a moderate correlation between *Rhythm* and *Total Language* scores.

3.2 Above average rhythm, below average language

I will now look at cases of students who got above average in *Rhythm* and below average in *Total Language*. As stated above the average *Rhythm* mark is 6.7 and the average score for *Total Language* is 29.7.

Two subjects got full marks in *Rhythm* and below average in *Total Language*. Of these, one got 29 in *Total Language* which is only very slightly below the average. The other got 23 in *Total Language*. Of the others who got above average in *Rhythm* and below average in *Total Language*, 19 got between 25 and 29 in *Total Language*, while 7 got between 19 and 24 and 1 person got 17.

To summarise, 86 subjects got above average in *Rhythm*. Of these, only 27 got below average in *Total Language* (31%). In other words, 69% of subjects with above average in *Rhythm* get average or above in *Total Language*.

3.3 Above average language, below average rhythm

An analysis of cases with above average *Total Language* and below average *Rhythm* reveals the following results. There are 19 such cases, 16 of which have *Total Language* scores between 29.7 and 35, i.e. between average and 35. There are only 3 cases of *Total Language*>34 and *Rhythm* below average. The scores of these three students were *Total Language*=40, *Rhythm*=6; *Total Language=36*, *Rhythm*=5; *Total Language*=43, *Rhythm*=5.

To summarise, it can be said that 78 students got greater than average *Total Language*. Of these, 59 also got greater than average *Rhythm*, i.e. 76% of subjects who got greater than average *Total Language* also got greater than average *Rhythm*.

3.4 Chinese and pitch

Examination of the correlation between the *Chinese* test and *Total Language* revealed that *Chinese* did not contribute all that much to the battery of language tests. The correlation is significant however, but is only 0.32, $p<0.01$ between *Total Language* and *Chinese*. Concerning the relationship between pitch in music and language, the correlation between *Chinese* and *Pitch* is 0.09, i.e. almost zero and not significant.

This was an unexpected result as the hypothesis had been that pitch in language and music might be processed in a similar way and therefore results in the *Chinese* test might be strongly correlated to those in *Pitch*. This hypothesis was based on the claim that there is a left-hemisphere bias for the processing of categorical pitch information in both language and music, see [14]. However, it is also noted that differences between musical and linguistic pitch (notably, the presence of scales in the former) may result in these two forms of pitch being processed separately within the left hemisphere. If this is the case, perhaps an individual will not show similar ability in musical pitch and linguistic pitch.

3.5 Correlations with non-verbal reasoning

Turning now to the correlation between *Non-Verbal Reasoning* and *Total Music* and that between *Non-Verbal Reasoning* and *Total Language*, *Raven* and *Total Language* show a positive correlation ($r=0.4, p<0.01$), as do *Total Music* and *Raven* ($r=0.34, p<0.01$).

The correlation between *Total Music* and *Total Language* ($r=0.37, p<0.01$) is almost equal to that between *Total Language* and *Raven* ($r=0.39, p<0.01$). This shows that although the correlation between non-verbal intelligence and the other factors is significant, the correlation between music and language seems to be almost as important as that between non-verbal intelligence and language.

3.6 Consideration of outliers from School B

As noted in Table 1, School B shows a much weaker correlation between *Total Music* and *Total Language* than the other schools. It is interesting to consider the difference in average marks when School B is included/excluded.

Average *Total Music* score is 32.97 with School B excluded, whereas including School B shows the average to be 33.81. Average *Total Language* score is 29.71 with School B included, while it is 29.85 with School B excluded. This shows that School B increases the average music score, while simultaneously decreasing the average language score. Average *Rhythm* with School B included is 6.74, while without School B it is 6.58. The students from this school seem to show superior musical ability but inferior language ability. Further questioning about musical and instrumental training, did not reveal anything particularly unusual, so at present I have no explanation to account for this superior musical aptitude.

4 CONCLUSION

These preliminary results are undoubtedly interesting and warrant closer inspection and further statistical analysis. At this stage, the relationship between rhythmic ability in music and foreign language aptitude seems to look most promising and certainly deserves attention. This study revealed a moderate correlation of between rhythm and language ($r=0.47$, $p<0.01$) based on a sample of 149 subjects. These results could prove highly significant for the development of foreign language aptitude tests, an area in which there has not been extensive progress since the 1970s. The discovery of a relationship between music and language would also be important for theories of cognition and human intelligence. Immediate and practical implications of a correlation between music and language are in the fields of education and speech and language therapy. Already, as shown in [18], the extra-musical benefits of musical instruction have been exploited in teaching children to read.

REFERENCES

[1] Arnold Bentley. Measures of musical abilities. NFER-Nelson Publishing Company, Berkshire., 1966.

[2] Arnold Bentley, *Musical Ability in Children*, Harrap & Co. Ltd., London, 1966.

[3] J. Carroll and S. Sapon. Modern language aptitude test. The Psychological Corporation, New York, 1955.

[4] John B. Carroll, 'The prediction of success in intensive foreign language training', in *Training Research & Education*, ed., R. Glaser, chapter IV, 87 – 136, University of Pittsburgh Press, Pittsburgh, (1962).

[5] John B. Carroll, 'Cognitive abilities in foreign language aptitude: Then and now', in *Language Aptitude Reconsidered*, eds., Thomas S. Parry and Charles W. Stansfield, Language in Education: Theory and Practice, 11–29, Prentice Hall Regents, New Jersey, (1990).

[6] Agnes S. Chan, Yim-Chi Ho, and Mei-Chun Cheung, 'Music training improves verbal memory', *Nature*, **396**, 128, (November 1998).

[7] Aaron Copland, *What to Listen For in Music*, McGraw-Hill Book Company, New York, 1957.

[8] Sheila Douglas and Peter Willatts, 'The relationship between musical ability and literacy skills', *Journal of Research in Reading*, **17**(2), 99 – 107, (1994).

[9] J.A. Fodor, *The Modularity of Mind*, M.I.T. Press, Cambridge, MA, 1983.

[10] W. Fries and A. Swihart, 'Disturbance of rhythm sense following right hemisphere damage', *Neuropsychologia*, **28**(12), 1317 – 1323, (1990).

[11] Howard Gardner, *Frames of mind: The theory of multiple intelligences*, Basic Books, New York, 1983.

[12] Howard Gardner, *Multiple Intelligences: The Theory in Practice*, Basic Books, New York, 1993.

[13] Burkhard Maess, Stefan Koelsch, Thomas C. Gunter, and Angela D. Friederici, 'Musical syntax is processed in broca's area: an meg study', *Nature Neuroscience*, **4**(5), 540 – 545, (May 2001).

[14] Aniruddh D. Patel and Isabelle Peretz, 'Is music autonomous from language?', in *Perception and Cognition of Music*, eds., Irène Deliège and John Sloboda, chapter 10, 191 – 209, Psychology Press, East Sussex, UK, (1997).

[15] Frances H. Rauscher, Gordon L. Shaw, and Katherine N. Ky, 'Music and spatial task performance', *Nature*, **365**, 611, (1993).

[16] Frances H. Rauscher, Gordon L. Shaw, and Katherine N. Ky, 'Listening to mozart enhances spatial-temporal reasoning: towards a neurophysiological basis', *Neuroscience Letters*, **185**, 44 – 47, (1995).

[17] J.C. Raven, J.H. Court, and J. Raven, *Raven Manual Section 3*, Oxford Psychologists Press, Oxford, 1996 edn., 1996.

[18] Maria Spychiger, 'Responses to katie overy's paper, "can music really improve the mind?"', *Psychology of Music*, **26**, 197 – 210, (1998).

[19] L. Stankov and J. L. Horn, 'Human abilities revealed through auditory tests', *Journal of Educational Psychology*, **72**, 21–44, (1980).

[20] Rosalind Stevenson, *Is there an association Between Musical and Linguistic Aptitude? An empirical study*, Master's thesis, University of Dublin, Trinity College, Dublin, 1999.

[21] Norman M. Weinberger, 'Creating creativity with music', *MuSICA Research Notes*, **V**(2), (Spring 1998). http://www.musica.uci.edu/mrn/V5I2S98.html (last verified 24/03/2003).

A Matrix Approach to Connectivity Analysis

Mark Hargaden[1] and **Paddy Doyle** and **Barry Smyth** and **Mark Keane**
{**firstname.lastname**}**@ucd.ie**

Abstract.

The World Wide Web has grown beyond most people's ability to effectively handle without some help. Search engines provide a partial solution and are often a user's main interface to the WWW. Advances in the field of search engines have used connectivity analysis (CA) of the Web in their indices. We analyze two of the most common CA algorithms and show a common framework for their calculation. We believe that this framework calculates efficient and accurate values for each algorithm.

1 Introduction

Since 1994 the Web has grown to well over 2.1 billion documents [7] with current estimates of over 429 million users [9], and it continues to grow and attract users. It is often very hard to find information that a user requires in this large interconnected maze of pages. Search engines provide a partial solution to this problem, by returning pages related to a user defined query string. They are often the main source of information for users of the Internet.

Recent research in the field has had success through the incorporation of new ideas into Web search engines. In particular, link information on the Web is complimenting more traditional keyword-based approaches in search engines such as Google. This has shown that the connectivity structure of the Web is important and can be utilized to improve search engines and other search tools. In this paper we will show a common framework for calculating the two most common connectivity algorithms and discuss the results obtained.

2 Related Work - Background

Much of the current work in the area has its basis in the work done by Kleinberg [8]. In his research, the link structure of the Web is used to find authoritative pages on a broad query topic. Kleinberg's idea is that by creating a link to a page, a person is conferring some kind of authority and relevance, on the page. The main technique of his research was to take a small set of pages, rich in pages related to a broad query, and find the authorities for the query topic using the link structure. This process was called HITS [2]. This technique relies solely on the connectivity structure of the topic area and doesn't include content analysis.

As well as defining an authoritative page as one that is "linked to" by many pages Kleinberg also described hub pages as pages that pointed to many good authorities. This created a mutually reinforcing relationship. A good hub points to many good authorities, and a good authority is pointed to by many good hubs. Hubs are often indexes or links pages of a site, since a human generated list would very likely point to good authorities. All pages are given initial hubs and authorities scores and an iterative algorithm is used to update the hub and authority scores. A page's (p) authority score ($x^{(p)}$) is the sum of the hubs scores ($y^{(p)}$) of the pages that point to it (Equation 1). A page's hubs score ($y^{(p)}$) is the sum of the authority scores ($x^{(p)}$) of the pages to which it points (Equation 2).

$$x^{(p)} \leftarrow \sum_{q:(q,p)\in E} y^{(p)} \tag{1}$$

$$y^{(p)} \leftarrow \sum_{q:(p,q)\in E} x^{(p)} \tag{2}$$

The process of updating the scores is repeated for a set number of iterations or until the scores converge. The PageRank [10] system is also based on a connectivity analysis of the Web and has been successfully incorporated into the Google search engine [3]. PageRank computes a rank for every Webpage based on its location in the graph of the Web. They define inlinks as the links into a page and outlinks as the links out of a page. The PageRank ($R(u)$) of a page (u) is the sum of the PageRank ($R(v)$) of the inlinks (B_u) of the page (v) divided by the number of links (N_v) out of each inlink page.

$$R(u) = c \sum_{v\in B_u} \frac{R(v)}{N_v} \tag{3}$$

A high PageRank can be given in two main situations, firstly if the page has a few highly ranked inlinks or, secondly if it has many inlinks. The PageRank for a page is divided equally along its outlinks, hence the more links a page has the more diluted the rank is to its "children". One problem with this approach is the phenomenon of "rank sinks" or cycles of pages that accumulate rank. Hence PageRank implements a random surfer model where a random jump is modeled to prevent rank sinks having a detrimental effect on the scores.

The main difference between PageRank and HITS is that PageRank calculates rank on the whole graph of the Web and not a subset. Also PageRank has no idea of a hub, it only confers authority between pages in the graph. The PageRank technique has been combined with many conventional search engine techniques such as content analysis to make the Google [3] search engine.

3 Collecting Connectivity Data

The analysis of connectivity data requires either live connectivity information or a store of downloaded connectivity data. For our analysis we created a Web Spider/Crawler called Lorg to collect connectivity data and store in a database. Lorg is split into three components. The download module downloads a HTML page and extracts

[1] University College Dublin, Belfield, Dublin 4, Ireland.
[2] Hyperlink-Induced Topic Search

its links. The queue module manages the queues of sites that the spider will visit. The connectivity database stores the connectivity information extracted from the Web.

3.1 Download/HTML Parse Module

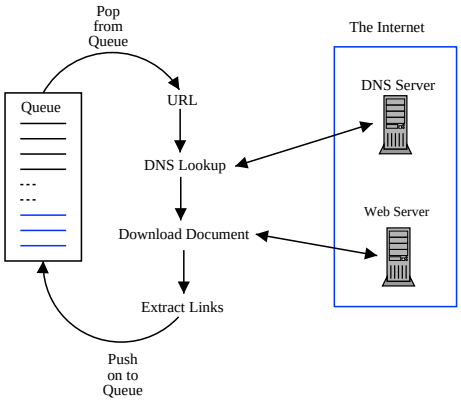


Figure 1. Download module cycle.

The download cycle is illustrated in Figure 1. The first step is to convert the hostname to an IP address. A DNS cache is kept to increase the speed of this stage. Next a socket connection is established, if this fails the site is recorded as a failed connection. If the connection is successful the appropriate HTTP commands are submitted to request the HTML source of the page. Once the HTML source stream is available it is passed to the parsing module, otherwise a HTTP error code is returned (e.g. 404 not found).

The Link Extractor will iterate over the stream until a link tag is encountered. It will return the link tag as a token and on the next call searches for the next instance of a link in the page. Each time the next method is called the next link in the page is returned. This process is repeated until all links have been extracted from the page. All parsed links are syntax checked for validity and converted to absolute URLs.

3.2 Queueing Module

The Queue stores the URL's of pages that have been encountered as links in pages but have yet to be accessed by the HTML download module. The Queue Module is based on a multi-queue system where all pages from a certain host are on the same queue (as in [6]). By cycling through the queues each host is only accessed every Nth time where N is the number of queues. This gap between accesses to the webservers, prevents overloading the webservers involved. This partitioning of the queue is accomplished by way of a hash function on the hostname of the URL (see Figure 2).

3.3 Connectivity Database

The connectivity database stores the information collected by the Lorg spider which forms the raw material for the analysis phase of the project. The basic unit of connectivity data is a page and an associated inlink or outlink. The outlinks for a page are easily discovered by downloading the page and parsing its links. To be certain to find all inlinks for a page every page on the Web would have to

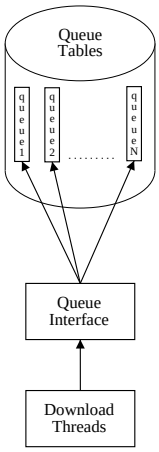


Figure 2. Partitioned Queue Tables

be examined and hence find all references to the page found. Storing and accessing connectivity information requires a large amount of disk space and therefore speed and efficiency are crucial (e.g. approx 3GB for 80 million URLs). Some systems have used large customized text files [5, 10] but our connectivity store incorporates a relational database in order to take advantage of the key efficiencies and scalabilities of this format.

OUTLINKS	
Page	Outlink
URL1	URL2
URL1	URL3
URL1	URL5

Figure 3. OutLinks Table

Figure 3 displays the structure of the outlinks table containing the URL to outlink mappings. For each link an entry in this table is created. This is a space intensive storage method but allows for fast searches to find the outlinks for a page.

To store each instance of a URL string takes on average 56 bytes, and storing the full URL's in the database for each link would be a waste of space. Hence a system of Unique Identifiers (UID) is used to reference a particular page. The UID is a 4 byte number from between 0 to a max of 4.2 billion. A master table called the UIDTABLE maps the URL string to a UID number. Everywhere else in the system the UID is used rather than the URL string, thus reducing space requirements. In Figure 4 we can see the different in storage requirements between storing the links as full URL string (Naive) and storing the UID reference. Further information on the workings of Lorg are available at [2] and [4].

4 Analyzing Connectivity Data

In our implementation of the PageRank and HITS algorithms we want to focus on their inherent similarities and build a system that

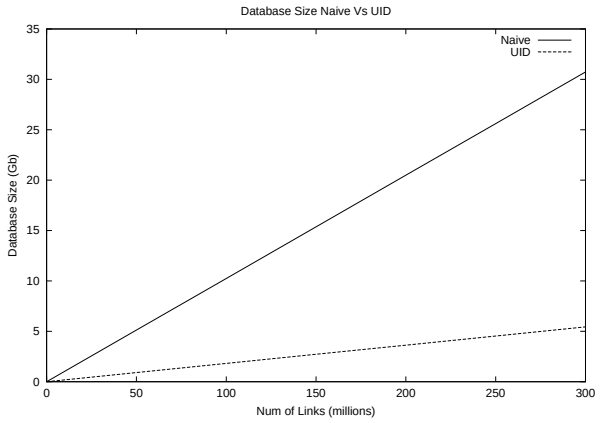


Figure 4. Naive versus UID-based URL storage

could calculate both with very little modification. Since both algorithms have a basis in the finding of principal eigenvectors of certain matrices we have used this to construct the system.

4.1 The Adjacency Matrix

In mathematical terms the Web can be viewed as a graph, with pages as nodes and links as directed edges. As such, it can be modelled by an adjacency matrix A. Whose entries a_{ij} are equal to 1 if there is a link from page j to page i. In the PageRank calculations $a_{ij} = \frac{1}{N_i}$ where N_i is the number of outlinks of page i (see Figure 5). This is because a rank from a page is divided along its outlinks. Another way to view this value is that this is the weighting of the link that passes to the child page. We also define the transpose of a matrix $A = a_{ij}$ as $A^T = a_{ji}$ where the rows are columns and the columns are rows.

	1	2	3	4	5	6
1	0	0	1/2	0	1/2	0
2	1/3	0	0	0	0	0
3	0	1	0	0	0	0
4	1/3	0	1/2	0	1/2	0
5	0	0	0	0	0	0
6	1/3	0	0	0	0	0

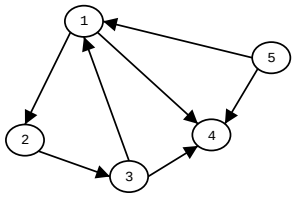


Figure 5. Simple Graph and Associated Adjacency Matrix

4.2 Eigenvectors and Eigenvalues

One of the properties of a square [3] matrix is that there is a value λ and a vector q such that:

$$Aq = \lambda q \tag{4}$$

This means that a square matrix (A) multiplied by some vector (q) is equal to some value (λ) times the same vector (q). This value λ is called the eigenvalue of the matrix A and q is the associated right eigenvector for A. There can be more than one eigenvalue and eigenvector pair and, for a matrix of order N, there are N eigenvalue and eigenvector pairs (they may not all be distinct). The eigenvalues of a matrix A and A^T are the same and for every λ associated with A there is an eigenvector q and for each A^T there is an eigenvector p. The vector p is the left eigenvector for the transpose of the matrix.

$$A^T p = \lambda p \tag{5}$$

This property of matrices and eigenvectors can be combined with the idea of Markov chains in order to prove the convergence of eigenvectors.

4.3 Markov Chains

Markov chains can be used for the analysis of the responses from a system which behaves spasmodically. If one assumes that there is a system which has n possible states and a transition can take place through these states. Each of the transitions may change the probabilities of the system. For each state j the probability p of changing state to i is p_{ij}.

$$\sum_{i=0}^{N} p_{ij} = 1 \quad 0 \leq p \leq 1 \tag{6}$$

Since no probability can be negative the second property holds through. The probabilities of a transition in the system can be represented as a matrix $P = p_{ij}$ known as a transition matrix. Each column of the matrix must sum to unity, since if you are in a state the sum of the probabilities of going to all other possible states is 1 (Equation 6). For this transition matrix it is possible to take a vector of the initial state of a system X^0. If this vector X^0 is multiplied by the transition matrix P the resulting vector X^1 is the probabilities of moving to new states from their initial states. Hence the element i of the vector X^1 is the probability that the system is in state i after one transition.

$$X^{K+1} = Px^k = p^{k+1}X^0 \tag{7}$$

The sequence X^k of probability vectors is called a "Markov chain" and converges to a long run probability vector (q) such that:

$$Pq = q \tag{8}$$

There is therefore a right eigenvector of the transition matrix P for which the corresponding eigenvalue is unity. Two important properties of this equation are that firstly all transition matrices have a long run probability vector. Secondly the long run probability vector is independent of the initial starting vector.

$$Av_i = v_{i+1} \tag{9}$$

These properties lead us to an interesting conclusion in that if we have a transition matrix A we can create a unit vector v which can be multiplied repeatedly by the matrix A such that the vector converges to the long run probability vector.

The matrix A, the transition matrix, is the matrix of links and the long run probability vector is our rank vector. Hence using the above properties we have shown that taking a transition matrix and multiplying it by a unit vector we get a set of converged values. This property can be used to form the basis of a unified connectivity analysis specifically the PageRanks and HITS calculations([1]).

[3] A matrix with the same number of rows and columns

4.4 Calculation of PageRank

```
Initialize Source Vector
while (z<num-iterations) {
    for  $i$=0...Num-Rows {
        Links=Get-RowLinks(i)
        Source=Get-RowSource(i)
        for  q=1...size(row)
            $Rank_i=Rank_i+(Source(q)*Links(q))
        $Rank_i=d+(Rank_i*(1-d))
    }
    z++
}
```

Figure 6. Pseudo code for computing PageRank

Initially the Source vector is assigned its initial values by inserting an entry for each page in our calculation in the Source database table. The structure of this entry is (Page, Value, Level) where Page is the Uid of the current page, Value is the initial P_r (initially one), and Level is the iteration at which the value P_r was recorded (the initial level is zero). At each iteration (i) the ith row of the links table is selected and the database function will select all columns in the links table where row=i. The number of entries that are returned is the number of inlinks for the particular page i in our system. We then extract the column indexes of the row and select the appropriate entries from the Source vector. For each item in the row the Source value is multiplied by the links table entry and added to the total rank for the page. The entry in the links table is $\frac{1}{N}$ and this multiplied by the Source entry gives Source/N. When the rank for a page i is calculated the dampening factor is applied. The dampening factor we used was $d = 0.15$. This process is repeated for each row of the table resulting in an iteration of the PageRank algorithm. This process is repeated a number of times or until the values converge (see Figure 6).

4.5 Calculation of HITS

The algorithm to compute HITS is described in Figure 7. It has two distinct parts, the calculation of the Hubs scores and the calculation of the Authorities scores. To calculate the Hubs scores each row i of the Links matrix is extracted, for each row the associated Hubs score is computed. The Authorities score for each page, given by the column indexes, are summed together to give the Hubs score for page i. The calculation of Authorities is the same process using the columns of the Links matrix. For each column of the Links matrix i, the associated Authorities scores are calculated by getting the sum of the Hubs scores for the row indexes of the Links column i. The Hubs and Authorities scores for each iteration are normalized before the next iteration is begun. This process is repeated for a number of iterations, z or until the values converge. The results are normalized to the interval [0-1] by getting the square root of the sum of the vector values and dividing all values in the vector by this number.

5 Experimental Analysis

We tested the algorithms on a subset of a crawl from the Lorg Spider over a period of 3 months. The data set used in the experiment was

```
Initialize Hubs and Authorities Vector
while (z<num-iterations) {
      for  i=0...Num-Rows {
           Links=Get-RowLinks(i)
           Auths=Get-RowAuths(i)
           for  q=1...size(Links col indexes)
                Hubs_i=Hubs_i+Auths(q)
      for  i=0...Num-Cols {
           Links=Get-ColLinks(i)
           Hubs=Get-RowHubs(i)
           for  p=1...size(Links row indexes)
                Auths_i=Auths_i+Hubs(p)
     Normalize(Auths)
     Normalize(Hubs)
      z++
}
```

Figure 7. Pseudo code for computing HITS

based on taking the first 1 million pages from the crawl that have at least one outlink to another page in the set and at least one inlink in the set. This is to ensure that all pages can gain rank from the incoming links and distribute rank on the outgoing links and hence contribute to the overall ranking within the system.

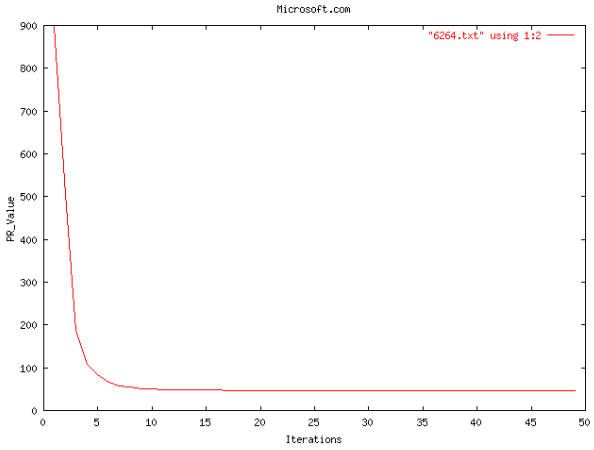


Figure 8. Convergence Data for Microsoft.com

5.1 Convergence of Ranks

The values for both HITS and PageRank on the set should ideally converge to constant values after a number of iterations. This is mathematically guaranteed if the graph that the Links table represents is irreducible and aperiodic. In practice the values converge after a number of iterations, depending on the data set. In our set of 1 million pages most of the pages converge to a reasonable limit after at most 20 iterations. An reasonable limit is the point beyond which there is no significant variation in the values.

The graph of microsoft.com in Figure 8 is an ideal example of convergence, where the values progress consistently to the converged limit. Most of the pages do not converge in this manner. The progression in Figure 9, representing the data for directory.google.com, is more indicative of the top ranked pages. The overall structure of the graph is similar but between the first and fifth iterations the values oscillate. The values still level off and converge between the fifth and tenth iterations. This is a more typical representation of the convergence of the values within set.

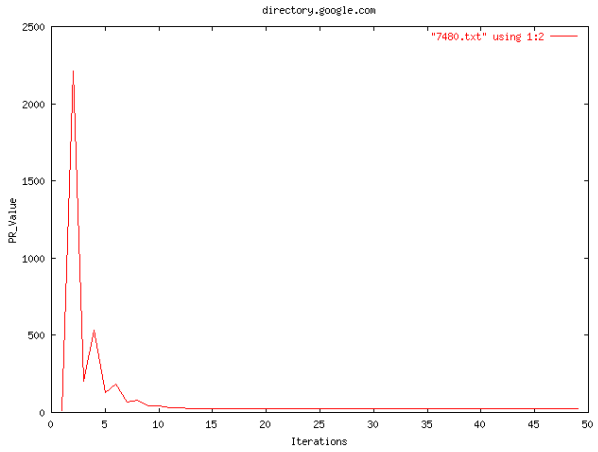


Figure 9. Convergence Data for directory.google.com

Figure 10 contains data for the rates of convergence of the top ranked pages from our PageRank. We can see that largest number of pages converge at iteration 16 and most pages have converged before iteration 27. There is a good distribution of convergence rates between iterations 5 and 25.

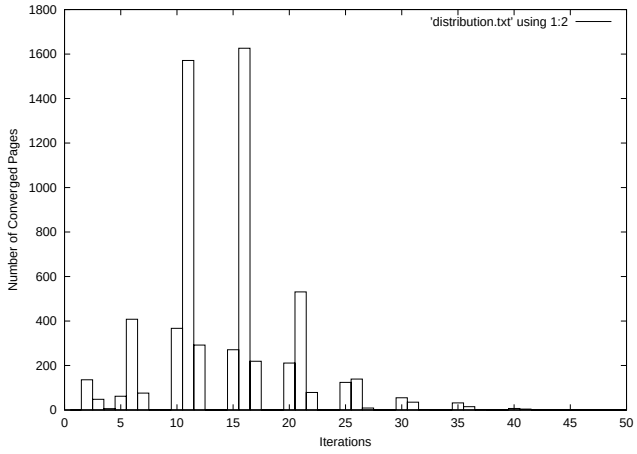


Figure 10. Convergence Rates for PageRank

This shows that the convergence of PageRank is a progression where initially small numbers of pages converge followed by the bulk of the pages and finally trailing off to the last remaining pages.

5.2 PageRank Results

The objective of the PageRank algorithm is to rank the relative importance of a page in a graph of pages. Hence we would hope to find the top ranked pages after running our PageRank would be important pages in the data set, of interest to a Web surfer.

The graph in Figure 11 is of the number of pages that have PageRanks in the ranges given at iterations 5, 25 and 50. The PageRanks in Figure 11 are distributed to each end of the scale indicating that most of the pages have values near zero or greater than one. The largest number of pages are near the zero end of the scale and these are the pages distributing more rank than they are receiving. There is also a small peak at the other end of the scale of pages above one. These are the pages that have converged to positive values.

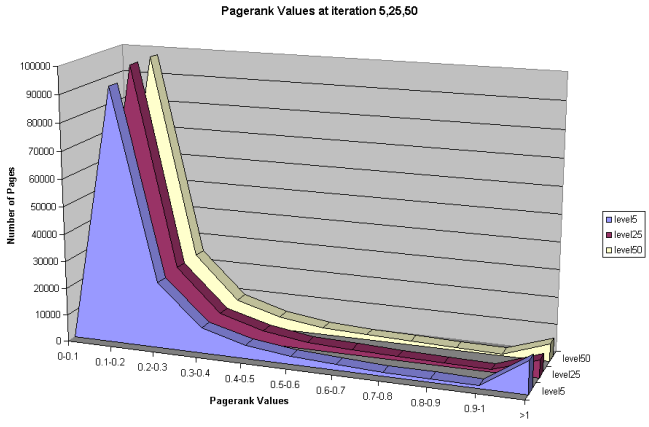


Figure 11. Distribution of Rank at Iterations 5, 25, 50 PageRank

Pages that converge to a non-zero value which makes up (roughly 10 %) of the set and those pages that converge towards zero (roughly 90 %). The pages that converge to non-zero values are the relatively important pages in the set and are gaining more rank per iteration than they are giving away.

5.2.1 Comparison to Google PageRank

In order to judge the effectiveness of our approach to PageRank we compared it to the Google PageRank available from the Google Bar [11]. This PageRank value is a very rough estimate of the PageRank a page has in the Google system. We would hope to find a similarity between well ranked pages in our system and this PageRank value from Google.

We have taken the top and bottom 200 pages based in our PageRank system and compared them to the PageRank from the Google bar. The results are shown in Table 1

The average PageRank from Google of the datasets top 200 pages is 7.790 and the average for its bottom 200 pages is 4.746. This would indicate that pages with a high PageRank in our system will have a relatively high Google PageRank. The averages for our top 200 and bottom 200 is 21.429 and 0.15 respectively. Some caution must be taken in comparing these results directly since our PageRank values

	Top 200	Bottom 200
Overall average Google PR	7.790	4.746
Overall average Our PR	21.429	0.15
Top 20 average Google PR	8.636	4.375
No Google PR	15	17
Page not available	35	57
No Backlinks	1	3

Table 1. PR (Our PageRank) Vs Google PR

are not normalized to a [0-10] range like the Google PageRank values. However we feel there is a significant correlation between the two PageRank values. The average Google PageRank for our top 20 pages is 8.636. Some pages have no Google PageRank values because they haven't been indexed by the search engine. The largest number of pages that were unavailable was in the bottom 200 pages at 57 pages. Some pages don't have any incoming links and hence can't receive any PageRank. Three pages in our top 20 had a Google PageRank of 10 these were the websites for Microsoft, NASA and MIT. Overall we can see that the pages from our system with a high PageRank have a high Google PageRank and the pages in our system with a low PageRank have a significantly lower Google PageRank.

5.3 HITS Results

We would hope that pages with a high Authorities score within our system would be generally good pages in the data set, and pages with a high Hubs score would point to a lot of the Authority pages.

Results obtained with our HITS also were very polarised with very few pages getting a significant Authority or Hubs score. Most pages converged to zero which would indicate a very small number of pages gaining all the rank. This could be explained by a small group of pages in some loop or rank sink, a common problem for these types of algorithms.

Authorities	Converging to a non-zero value at iteration 50:	875
Authorities	Converging to zero at iteration 50:	202047
Hubs	Converging to a non-zero value at iteration 50:	22838
Hubs	Converging to zero at iteration 50:	279079

Table 2. Distribution of Values

Figure 12 shows the distribution of rank for the Authorities scores for iterations 5,25 and 50 iterations. At iteration 5 the most number of the pages are at the higher end of the scale. For iteration 25 through to iteration 50 most of the pages are converging to zero. This increase in the number of pages at low values would indicate that most of the pages are converging to zero leaving a small set of pages that are gaining all the rank.

5.3.1 HITS Analysis

Our HITS results contained a set of strongly interlinked pages which skewed the results. These pages would normally be removed if they are on the same sites, but these pages were on different hosts. The pages contained little content, but all had a list of links at the end of the page to other similar pages on different hosts. This set of pages gained high Authority and Hubs scores and only distributed rank to other pages in its set. Finding and removing such sets of pages is not an easy task, though searching for pages with similar links may help to find these groups of pages.

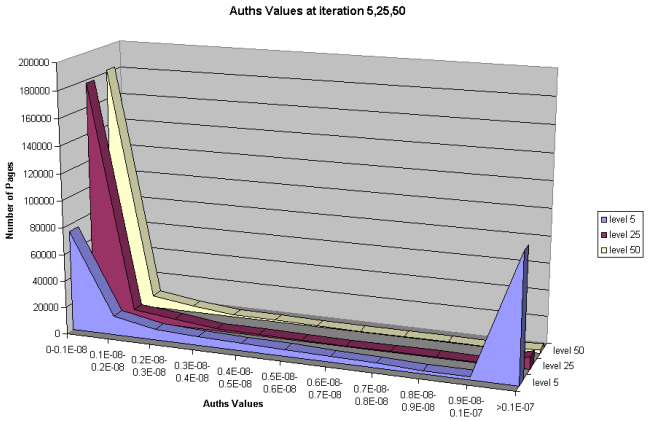


Figure 12. Distribution of Rank at Iterations 5, 25, 50 Authorities

6 Conclusions

In our experimentation we have calculated PageRank and HITS values on a dataset obtained from a run from the Lorg spider. The convergence properties of the PageRank and HITS algorithms have been displayed. In both cases the values for a page in the system have been shown to converge to a constant value after a number of iterations. A favourable comparison has been made between the PageRank values we have obtained and those available from Google thus validating to a degree our calculation of PageRank against the definitive benchmark of the Google PageRank.

ACKNOWLEDGEMENTS

The support of Enterprise Ireland under the Basic Research Grants Scheme is gratefully acknowledged.

REFERENCES

[1] Chris Ding, Xiaofeng He, Parry Husbands, Hongyuan Zha, and Horst Simon, 'Pagerank, hits and a unified framework for link analysis', LBNL 49372, Lawrence Berkeley National Lab, (2001).
[2] Paddy Doyle, *Efficient Modular Crawling of Subgraphs of the Web*, Master's thesis, 2003.
[3] Google. http://www.google.com.
[4] Mark Hargaden, *Connectivity Analysis of Subgraphs of the Web*, Master's thesis, 2003.
[5] T. Haveliwala, 'Efficient computation of pagerank'. Technical report Stanford University, 1999.
[6] A. Heydon and M. Najork, 'Mercator: A scalable, extensible web crawler', *World Wide Web*, **2**, 219–229, (1999).
[7] Julie Kennon Aimee Johnson. http://www.cyveillance.com/.
[8] Jon Kleinberg, 'Authoritative sources in a hyperlinked enviroment', *Journal of the ACM*, 604–632, (1998). Hits.
[9] NetRating. http://www.nielsen-netratings.com.
[10] Larry Page, Sergey Brin, R. Motwani, and T. Winograd, 'The pagerank citation ranking: Bringing order to the web'. Pagerank, 1998.
[11] Google ToolBar. http://toolbar.google.com.

Applying VerbNet for automatic semantic role identification[1]

Svetlana Hensman and **John Dunnion** [2]

Abstract.

The problem of automatic semantic role identification is an important part of many natural language processing systems and while recent syntactic parsers can correctly label over 95% of the constituents of a sentence, finding a representation in terms of semantic roles is still unsatisfactory.

This paper describes a method for identifying and labelling the semantic roles of a sentence. Our system creates a verb pattern using the parse tree of the sentence and then matches it to the possible semantic frames extracted from the VerbNet lexicon for this verb in order to identify the semantic roles.

1 Introduction

The problem of automatic semantic role identification is an important part of many natural language processing systems and while recent syntactic parsers can correctly label over 95% of the constituents of a sentence, finding a representation in terms of semantic roles is still unsatisfactory.

The term *semantic roles* (also called *cases* or *thematic roles*) was first defined as part of *case grammar* theory [5], the purpose of which was to describe the functional relationships between the constituents of a sentence in relation to the verb, and, more specifically, verb sense (verbs that have more than one sense are treated and described separately).

One of the assumptions of case grammar is that each verb tends to always require a specific type of surrounding words, which are the verb *cases*. For example, the verb *buy* is almost always accompanied by at least two constituents, one representing the buyer and another one the thing being bought.

As it can be seen in the following example, there is no direct match between the syntactic and the semantic label of the constituents. For example, in the following sentences ***Bob*** *opened the door with a key*, ***The key*** *opened the door* and ***The door*** *opened*, the syntactic subject in each sentence corresponds to different semantic roles – *Agent*, *Instrument* and *Patient* respectively.

There are number of existing approaches for identifying the semantic roles that differ significantly. The traditional parsing approaches, such as HPSG grammars and Lexical functional grammars, to a certain extent all suggest semantic relationships corresponding to the syntactic ones. They strongly rely on manually-developed grammars and lexicons, which must encode all possible realizations of the semantic roles. Developing such grammars is a time consuming and tedious process and such systems usually work well within limited domains only.

The data-driven approach is an alternative approach, based on filling semantic templates. Applying such a model to information extraction, in AutoSlog Riloff [11] builds a list of patterns for filling in semantic slots in a specific domain. In the domain of the Air Traveler Information System, Miler at al. [9] apply statistical methods to compute the probability of a constituent in order to fill in a semantic slot within a semantic frame.

Gildea and Jurafsky ([6] and [7]) describe a statistical approach for semantic role labelling using data collected from FrameNet. They investigate the influence of the following features for identification of a semantic role: *phrase type* (the syntactic label of the constituent - e.g. noun phrase (NP) and verb phrase (VP)), *grammatical function* (the relationship of the constituent to the rest of the sentence; the simplest version of the function only considers the syntactic label of the parent), *position* (the position of the constituent in relation to the predicate – before or after), *voice* (the voice of the predicate) and *head word* (the main verb in the predicate). The system achieved 82.1% accuracy.

This paper describes a method for identifying and labelling the semantic roles of a sentence. Our system creates a verb pattern using the parse tree of the sentence and then matches it to the possible semantic frames extracted from the VerbNet lexicon for this verb in order to identify the semantic roles.

VerbNet [1] is a computational verb lexicon, based on Levin's verb classes [8], that contains syntactic and semantic information for English verbs. Levin describes 4183 verbs grouped in 191 verb classes, each of which shares similar semantic and syntactic behaviour. VerbNet describes the intersective Levin's classes as defined by Dang at al. [3] using Lexicalized Tree Adjoining Grammar formalism.

Each VerbNet class defines a list of *members*, a list of possible *thematic roles*, and a list of *frames (patterns)* of how these semantic roles can be realized in a sentence. A sample VerbNet class – the class **say-37.7** – can be seen in Figure 1.

2 Corpus and corpus pre-processing

The corpus we currently use to develop and test our system is the freely available Reuters-21578 text categorization test collection [10]. The data was originally collected and labelled by Carnegie Group, Inc. and Reuters, Ltd. in the course of developing the CONSTRUE text categorization system.

[1] This work is developed as part of the INTINN project, funded by Enterprise Ireland.

[2] Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland, email:svetlana.damianova, john.dunnion@ucd.ie

say-37.7

Members
[announce(1 3), articulate(2 4), blab(1), blurt(1), claim(1),confess(2), confide(1), convey(1), declare(1 3), mention(1 2), note(1), observe(2), proclaim(2), propose(?1), recount(1), reiterate(1), relate(3), remark(1 2), repeat(1 4),report(1 2), reveal(2), say(1 2 6 7), state(1), suggest(1 3)]

Thematic Roles
Agent[+animate OR +organization]
Topic[+communication]
Recipient[+animate OR +organization]

Frames
Basic Transitive (Topic Object)
"Ellen said a few words"
Agent V Topic
transfer_info(during(E),Agent,?Recipient,Topic)
cause(Agent,E)

Transitive (+ Recipient PP)
"Ellen said a few words to Helen"
Agent V Topic Prep(to) Recipient
transfer_info(during(E),Agent,Recipient,Topic)
cause(Agent,E)

Sentential Complement
"Ellen said that melons were selling well"
Agent V Topic[+sentential -infinitival]
transfer_info(during(E),Agent,?Recipient,Topic)
cause(Agent,E)

Sentential Complement (+ Recipient PP)
"Ellen said to Helen that melons were selling well"
Agent V Prep(to) Recipient Topic[+sentential -infinitival]
transfer_info(during(E),Agent,Recipient,Topic)
cause(Agent,E)

Figure 1. The VerbNet description for the verb class **say-37.7**

The collection contains 4 SGML-format files (reut2-000.sgm, reut2-001.sgm, reut2-002.sgm, reut2-003.sgm), each of which is approximately 1.25MB in size and contains 1000 news reports.

Before we attempt to identify the semantic roles for the constituents of a sentence, we do some initial pre-processing. More specifically, we identify the sentence boundaries (sentence tokenization) and construct a parse tree for each sentence. To detect the sentence boundaries we apply a sentence tokenization algorithm based on a dictionary of abbreviations collected manually from the corpus, and set of rules. The most common rule is that

IF a word (assume words are separated by spaces) ends with a '.' and is not a known abbreviation, and the next word is capitalized, THEN mark a sentence boundary.

There are also some corpus specific rules, such as

IF the word 'Reuters' appears alone on a line, and if it is indented, and if it is followed by '</BODY>', THEN mark a sentence end.

Our tests show that for Reuters corpus this algorithm identifies over 95% of the sentences boundaries correctly. Figure 2 shows an extract from reut2-000.sgm after the text is tokenized into sentences.

```
<REUTERS NEWID="2">
<TEXT>
<BODY>
<s>Standard Oil Co and BP North America Inc said
they plan to form a venture to manage the money market
borrowing and investment activities of both companies.
</s>
<s>BP North America is a subsidiary of British
Petroleum Co Plc (BP), which also owns a 55 pct
interest in Standard Oil.</s>
<s>The venture will be called BP/Standard Financial
Trading and will be operated by Standard Oil under the
oversight of a joint management committee.</s>
</BODY></TEXT>
</REUTERS>
```

Figure 2. An SGML document extract from the Reuters corpus after the pre-processing

After the sentence boundaries are identified, each sentence is parsed using Eugene Charniak's maximum entropy inspired parser [2]. This probabilistic parser produces Penn tree-bank style trees and achieves 90.1% average accuracy for sentences not exceeding 40 words long and 89.5% for sentences with length under <100 words when trained and tested on the Wall Street Journal treebank. An example of a parse tree produced by Charniak's parser for the Reuters corpus is shown in Figure 3.

3 Semantic role labelling using VerbNet

The algorithm for semantic role identification of a sentence that we propose consists of the following three steps:

1. Firstly, for each clause in the sentence we identify the main verb and build a sentence pattern using the parse tree;
2. Secondly, for each verb in the sentence we extract a list of possible semantic frames from VerbNet, together with selectional restrictions for each semantic role;
3. Lastly, we match the sentence pattern to each of the available semantic frames, taking into account the semantic role's constraints. As a result a list of all possible semantic roles assignments is returned, from which we manually identify the best (correct) one.

The following sections describe each of these steps in more detail.

```
(S1 (S (ADVP (RB Earlier))
  (NP (NNP USAir))
  (VP (VBD said)
    (SBAR (S (NP (PRP it))
      (VP (VBD agreed)
        (S (VP (TO to)
          (VP (VB buy)
            (NP (NNP Piedmont))
            (PP (IN for)
             (NP (NP (NP (CD 69) (NNS dlrs))
              (NP (NN cash)))
               (PP (IN per) (NP (NN share))))))))))))
```

Figure 3. The parse tree produced by the Charniak's parser for the sentence **Earlier USAir said it agreed to buy Piedmont for 69 dlrs cash per share**

3.1 Constructing sentence patterns for verbs in Reuters

As mentioned earlier, during the pre-processing stage we produce a parse tree for each sentence using the Charniak's parser. From this parse tree we construct a sentence pattern for each clause of the sentence. The verb identification is done by selecting the main verb for every VP node in the parse tree. Each sentence pattern is a flat parse representation that identifies the main verb and the other main categories of the clause. For example, the sentence patterns constructed for the parse tree shown in Figure 3 are displayed in Figure 4 (both short and extended form).

ADVP, NP, **VERB(said)**, SBAR
ADVP(Earlier), NP(USAir), **VERB(said)**, SBAR(it agreed to buy Piedmont for 69 dlrs cash per share)

NP, **VERB(agreed)**, S
NP(it), **VERB(agreed)**, S(to buy Piedmont for 69 dlrs cash per share)

VERB(to buy), NP, PP
VERB(to buy), NP(Piedmont), PP(for 69 dlrs cash per share)

Figure 4. The semantic frames identified for the parse tree shown in Figure 3

Note that in the last pattern in Figure 4 there is no category preceding the verb. Nevertheless, in certain syntactic constructions, like *USAir agreed to buy Piedmont*, the subject of the verb in infinitive form (in this case *buy*) is the same as the subject of the verb preceding it (in this case *agree*), and therefore we can say that some of the categories preceding the first verb can be used as categories preceding the infinitive verb. Thus in our third pattern the verb *buy* can "inherit" the categories that precede the verb *agree* and can be rewritten as

(NP, **VERB(to buy)**, NP, PP)
(NP(it), **VERB(to buy)**, NP(Piedmont), PP(for 69 dlrs cash per share)

Currently, to resolve cases that may require subject inheritance like this, we keep three lists of verbs. The first list contains the most common verbs that occur in constructions like *Verb + Infinitive* (e.g. *afford*, *agree*, etc.), and for these we assume that the subject of the infinitive is the subject of the preceding verb, and therefore we are allowed to inherit the preceding categories.

The second list contains verbs that require the following verb to be gerund, or constructions like *Verb + Verb-ING* (e.g. *appreciate*, *avoid*, etc.). Similarly to the previous case, we allow the gerund to inherit the preceding categories of the first verb. Some of the verbs may be listed in both the first and the second list (e.g. *love*, *start*, etc.)

The third list contains verbs that occur in the *Verb + Object+ Infinitive* construction (such as *advise*, *permit*, etc.), and for them the infinitive should inherit as its subject the object of the first verb. Note that there are verbs that are members both of the first and the last list (e.g. the verb *want* as in *I want to go* and *I want him to go*).

In our corpus there are few cases when there is more than one category available to be "inherited" from the preceding verb and to identify the correct one we use the semantic roles' selectional constraints and the part of speech of the available constituents.

3.2 Extracting the possible semantic frames for a verb using VerbNet

VerbNet is a collection of verb classes and the members of each class share similar syntactic and semantic behaviour. Because a verb can be a member of more than one VerbNet class (for example the verb *make* is listed as a member of the verb classes *dub-29.3* and *build-26.1* each of which corresponding to different verb senses), the list of its possible semantic frames is a combination of the semantic frames defined in each of the classes in which it participates (currently we do not distinguish between different verb senses and therefore do not process the WordNet sense information attached to each verb class member).

Each verb frame in a VerbNet class is defined in four ways (see Figure 1) – it consists of a description of the language phenomena (e.g. Basic Transitive, Theme as direct object, etc.), one or more sample sentences, a semantic frame description, and a semantic predicate description defining the participant in the action or event during the different stages. Our algorithm extracts all the semantic frames in a class and defines them as possible semantic frames for each of the verbs that are members of this class. For example, from the VerbNet class **say-37.7**, shown in Figure 1, the semantic frames we extract are shown in Figure 5. Therefore, for the verbs that are members of the verb class **say-37.7** these would be some of the allowed sentence frames. If any of the verbs are also members of other classes we add the frames from these classes as well.

The extracted semantic frames should only be considered together with the defined selectional constraints for the semantic roles. For example, for the frames shown in Figure 5

Agent V Topic
Agent V Topic Prep(to) Recipient
Agent V Topic[+sentential -infinitival]
Agent V Prep(to) Recipient Topic[+sentential -infinitival]

Figure 5. Semantic frames extracted for the verbs in class **say-37.7**

the Agent and the Recipient should be *animate* or an *organization* and the Topic should be *communication*.

Agent[+animate OR +organization]
Topic[+communication]
Recipient[+animate OR +organization]

Figure 6. Selectional constraints for the semantic roles defined in class **say-37.7**

When a frame defines its own restrictions for a semantic role, the restrictions defined in the class are added to them provided they are not contradicting. For example, for the third and fourth frame in Figure 5 the role Topic should be *communication*, as well as *sentential* and not *infinitival*.

3.3 Matching algorithm

The match of the sentence pattern to each of the possible semantic frames for this verb is done in three steps. Firstly, we check the selectional constraints of each semantic role. This includes a check on whether a candidate constituent for a semantic role fulfills the selectional constraints. For example, a common requirement for a constituent to fill the role of *Agent* is to be of type *animate* or *organization*. The selectional constraints check is implemented using different techniques and is illustrated with a few examples.

Some of selectional constraints are resolved using data from WordNet [4]. For example the restriction *machine* is fulfilled if the word represented by the constituent is a member of a WordNet synset that is a hyponym of the WordNet synset containing the word *machine* (or in other words, if the word is a type of machine).

Other restrictions, like *infinitival* and *sentential*, are resolved by checking the structure of the parse tree.

Restrictions like *animate* and *organization* try to utilize the information provided in WordNet and pre-compiled lists of organization and personal names. If no satisfactory answer is found, we apply heuristics such as a check if the phrase contains proper nouns.

We also check for a suitable preposition before the constituent to be matched. For example, for the frame

Agent V Topic Prep(to) Recipient

the constituent to be matched to the semantic role Recipient should be part of a prepositional phrase containing the preposition *to* (e.g. *Bob said a few words* ***to Mary***).

Secondly, we match the required frame elements to the available sentence constituents, trying to find a match for each semantic role. If the number of the available constituents is less than the number of the required slots in the frame, the match fails. If there are more than one constituents available to fill a required slot in the semantic frame, the most probable one is used (priority for most roles is given to NP).

Lastly, if for a possible semantic frame we find allowed constituents to fill all the semantic roles slots that comply with the selectional constrains, the algorithm considers this a possible match.

3.4 Example

We will illustrate the algorithm for the verb *say* in the following example sentence from the Reuters corpus:

> *Earlier USAir said it agreed to buy Piedmont for 69 dlrs cash per share.*

The parse tree produced by Charniak's parser is shown in Figure 3. Firstly, we identify the sentence frame pattern. Each tree with an S (or SBAR, SINV etc.) node as a root represents a separate clause and therefore is treated separately. The sentence frames identified are shown in Figure 4 and we process them individually.

So, for the verb *said*, we try to match the sentence frame

> ADVP(Earlier), NP(USAir), **VERB(said)**, SBAR(it agreed to buy Piedmont for 69 dlrs cash per share)

to each of the frames shown in Figure 5. The list is split into two parts, one containing the nodes before the verb and the other, the nodes after the verb; each of which are matched separately and the result is then combined. Failing to match either of the lists causes the overall match to fail.

We can directly discard the second and the fourth frames as possible matches because they require two roles following the verb and our sentence pattern has only one available role filler (SBAR).

For the first frame we try to match

> ADVP(Earlier), NP(USAir)

to

> Agent[+animate OR +organization]

and

> SBAR(it agreed to buy Piedmont for 69 dlrs cash per share)

to

> Topic[+communication]

To fill the role of the Agent we have two possible phrases – ADVP(Earlier) and NP(USAir). USAir is listed as a known organization name, and together with the fact that the *Agent* role requires a noun phrase filler, while adjective phrases usually represent manner, location or time, the algorithm assigns NP(USAir) to be the *Agent*. Further on, the match of *SBAR(it agreed to buy Piedmont for 69 dlrs cash per share)* to

Topic[+communication] succeeds (currently a check for *communication* selectional constrains always succeeds). Therefore the first frame is a possible match.

Similarly, matching the third frame succeeds. Here the selectional constraints for the Topic are to be *communication*, *sentential* (i.e. the root is not allowed to be S, SBAR, etc.) and not *infinitival* (i.e. the phrase can not start with *to* followed by a verb), which are easily checked.

As a result we are presented with frames one and three, shown in Figure 5 as two possible solutions, both of which mark *USAir* as the Agent and *it agreed to buy Piedmont for 69 dlrs cash per share* as the Topic of the verb *say*.

4 Experimental results

We have tested the coverage and performance of our algorithm on one of four files provided by Reuters, reut2-003.sgm. The number of verbs that are present (including repetitions) is 13421, 32.38% of which are not described in VerbNet and 1.72% are phrasal verbs and therefore also not present in VerbNet. Therefore VerbNet covers 65.9% of the verbs present in the Reuters corpus. Approximately 5.05% of the recognized verbs are in passive voice, and because our system currently does not process passive voice constructions, that gives us a target accuracy of 60.84% (the percentage of the verbs that have an entry in VebNet and are in active voice).

The verb is not present in VerbNet or is phrasal	31.1%
The verb is in passive form	5.9%
Assigns both correct and incorrect semantic roles	6.7%
Correctly identified semantic roles	32.6%
There are no frames defined in VerbNet for this sense of the verb	9.6%
The frame for the verb realised in the text is not defined in VerbNet	7.4%
Wrong semantic roles assignment due to an incorrect parse tree	5.2%

Table 1. Results of the evaluation of the algorithm on Reuters corpus

To evaluate the performance of the algorithm we randomly selected 1% of the verbs and manually analysed the assigned semantic roles. The summarized results are represented in Table 4. Our tests show that 31.1% of the verbs are either phrasal verbs or are not described in VerbNet and 5.9% are verbs in passive form. The algorithm correctly identifies the semantic roles in 39.3% of all cases.

5 Conclusions

In this paper we described an approach for automatic semantic role identification using syntactic and semantic information from VerbNet. We tested the algorithm on the Reuters corpus. The archived accuracy is strongly influenced by the lack of VerbNet description of many verbs present in the corpus, as well as the lack of a possible semantic frame for the present verb sense. The work on the system is ongoing and more efforts are required to implement the verb sense disambiguation and to test the algorithm on different corpus.

REFERENCES

[1] T. Bleam, M. Palmer, and V. Shanker, 'Motion verbs and semantic features in tag', in *Proceedings of the TAG+-98 Workshop*, Philadelphia, PA., (1998).

[2] Eugene Charniak, 'A Maximum-Entropy-Inspired Parser', in *Proceedings of NAACL-2000*, pp. 132–139, (2000).

[3] Hoa Trang Dang, Karin Kipper, Martha Palmer, and Joseph Rosenzweig, 'Investigating regular sense extensions based on intersective Levin classes', in *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics*, eds., Christian Boitet and Pete Whitelock, pp. 293–299, San Francisco, California, (1998). Morgan Kaufmann Publishers.

[4] *WordNet: An Electronic Lexical Database*, ed., Christiane Fellbaum, MIT Press, May 1998.

[5] Charles J. Fillmore, *The case for case*, 1–88, Holt, Rinehart, and Winston, New York, 1968.

[6] Daniel Gildea and Daniel Jurafsky, 'Automatic Labeling of Semantic Roles', in *Proceedings of 38th Annual Conference of the Association for Computational Linguistics (ACL-00)*, pp. 512–520, Hong Kong, (October 2000).

[7] Daniel Gildea and Daniel Jurafsky, 'Automatic Labeling of Semantic Roles', *Computational Linguistics*, **28**(3), 245–288, (2002).

[8] Beth Levin, *English Verb Classes And Alternations: A Preliminary Investigation*, The University of Chicago Press, 1993.

[9] Scott Miller, David Stallard, Robert Bobrow, and Richard Schwartz, 'A fully statistical approach to natural language interfaces', in *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 55–61, Santa Cruz, CA, (June 1996). Morgan Kaufmann Publishers, Inc.

[10] Reuters. Reuters-21578 text categorization collection. http://kdd.ics.uci.edu/databases/reuters21578/-reuters21578.html, 1987.

[11] Ellen Riloff, 'Automatically Constructing a Dictionary for Information Extraction Tasks', in *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*, pp. 811–816. AAAI Press/The MIT Press, (1993).

Analysing the Relationship between Epistasis and Inversion

Seamus Hill [1] and **Colm O'Riordan** [2]

Abstract.

This paper presents results on the use of problem generators to analyse the performance of a genetic algorithm, which combines selection, crossover, mutation and an inversion operator. Inversion is used as a reordering mechanism to change the linkage between genes in a chromosome. The main conjecture is that inversion may be useful for particular classes of problems depending on the levels and type of epistasis present. Inversion works by reversing the order of genes between two randomly chosen positions within the chromosome. While other genetic operators search for good combinations of alleles, an inversion operator has the ability to search among good string arrangements. Interaction (also called epistasis) between genes means that the contribution of a gene to the fitness depends on the fitness of other genes in the chromosome. By using a number of problem generators to alter both the level and type of epistasis and by varying the rate of inversion, we develop an empirical methodology to analyse a classic inversion operator in a simple genetic algorithm and present results of such analysis.

1 Introduction

The focus of this paper is to present an empirical methodology for conducting analysis on the use of an inversion operator, and on results regarding the potential usefulness of the inversion operator. The main hypothesis is that the use of an inversion operator can be of benefit in searching the search space for particular classes of problems. The term epistasis refers to any kind of effective interaction among genes. In Genetic Algorithms (GAs), epistasis is considered to be closely related to the difficulty of a problem. Higher levels of epistasis normally indicate more difficult problems, however, it should be noted that exceptions exist. The experiments conducted are an attempt to identify where inversion may be of use, and to illustrate how the level and type of epistasis affects the success of an inversion operator.

Many comparative studies on Evolutionary Algorithms (EAs) are carried out on a set of test problems and results are used to highlight the merits of an evolutionary algorithm for a particular set of problems. However, these results may not generalise beyond the test problem used[13]. This results in a situation where neither the theoretical models nor the empirical studies produced can make accurate predictions about the performance of EAs on many classes of problems. To strengthen the results obtained from experiments conducted, we test an EA using problem generators which act as independent agents and remove the ability to hand tune the algorithm during evaluation. Problem generators allow results to be obtained over a set of problems rather than a number of preselected problems. This increases the predictive power of the results for the problem class as a whole. The paper is organised as follows. Firstly, the NK landscape generator[14], the Random L-SAT Boolean satisfiability generator[19], and the Multimodal generator[15] are outlined in Section 2. Section 3 outlines the inversion operator. Section 4 describes experiments conducted and results obtained and finally, Section 5 outlines the conclusions and possible future work.

[1] Dept. of Information Technology, National University of Ireland, Galway. Email: seamus@it.nuigalway.ie

[2] Dept. of Information Technology National University of Ireland, Galway Email: colmor@geminga.it.nuigalway.ie

2 Problem Generators

Problem generators remove the ability to hand-tune algorithms to a particular class of problems and are therefore useful in experimenting with EAs given that they can be parameterised to allow the construction of well designed experiments. We use three problem generators to randomly generate problem landscapes.

2.1 NK Landscape Generator

The NK model provides a means for constructing fitness landscapes that can be tuned from smooth to rugged. The main parameters of the model are N, the number of genes in the genotype, i.e. the length of the strings that form the points in the landscape, and K, the number of other genes that epistatically influence a particular gene (i.e., the fitness contribution of each gene is determined by the gene itself plus K other genes)[12]. K sets the level of epistasis by determining the dependence the partial fitness of a gene at location n has on the genes in a neighbourhood of K other locations. The neighbourhood may be at the K locations nearest to n in the genotype or a set of K locations randomly picked from anywhere on the genotype.

Following this, a series of N look-up tables are then generated, one for each gene location in the genotype. Each table has 2^{K+1} random entries in the interval (0,1). The fitness,

f_{chrom}, of a particular genotype is calculated by the function:

$$f_{chrom} = \frac{1}{N}\sum_{n=1}^{N} f(locus_n)$$

where the fitness contribution of each locus, $f(locus_n)$, is determined by using the (binary) value of gene n together with the values of the K interacting genes. The partial fitness $f(locus_n)$ is obtained from the nth look-up table using the values of the genes in location n and its neighbourhood as the look-up key[18].

There are a number of problems associated with the creation of an NK landscape problem. The main problem is the space required to store the tables created. $N^{2^{K+1}}$ separate tables are required. This restricts the size of the models used in experiments. A further limitation associated with NK landscape problems is that all genes have an identical degree of epistasis (K), whereas most "real world" problems vary substantially in the amount of epistasis among subsets of genes.

2.2 Random L-SAT Generator

Boolean satisfiability problems are NP-complete and are often used as a canonical representative of this class of problems[22]. The most general statement of Boolean satisfiability (SAT) problems is as follows: given an arbitrary Boolean expression involving V Boolean variables, find an assignment of truth values to the V Boolean variables that makes the whole expression true (representable in a canonical form). These forms can be simplified by assuming that all clauses are of the same length L, which makes it easier to quantify the level of complexity exhibited by a Boolean expression in terms of the number of conjunctive clauses C.

The Random LSAT generator creates random problems in conjunctive normal form (CNF) subject to the three parameters outlined (V, C, and L). Every clause is generated by selecting L of the V variables and then negating each variable with a probability of 0.5. By controlling and varying these parameters, the type and amount of epistasis can be varied. The fitness function is the fraction of clauses that are satisfied by the assignment. The fitness, f_{chrom}, of a chromosome is calculated by the function:

$$f_{chrom} = \frac{1}{C}\sum_{n=1}^{C} f(clause_n)$$

where the fitness contribution of each clause, $f(clause_n)$, is 1 if the clause is satisfied and 0 otherwise. The Random L-SAT generator has a much smaller storage requirement than that of the NK landscape generator, thus allowing larger experiments to be conducted. Furthermore, the amount of epistasis varies among subsets of genes producing more realistic representations of practical problems than the uniform levels of epistasis exhibited by the NK model.

2.3 Multimodal Generator

The Multimodal problem generator produces random problems with a controllable degree of multimodality. The generator creates a set of P random N-bit strings, which represent the location of P peaks in the search space. In order to evaluate the fitness of a bit string, the nearest peak must be located (in Hamming Space). The fitness of the string, $f(chrom)$, is the number of bits the string has in common with that nearest peak, divided by N. The optimum fitness for an individual is 1. Fitness is calculated by the function:

$$f_{chrom} = \frac{1}{N}\max_{n=1}^{p}\{N - Hamming(chrom, Peak_n)\}$$

Problems generated by the multimodal generator which have a small number of peaks have a low level of epistasis, while those with a large number of peaks have a high level of epistasis.

3 Inversion Operator

Genetic Algorithm theory as proposed by Holland[11] was founded on four operators: proportional selection, one-point crossover, mutation and inversion. The dominant natural mechanism responsible for recoding a problem is the inversion operator[8]. Holland proposed inversion as a method to increase the linkage of schemata which were highly fit. In other words, if the population contain orderings which are considered bad i.e. where alleles which are highly epistatic are spread over great distances on the chromosome, crossover will with a high probability destroy important allele packets. Reordering can rearrange allele placement. This increases the probability of a building block to survive the disruptive effect of one-point crossover, and to combine with other highly fit building blocks, as per Goldberg's *Building Block Hypothesis*[8][3].

The inversion operator used in this paper performs a "classical" inversion, that is, two points are chosen on the chromosome, the chromosome is cut at those points, and the points of the cut section are reversed. However, it is assumed that an allele's meaning is linked to its locus, whereas in nature an allele's meaning is position-independent. To provide the same flexibility in a GA's representation, the alleles can be named using integers. When the inversion operator is applied to the extended representation, the allele's name information is carried as well. This means that the bit values retain their intended meaning regardless of their position in the string, allowing inversion to produce orderings in which beneficial schemas are more likely to survive. Consider, if in the original ordering the schema `10**01**` is highly fit, under the new ordering in the above example, that schema is `1010****`. This important schema will now have a stronger chance of survival under single-point crossover than the original string.

Inversion operates by randomly selecting two points in the string and reversing the order of the bits between them. For example, if the string `10010101` is encoded as

$$(1,1)(2,0)(3,0)(4,1)(5,0)(6,1)(7,0)(8,1)$$

[3] Although the schema theorem and the building block hypothesis have been criticised in recent years, i.e. Altenberg[1], Macready & Wolpert[17] and Fogel & Ghozeil[5] [6], more recent research carried out by Stephens and Waelbroeck[23] [24], and Poli[21] [20] indicate the usefulness of the schema theorem.

then when inverted after the second and before the seventh locus, the string appears as

$$(1,1)(2,0)(6,1)(5,0)(4,1)(3,0)(7,0)(8,1).$$

An inversion map is created to record the position of the indices after reordering. This inverting does not effect the fitness of the string, the linkage however, has changed. Inversion has no immediate effect on the string fitness as strings are returned to their original ordering prior to being sent to the fitness function for evaluation.

A problem which exists in combining inversion with single-point crossover, is to ensure that each offspring is created with a full set of loci. If two parents

$$(1,1)(2,0)(6,1)(5,0)(4,1)(3,0)(7,0)(8,1)$$

$$(1,0)(2,0)(3,1)(4,0)(5,1)(6,0)(7,1)(8,0)$$

are crossed after the third locus. The offspring created are

$$(1,1)(2,0)(6,1)(4,0)(5,1)(6,0)(7,1)(8,0)$$

$$(1,0)(2,0)(3,1)(5,0)(4,1)(3,0)(7,0)(8,1).$$

The first offspring has two copies of bit 6 and no copies of bit 3, while the second offspring has two copies of bit 3 and no copies of bit 6. In order to ensure that crossover will produce offspring with a full set of loci, each individual in the population makes the same alteration in its chromosome as just made on the inversion map.

There are parallels between the way alleles are named using integers in the inversion operator and the way they are named in messy GAs (mGAs). mGAs work by searching for tight building blocks and combine them together to form the optimal, they guarantee that appropriate building blocks will be formed and they will be tight[10]. The reason mGAs are called messy is that they use variable-length strings that may be under or overspecified with regard to the problem being solved. In mGAs a three bit string `1 1 1` can be represented as `((1 1)(2 1)(3 1)`. Also with mGAs strings can contain too few or too many bits e.g. `((1 1)(2 1))` and `((1 0)(2 1)(3 1)(1 0))` are both valid in a three bit problem. mGAs deal with the problem of two first bits in the second example by using a first-come-first-served rule and have been used with a left-to-right scan to arbitrate such conflicts[4]. Underspecification is solved by the use of a competitive templates. Competitive templates allow the evaluation of partial strings. The underspecification fills unnamed genes with a locally optimal structure. To implement this a string specifying an allele at each gene position is chosen as a template and the unspecified genes in a string are borrowed from the template. Let us use the above example, if `0 0 0` is the chosen template, the string `((1 1)(2 1))` becomes `((1 1)(2 1)(3 0))` where the third string is taken from the template. How the template is actually selected is beyond the scope of this paper, for a detailed description see[10].

Our inversion operator differs from Goldberg's mGA in that we do not have to deal with under or overspecification as we use strings of a fixed length which are reordered to match the alterations made on the inversion map.

A number of authors have undertaken research on the application of inversion in GAs, including Bagley[2] Cavicchio[3] and Frantz[7], each reporting different results. Others have included inversion in the structure of their algorithm e.g. Whitley[26], Syswerda[25] and Wienholt[27]. Whitley recognised that inversion may have another role in searching the search space in that it appears to provide an element of nondestructive *noise* that helps crossover escape a local maxima. Syswerda suggested using inversion as a method to change the representation of the chromosome before using one-point crossover. This changed representation may be beneficial for a GA when tight linkage is required. Wienholt included inversion in a refined GA which attempted to mirror in more detail the biological model, stating that inversion works at the DNA level. He created a genotype as well as a phenotype inversion, which produces different information at two different levels.

Research by Goldberg and Bridges[9] shows how reordering can overcome the combination of loose linkage and deception which cause GAs to diverge. Although the analysis raises questions as to whether unary reordering operators are quite slow in searching for tight linkages, the work does show how tight building blocks can be preferred over looser building blocks in epistatic problems even when operators such as the inversion operator have no effect on a string's fitness. Further research into the use of an inversion operator includes work by Kennedy[16] where a new double stranded encoding scheme is used in conjunction with a simple inversion operator.

4 Experiments

The initial experiments compare the average best fitness levels achieved with the inversion rate set at 0% and 1%. The rate of crossover is 60% and the mutation rate is 0.031. The aim is to illustrate how varying the level and type of epistasis can have an effect on the usefulness of inversion. The empirical studies report the performance of the algorithm for a population of 100 chromosomes for 200 generations over 200 runs.

4.1 NK Experiments

The first set of experiments conducted were on the NK Landscape problems. The parameters for the NK model were $N = 32$ and K varying from 0 to 31. Figures 1, 2 and 3 compare results achieved with K set to 0, 16 and 31. They indicate that as the GA is applied to more difficult landscapes, caused by higher levels of epistasis, it benefits from the introduction of an inversion operator at a low rate. In other words, the algorithm performs better on the more difficult landscape (i.e. K=31) when the inversion operator is at 1% than when it is set to 0%.

The figures show the average best fitness levels achieved. In figure 1, the search space is easier to search and inversion proves to have a disruptive influence and produces lower levels of fitness. When the level of epistasis is increased to $K = 16$, as illustrated in figure 2, the performance of the GA with inversion has moved closer to the results achieved by the

GA without inversion. Bearing in mind the extra overhead involved, it is still not practical to include an inversion operator.

However, as K increases towards $K = 31$, the use of inversion at a rate of 1% produces significantly higher fitness levels. Figure 3 illustrates this point showing how inversion set to 1% benefits the search when $K = 31$, producing higher rates of fitness than achieved with inversion set to 0%. This indicates that for the class of problem produced by the NK problem generator the use of a inversion operator at 1% appears to be of benefit in searching the search space at higher levels of epistasis.

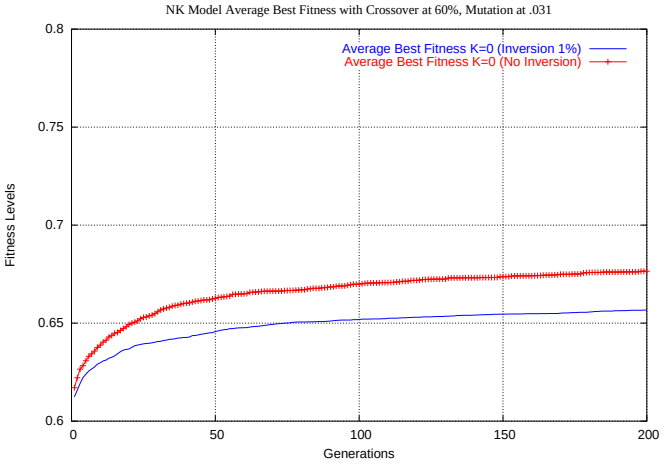


Figure 1. NK Model: K=0 with Inversion at 0% & 1%.

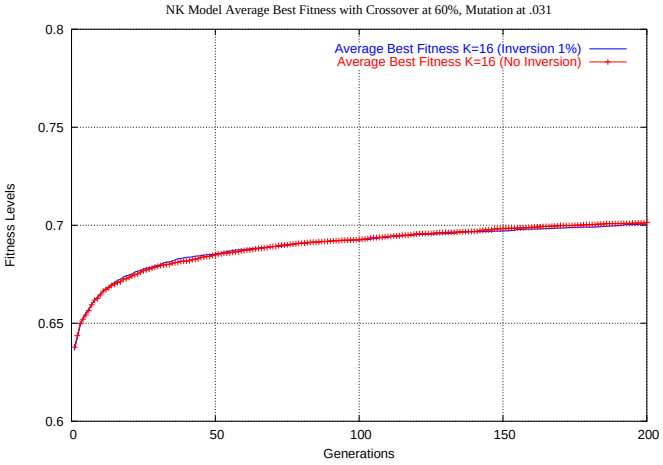


Figure 2. NK Model: K=16 with Inversion at 0% & 1%.

4.2 L-SAT Experiments

The next set of experiments conducted were the Random L-SAT problems. We again compare the performance of GAs on landscapes with various levels of epistasis and an inversion rate set at 0% and 1%. The variables for the Random

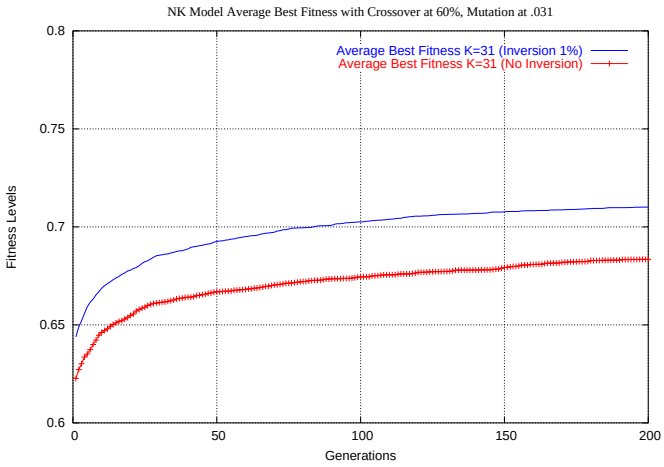


Figure 3. NK Model: K=31 with Inversion at 0% & 1%.

L-SAT experiments were fixed as follows: variables V were set to 100, the clause length L was set to 3, the population to size 100 and C varied from 200 through to 3200. Figures 4 $C = 200$, 5 $C = 2400$ and 6 $C = 25000$ illustrate that the GA using inversion produces higher levels of fitness than a GA operating without inversion at each level of epistasis. As the number of clauses increases, i.e., from 200 clauses (low epistasis) to 3200 clauses (high epistasis), fitness levels achieved decrease as the problems become more complex. The relation-

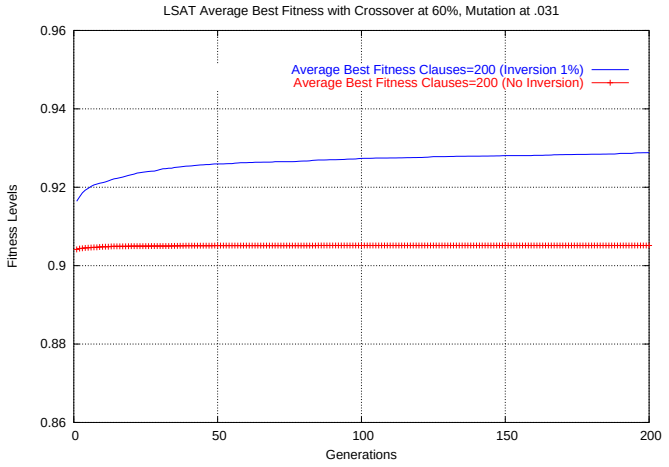


Figure 4. Random L-SAT: C at 200, inversion at 0% & 1%.

ship between changes in epistasis and corresponding changes in fitness levels is far more marked in problems generated by the L-SAT model than those generated by the NK model, thereby indicating a more difficult set of landscapes for a simple GA to search. However, the differences between the fitness levels obtained, when including and not including inversion, decrease as the levels of epistasis increase. This differs from the results of the NK Landscape experiments, indicating that problems which exhibit similar characteristics to an L-SAT problem may benefit from the use of inversion, depending on

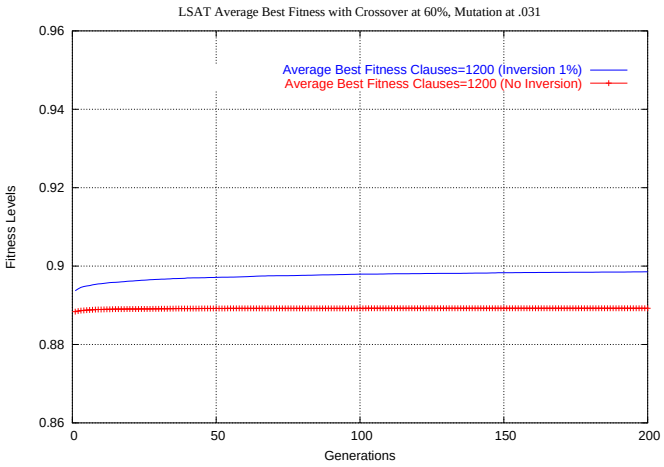


Figure 5. Random L-SAT: C at 1200, inversion at 0% & 1%.

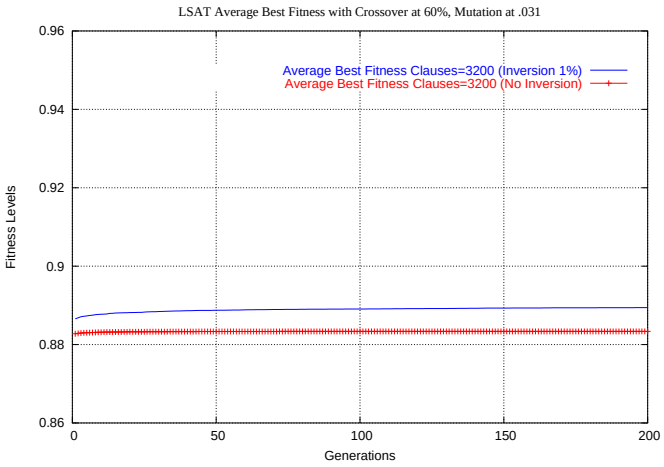


Figure 6. Random L-SAT: C at 3200, inversion at 0% & 1%.

the level of epistasis present. The uniformity of the epistasis may be an important feature to consider prior to introducing an inversion operator, as the landscapes generated by the NK model produce an identical level of epistasis, whereas, the landscapes produced by the L-SAT generator produce epistasis which varies among subsets of genes producing a more realistic representation of practical problems.

4.3 Multimodal Experiments

For the Multimodal experiments the crossover, mutation and inversion parameters were as before, while the number of peaks varied from 1 to 15000. Figures 7 and 8 illustrate average best fitness levels achieved when $P = 1$ and $P = 50000$ respectively. They show that at both lower and higher levels of epistasis, the inversion operator appears to be of no benefit in searching the search space for the class of problems generated by the multimodal generator.

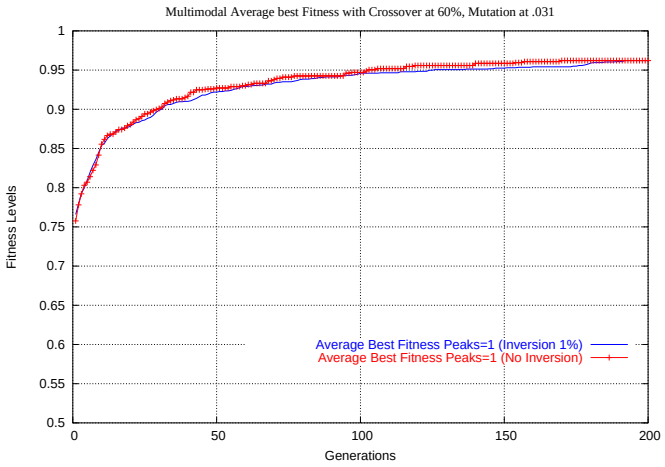


Figure 7. Multimodal: Peaks at 1. Inversion at 0% & 1%.

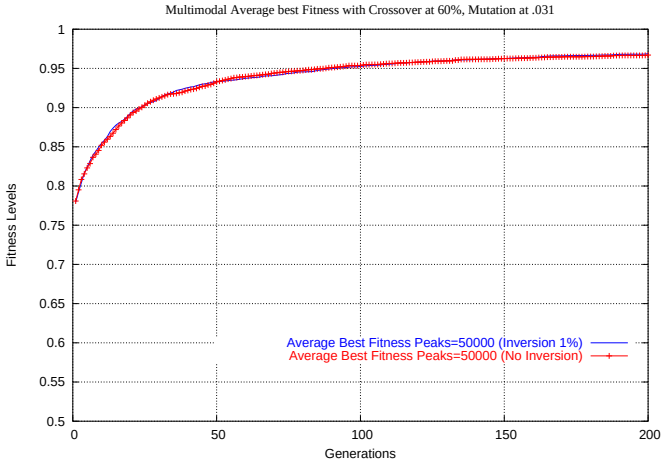


Figure 8. Multimodal: Peaks at 50000. Inversion at 0% & 1%.

5 Conclusion & Future Work

The findings of the experiments conducted show that there are possibilities for using inversion successfully. The results indicate that the success of inversion depends on the level and uniformity of epistasis contained within the problem. By using randomly generated classes of problems we have shown that in many cases, the inversion operator has improved the search ability of the GA. Moreover, in the L-SAT experiments, the gain achieved by use of the inversion operator decreased as the complexity grew. This differs from results obtained with the NK-model. We suggest the differences are based solely on the different forms of epistasis found in the landscape. The results from the Multimodal experiments indicate that for this class of problems there appears to be no benefit gained by including an inversion operator, and the extra overhead involved would appear to rule out the use of inversion on landscapes similar to those created by the multimodal generator. Again, we believe that this is due to the form of epistasis found in the multimodal landscape. We feel that the overall results indicate that the use of an inversion operator can be of benefit in

searching the search space for particular classes of problems, and that the levels, type and uniformity of epistasis found in a landscape are critical to successfully using inversion in a GA. Future work extending from this includes;

- The development of an adaptive reordering operator, which is introduced into the algorithm once a certain number of criteria have been reached, thereby not adding the overhead of reordering until the search has slowed.
- Using other types of representations to see how changes in the type of representations used in a GA affects the GAs performance.
- Applying the same methodology to other Evolutionary Algorithms.

6 Acknowledgments

The first author acknowledges the support of the Irish Research Council for Science, Engineering and Technology under scholarship RS/2002/697-2.

REFERENCES

[1] L. Altenberg, 'The Schema Theorem and Price's Theorem', in *Foundations of Genetic Algorithms 3*, eds., Darrell Whitley and Michael D. Vose, 23–49, Morgan Kaufmann, San Mateo, CA, (1995).

[2] J. D. Bagley, *The behaviour of adaptive systems which employ genetic and correlation algorithms.*, Ph.D. dissertation, (Doctoral dissertation, University of Michigan) Dissertation Abstracts International, 28(12),5106B. (University Microfilms No. 68-7556), 1967.

[3] D. J. Cavicchio, *Adaptive search using simulated evolution.*, Ph.D. dissertation, Unpublished doctoral dissertation, University of Ann Arbor, 1970.

[4] K. Deb and D. Goldberg, 'mga in c: A messy genetic algorithm in c', Technical Report Report no. 91008, Illinois Genetic Algorithms Laboratory (IlliGAL), (1991).

[5] D. B. Fogel and A. Ghozeil, 'Schema processing under proportional selection in the presence of random effects', *IEEE Transactions on Evolutionary Computation*, **1**(4), 290–293, (1997).

[6] D. B. Fogel and A. Ghozeil, 'The schema theorem and the misallocation of trials in the presence of stochastic effects', in *Evolutionary Programming VII: Proc. of the 7th Ann. Conf. on Evolutionary Programming*, eds., D. Waagen V.W. Porto, N. Saravanan and A.E. Eiben, pp. 313–321. Springer. Berlin, (1998).

[7] D. R. Frantz, *Non-linearities in genetic adaptive search.*, Ph.D. dissertation, (Doctoral dissertation, University of Michigan) Dissertation Abstracts International, 33(11),5240B-5241B. (University Microfilms No. 72-11116), 1972.

[8] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Company, Inc., Reading, MA, 1989.

[9] D. E. Goldberg and C. L. Bridges, 'An analysis of a reordering operator on a ga-hard problem', *Biological Cybernetics*, **62**, 397–405, (1990).

[10] D. E. Goldberg, K. Deb, and B. Korb, 'Don't worry, be messy', in *Proc. of the Fourth Int. Conf. on Genetic Algorithms*, pp. 24–30. Morgan Kaufmann, (1991).

[11] J. H. Holland, *Adaptation in natural artificial systems*, University of Michigan Press, Ann Arbor, 1975.

[12] W. Hordijk, *Population Flow on Fitness Landscapes*, Master's thesis, Erasmus University, Rotterdam, 1994.

[13] K. A. De Jong, M. A. Potter, and W. M. Spears, 'Using problem generators to explore the effects of epistasis', in *Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA97)*, ed., Thomas Bäck, San Francisco, CA, (1997). Morgan Kaufmann.

[14] S. A. Kauffman, *Origins of Order*, Oxford University Press, Oxford., 1993.

[15] J. Kennedy and W. Spears, 'Matching algorithms to problems: an experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator', in *Proceedings of the IEEE World Congress on Computational Intelligence, pp. 74–77, Anchorage: Alaska, May 1998. IEEE.*, (1998).

[16] P. J. Kennedy and T. R. Osborn, 'A double-stranded encoding scheme with inversion operator for genetic algorithms', in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, eds., Lee Spector, Erik D. Goodman, Annie Wu, W.B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, pp. 398–407, San Francisco, California, USA, (7-11 July 2001). Morgan Kaufmann.

[17] W. G. Macready and D. H. Wolpert, 'On 2-armed gaussian bandits and optimization', Technical Report 96-03-009, Santa Fe, NM, (1996).

[18] G. Mayley, 'The evolutionary cost of learning', in *From Animals to Animats 4: Proc. of the Fourth Int. Conf. on Simulation of Adaptive Behavior*, eds., Pattie Maes, Maja J. Mataric, Jean-Arcady Meyer, Jordan Pollack, and Stewart W. Wilson, Cambridge, MA, (1996). The MIT Press.

[19] D. G. Mitchell, B. Selman, and H. J. Levesque, 'Hard and easy distributions for SAT problems', in *Proceedings of the Tenth National Conference on Artificial Intelligence*, eds., Paul Rosenbloom and Peter Szolovits, pp. 459–465, Menlo Park, California, (1992). AAAI Press.

[20] R. Poli, 'Why the schema theorem is correct also in the presence of stochastic effects', in *Proceedings of the 2000 Congress on Evolutionary Computation CEC00*, pp. 487–492, La Jolla Marriott Hotel La Jolla, California, USA, (6-9 2000). IEEE Press.

[21] Riccardo Poli and William B. Langdon, 'Schema theory for genetic programming with one-point crossover and point mutation', *Evolutionary Computation*, **6**(3), 231–252, (1998).

[22] W. M. Spears and K. A. De Jong, 'Using genetic algorithms to solve np-complete problems.', in *Proc. of the Third Int. Conf. on Genetic Algorithms*, ed., J.D. Schaffer, pp. 124–132, San Mateo CA, (1989). Morgan Kaufmann.

[23] C. R. Stephens and H. Waelbroeck, 'Effective degrees of freedom in genetic algorithms and the block hypothesis', in *Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA97)*, ed., Thomas Bäck, San Francisco, CA, (1997). Morgan Kaufmann.

[24] C. R. Stephens and H. Waelbroeck, 'Schemata evolution and building blocks', *Evolutionary Computation*, **7**(2), 109–124, (1999).

[25] G. Syswerda, 'Uniform crossover in genetic algorithms.', in *Proc. of the Third Int. Conf. on Genetic Algorithms*, ed., J.D. Schaffer, pp. 2–9, San Mateo CA, (1989). Morgan Kaufmann.

[26] D. Whitley, 'Using reproductive evaluations to improve genetic search and heuristic discovery.', in *Genetic Algorithms and their Applications: Proc. of the second Int. Conf. on Genetic Algorithms*, ed., J. J. Grefenstette, pp. 108–115. Lawrence Erlbaum Associates Publ., (1987).

[27] W. Wienholt, 'A refined genetic algorithm for parameter optimization problems', in *Proc. of the Fifth Int. Conf. on Genetic Algorithms*, ed., S. Forrest, pp. 589–596, San Mateo CA, (1993). Morgan Kaufmann.

Computationally Efficient Pricing for Resource Providers in a Grid Environment [1]

Alan Holland and **Barry O'Sullivan** [2]

Abstract. Ensuring truthfulness amongst self-interested agents who are bidding against one another in an auction is computationally expensive. The Vickrey-Clarke-Groves mechanism guarantees that each agent's dominant strategy is to tell the truth, but it requires solving $n+1$ optimisation problems for n agents. This paper presents an algorithm for computing prices for all agents in the same asymptotic time complexity as solving just one problem in the case of n agents bidding for m tasks where $m < n$, thus solving an open problem presented by Nisan and Ronen [14] for the specific case of task scheduling. This problem was also identified by Hershberger and Suri [9]. We also propose the use of constraints in conjunction with Operations Research algorithms to further improve the performance of truthful mechanisms in the context of a grid scenario, where resource providers are competing for tasks in an auction.

1 Introduction

The Grid has most recently been defined as *"coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organisations"* [5]. This new definition presents difficulties in terms of scheduling and brokering for distributed applications. We consider the service providers in the Grid as self-interested agents whose principle motivation is to maximise their own profit when bidding for tasks. Truthfulness in such a scenario is essential, but implementing it within a mechanism is computationally expensive.

Resource sharing in grids is not merely concerned with file exchange or CPU harvesting but rather direct access to software, data, instrumentation devices and other resources, as is required by a range of collaborative problem-solving and resource-brokerage strategies emerging in science and industry. The vision for the future of the grid is analogous to that of the electricity grid. Computing power, storage facilities, specialised software and other resources should be provided on-demand and the source of the desired resource should be transparent to the end-user. These resources are ultimately provided by a wide range of Grid Service Providers (GSPs), who can be regarded as self-interested agents motivated by profit and may be dishonest if they deem it beneficial. These agents can pool their resources in a co-operative in which a broker disseminates tasks based upon solicited bids for the given tasks.

Eliciting truthful responses from self-interested agents has been previously studied in game theory and economics. A class of Vickrey-Clarke-Groves mechanisms have been developed whereby the dominant strategy for any agent is to tell the truth, meaning that rational agents maximise their utility by truthfully revealing their preferences. [18]

[1] Funding from Enterprise Ireland, Research Innovation Fund (Grant Number RIF-2001-317).

[2] Cork Constraint Computation Centre, Dept of Computer Science, University College Cork, Ireland. email: [a.holland| b.osullivan]@cs.ucc.ie

This is achieved by soliciting bids from the agents for a range of tasks to be scheduled. An optimal solution is found, and subsequently the price paid to each agent is determined by finding the cost of an optimal solution without that agent present. Each agent cannot control the price paid to it because it is a function of the other agents preferences. This removes any potential profiteering by insidious providers. Mas-Collel et al have provided a proof of the VCG mechanisms truthfulness [13].

The main contributions of this paper are as follows:

- We present an algorithm that reduces the asymptotic time complexity for solving Vickrey prices for n agents in a task auctioning scenario where m tasks are being auctioned to that of solving just one problem.
- We outline how constraints may be used to further reduce the computational complexity for pricing problems.

This paper is organised as follows. Section 2 outlines the how a truthful mechanism is implemented in a task scheduling scenario and describes some of its drawbacks. Section 3 describes the Assignment Problem which is relevant to online scheduling for m tasks for n machines where $m < n$. It covers a general description of the problem, and describes in depth an Operations Research algorithm called the Hungarian Method that can be used to solve the problem. Section 4 describes an algorithm for computing all the solutions to the m Vickrey pricing problems in the same asymptotic time complexity as solving one problem, and Section 5 shows experimental results for this algorithm and Section 6 examines potential benefits of CP in this scenario and describes some related work. Section 8 summarises our conclusions and describes future work we plan to undertake in this area.

2 Mechanism Design and Task Scheduling

In the case of task scheduling each agent owns its own resources and his bid type is his cost to complete a given task using these resources. The task description contains the due date and Quality of Service requirements, and if a Grid Service Provider cannot match these demands, his bid for that task is ∞. The mechanism computes an outcome and each agent has a valuation for outcome o, based on its type. This valuation function $v^i(t^i, o)$ determines agent i's preference for outcome o, given its cost. The valuations would then be the negation of these respective costs.

The mechanism then assigns payment p^i to the agent i and the utility of each agent, u^i, is determined using the following equation. [3]

$$u^i = p^i + v^i(t^i, o)$$

[3] The mechanism function requires a common currency, outside of the scope of this discussion.

2.1 Mechanism Goals

The goal of each agent is to maximise its utility, whereas the goal of the mechanism in our case is to try to find the cheapest way of scheduling a set of given tasks so that they meet all their timing and quality of service constraints. This problem is NP-complete and therefore a more efficient means of calculating prices by solving $n+1$ optimisation problems is highly desirable. Hershberger and Suri solved an open problem presented by Nisan and Ronen [14], when they presented an algorithm for reducing the computational complexity of Vickrey payments [9]. They examined the specific case of shortest path routing in networks, and reduced the overall time complexity to the same asymptotic complexity of one agent. This proved that the time complexity for calculating Vickrey payments can be reduced in scenarios where the goal is shortest path routing, but they also posed the question, "*Can one achieve similar improvements in other network settings*"? [9]

Hershberger and Suri also identified the problem of auctioning m tasks amongst n self-interested agents, where $m \leq n$, and how an optimal task allocation requires solving the min-cost bipartite matching problem. This is also referred to as the *Assignment Problem*, and an algorithm for solving it is the *Hungarian Method*. However, determining payments for these agents requires solving a further m matching problems.

2.2 Drawbacks of truthful mechanisms

It has been shown that if non-optimal solutions are found to the optimisation problems that determine the prices paid in VCG mechanisms, then the mechanism is no longer guaranteed to be truthful [15]. This is a major drawback because various polynomial time heuristics and approximation algorithms can provide good or near optimal solutions very quickly. Nisan and Ronen showed in [15] that any truthful non-optimal VCG-based mechanism for combinatorial auctions suffers from abnormal behaviour, where agents may actually find it in their best interests to lie about their costs. They also showed that for many natural cost minimisation problems, any truthful VCG-based mechanism is either optimal or produces results that are arbitrarily far from the optimal. This means that such mechanisms seeking to approximate optimal solutions using polynomial time algorithms produce results that are not reliable.

3 The Assignment Problem

The Assignment Problem is otherwise known as the (Minimum Cost/Maximum Weight) Bipartite Matching Problem, in which the objective is to find the matching with the greatest total weight, or alternatively with the lowest total cost. It has a diverse range of applications including aircraft pilot rostering and dormitory room assignment.

Consider a complete bipartite graph, $G(V_1, V_2, E)$, that consists of a set of graph vertices decomposed into two disjoint sets such that no two graph vertices in the same set are adjacent. The edges $(u, v) \in E$ where $u \in V_1$ and $v \in V_2$ each has weight $w(e_i)$. In our case the set V_1 represents the Grid Service Providers and the set V_2 represents the tasks to be scheduled, and the edges represent each providers respective bid for a given task. The goal is to find a matching with minimal overall cost.

3.1 Bipartite Matching Preliminaries

A graph G is bipartite if the vertex set $V(G)$ can be partitioned into two sets X and Y in such a manner that no two vertices in the same set are adjacent. See Figure 1 for an example of a bipartite graph.

A *matching* in a graph G occurs if no two edges have a common end vertex. A matching with the largest possible number of edges is called a *maximum matching*. If M is considered to be a matching of a graph G, a vertex v, is said to be *saturated* or *covered* by M, if any edge of M is incident with v. A vertex that is *unsaturated* is also called *free*. A path or cycle is *alternating*, relative to M, if its edges are alternately in $E(G) \setminus M$ and M, where $E(G)$ denotes the set of edges in G. A path is an *augmenting path* if it is an alternating path with free origin and terminus. A graph H is a *subgraph* of G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ denoted by $H \subseteq G$. H is a spanning subgraph of G when $V(G) = V(H)$.

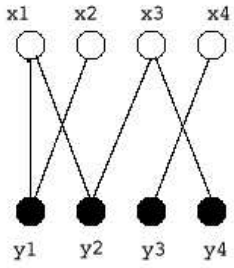


Figure 1: An example of a bipartite graph.

The following property of matchings illustrates the importance of augmenting paths[17].

Property 1 *Let M be a matching of G and P is an augmenting path relative to M. The set difference of M and P, $M \triangle P$, is also a matching for G and $|M \triangle P| = |M| + 1$*

The exploitation of this property is integral to most maximum matching algorithms because it permits them to incrementally solve assignment problems.

3.2 Operations Research Algorithms

The *Hungarian Method* is an algorithm that finds a maximum cardinality match in a graph. In our scenario this would equate to solving a matching problem in which all the edges have the same cost. The Kuhn-Munkres algorithm is an extension of the Hungarian Method in which an optimal assignment to a problem with weighted edges is solved, and this algorithm is often referred to as the Hungarian Method itself [11] [16]. It is no longer the state-of-the-art OR algorithm for solving the Assignment Problem but it does offer a far easier implementation and an incremental architecture for finding solutions to slightly differing problems. The Hungarian Method has $O(mn^2)$ time complexity, therefore solving m problems is $O(m^2n^2)$.

Edmonds and Karp [3] also developed an algorithm to find the minimum cost maximum matching in G. Dijkstra's algorithm [2] is used to determine the adjustment of the dual variable, so that for every run of Dijkstra's algorithm there is one adjustment of the dual variable. A least one augmenting path in the Equality Subgraph is also guaranteed. The time complexity for this algorithm is $O(mn + n^2 \log n)$ using a heap implementation developed by Fredman and Tarjan[7].

In our case we are only considering the Hungarian Method for now, because the algorithm incrementally solves the problem thereby lending itself towards solving related problems.

3.3 Hungarian Method

The Hungarian Method 3.3 is an efficient algorithm for finding the optimal assignment in $O(mn^2)$ time [12]. It is a primal-dual algorithm in which the primal program finds a maximal matching of minimal weight whilst the dual program finds positive labels associated with vertices such that $l_{v_1} + l_{v_2} \leq weight(v_1v_2)$. Control is alternatively swapped between the primal and dual programs. The initial feasible labeling initially sets all $v \in V_2 = 0$, see Figure 2.

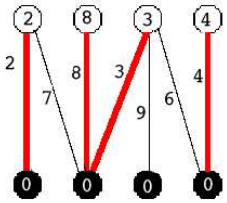


Figure 2: An example of an initial feasible labeling.

Let $f : X \cup Y \longrightarrow \mathbb{R}$ be a function with the property that

$$f(u) + f(v) \geqslant w(uv) \quad \text{for each } u \in V_1, v \in V_2.$$

f is therefore a *feasible* vertex labeling, and $f(u)$ is the *label* of the vertex *u*. A feasible labeling is always achievable, irrespective of the weight function w. The initial labeling, $f_0(x)$ is as follows:

$$f_0(x) = \begin{cases} \max\{w(xy) : y \in N(x)\}, & \text{if } x \in V_1 \\ 0, & \text{otherwise} \end{cases}$$

For a given feasible vertex function f let E_f be the subset of edges uv for which $f(y) + f(v) = w(uv)$. The *Equality Subgraph* of f, G_f, is then the spanning subgraph of G with edge set E_f. Theorem 1 describes the relationship between the graph G_f and optimal matchings of G.

Theorem 1 *Let f be a feasible vertex labeling. If G_f contains a perfect matching M, then M is an optimal matching for G. [17]*

Proof Suppose that G_f has a perfect matching M. Then since G_f is a spanning subgraph of G, M is also a perfect matching for G. Also, since each vertex of G is covered once and once only by M, and G_f is the equality subgraph of f, $w(M) = \sum_{e \in M} w(e) = \sum_{x \in V(G)} f(x)$. But if N is any perfect matching in G then $w(N) = \sum_{e \in M} w(e) \leq \sum_{x \in V(G)} f(x)$ [17]

Papadimitriou and Steglitz [16] proposed a data structure for the Hungarian method that reduces the time complexity to $O(mn^2)$. They kept a measure of the minimal *slack* on a set of edges that only requires a single update as opposed to maintaining the slack for all edges, where the slack of an edge is defined as $slack(x, y) = l(x) + l(y) - w(x, y)$.

3.4 Example

The following example, taken from [8], describes in simple terms how a bipartite matching problem is solved sequentially using the Hungarian Method. Consider the problem of five providers bidding for five tasks. The VCG mechanism requires that each provider declare its costs for all tasks. These costs correspond to the entries c_{ij} in Table T1, where c_{ij} is providers j's cost for performing task i. It is easy to see that if the cost table is transformed by adding or subtracting any amount from all the entries in any row or column, the choice of optimal assignment of tasks to providers still remains the same. For example if we added 5 to row 2, the total cost of the optimal permutation increases by 5, but it still remains the optimal solution. This property is used to try and increase the number of zeros in the table until eventually there are m independent zeros (none sharing a column or row), at which point the optimal solution is found.

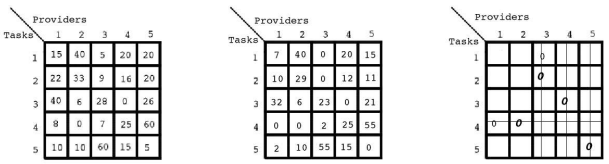
Algorithm 1: Hungarian-Method [12]

input : Graph G, and feasible labeling f
output : Max matching of G, Equality Subgraph G_f, Feasible labeling l

begin
  $G_f \leftarrow$ Equality Subgraph induced by f
  $M \leftarrow$ Any matching in G_f
  if *M is perfect* **then**
    return M,G_f,l
  else
    Let $x \in X$ be an unmatched vertex **while** $N(S) \neq T$ **do**
      if *A new matching M' is found* **then**
        Grow the alternating tree T, $M \leftarrow M'$
        if *M is perfect* **then**
          return M
    for $x \in X, y \notin T$ **do**
      $\alpha_l \leftarrow \min w(x, y) - l(x) - l(y)$
    for $v \in X \cup Y$ **do**

$$\bar{l}(v) \leftarrow \begin{cases} l(v) - \alpha_l, & \text{if } v \in S \\ l(v) + \alpha_l, & \text{if } v \in T \\ l(v), & \text{otherwise} \end{cases}$$

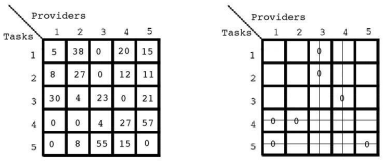
    return Hungarian-Method(G,$\bar{l}$)
end



T1 T2 T3

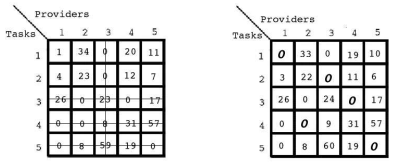
The first stage of the Hungarian Method ensures that there is at least one zero on each row and column. In this example 8,5 and 5 are subtracted from columns 1, 3 and 5 respectively, and then 4 is subtracted from row 2. The new table of costs is presented in T2. We inspect T2, looking for the largest set of independent zeros, which equates to looking for the maximum matching in the Equality Subgraph G_f, consisting of the arcs with *reduced* costs of 0. The maximum matching we can achieve has cardinality of 4. Take for example the four independent zeros in $c_{[2,3]}$, $c_{[3,4]}$, $c_{[4,1]}$, $c_{[5,5]}$, that are highlighted in bold in T3. The minimum cut in G_f also indicates a

minimal cover of zeros of the table T3 (columns 3,4,5 and row 4). We require five independent zeros therefore further manipulation of the table is required to produce a further zero. We therefore proceed to look for the smallest element not covered by a line in T3. By inspecting T2 we can see this is $c_{[5,1]}$ corresponding to the value 2. We can then subtract 2 from all entries in the table and then add 2 to all entries in the rows and columns marked by the minimal covering. After this we then end up with table T4 in which there is an additional zero.



T4 T5

A new minimal cover of zeros is calculated and the result of this covering is visible in table T5. The cardinality of this covering is still only 4 however. The process is repeated and the smallest uncovered value is 4, $c_{[3,2]}$. Subtracting 4 from all values and then adding to the values under each of the lines in T5 we arrive at the table in T6, whose minimal covering of cardinality 4 is also shown.



T6 T7

After this however the value 1, in $c_{[1,1]}$ can be made zero and the resultant minimal covering has cardinality 5, therefore a solution has been found. The bold zeros in T7 correspond to the solution, and the total cost can be calculated from T1.

- Provider 1 → Task 1
- Provider 2 → Task 4
- Provider 3 → Task 2
- Provider 4 → Task 3
- Provider 5 → Task 5

The minimum overall cost in this example is therefore $15 + 0 + 9 + 0 + 5 = 29$. [4]

4 Our Approach: m-Best-Robust Solutions

In our case we are looking for the m-Best-Robust matchings or in other words, we are looking for the best matchings $M_i = M(G \setminus x_i)$ when $i = 1, \ldots, m$. Recall that optimal solutions are required for each Grid Service Provider absent in turn, in order to determine payments to each GSP. We use the term *robust* because each solution is independent of at least one of the available providers $x \in X$, therefore it could alternatively be viewed as being a range of robust solutions to choose from if X is viewed as a set of machines in a Job Shop Problem and one of the machines breaks down, for example.

[4] Note however that if we were to determine prices for the providers we would remove one column from the matrix and resolve. But this produces a matrix in which $m > n$, therefore it is no longer a matching problem. We must therefore introduce a rule that $m < n$ if Vickrey-prices are to be determined successfully.

The Hungarian Method is incremental in its solving of the problem, and we can take advantage of its ability to recompute the optimal matching in a single iteration when a single edge is discarded at a node of the search tree. After computing the solution to the first minimum cost matching problem with a single vertex (the first in this case) absent, then we can effectively discard all the edges incident with the first vertex in our problem already computed and replace them with the costs of the vertex that were missing from the previous problem.

Algorithm 2: m-Best-Robust Solutions

input : Bipartite Graph $G = (X, Y, E)$ with $|X| = m$ and $|Y| = n$ under the cost c_{ij}, with initial feasible labeling f_0

output : Set of solutions S, where $s_i \in S$ is the optimal solution for $G \setminus x_i = (X \setminus x_i, Y, E)$

begin
  Get optimal overall soln. s_o =Hungarian-Method(G,f_0)
  s_1,G_{f_1},f_1=Hungarian-Method($G \setminus x_1$,f_0)
  $S \leftarrow S \cup s_1$
  for $i = 2 \ldots m$ **do**
    s_i,G_{f_i},f_i =Hungarian-Method($G_{f_{i-1}} \setminus x_{i-1} \cup x_i$,$f_{i-1}$)
    $S \leftarrow S \cup s_i$
  Return S
end

It has been shown that by modifying one value in the cost matrix, an optimal solution can be calculated in $< O(mn^2)$, through a simple augmenting path step. If m costs in a single row can be changed and the solution recomputed in $O(n^2)$ then we can say that the overall complexity of solving m bipartite matching problems $< O(m^2n^2)$.

Theorem 2 *Pricing for m providers bidding for n tasks, where $m < n$, in the Vickrey-Clarkes-Groves mechanism can be computed in $O(mn^2)$ time complexity using the Hungarian Method.*

Proof Each stage of the Hungarian Method augments the matching by one edge therefore there is a maximum of $m + 1$ stages. To analyse each stage, let us first examine the complexities for the search and dual variable modifications separately. Search requires $O(n^2)$ operations, because each vertex is examined sequentially and never re-examined[16]. The removal of a vertex from X requires $O(n)$ operations. For dual variable modifications however, we either have a new vertex that is labeled or we have an augmentation. We can therefore have a maximum of n dual variable modifications at each stage. Each modification takes $O(n)$ time so each stage requires $O(n^2)$ operations. Solving m problems from scratch incurs $m(m + 1)$ stages in total, therefore the overall time complexity becomes $O((m^2 + m)(n^2)) = O(m^2n^2)$. In our algorithm however we use G_f from the previous problem to our advantage. One of the vertices in X is replaced by another and the subsequent problem therefore only requires one further stage to modify the dual variables and search for an augmenting path. The time complexity to solve m subproblems becomes $O(mn^2)$. [5]

[5] We only need to solve m subproblems, even though there are n providers because only m providers were assigned tasks in the original solution so each one of these providers is omitted from each subproblem in turn.

5 Experimental Results

We have used an implementation of the Hungarian Method from Donald Knuth's Stanford GraphBase [11] that uses data structures proposed in [16] and gives a time complexity of $O(mn^2)$. When calculating prices we have $m+1$ problems to compute so we'd naturally expect the running time to become $O(m^2n^2)$, but the incremental nature of the Hungarian Method allows us to change one row of the matrix and recompute a solution in $O(n^2)$.

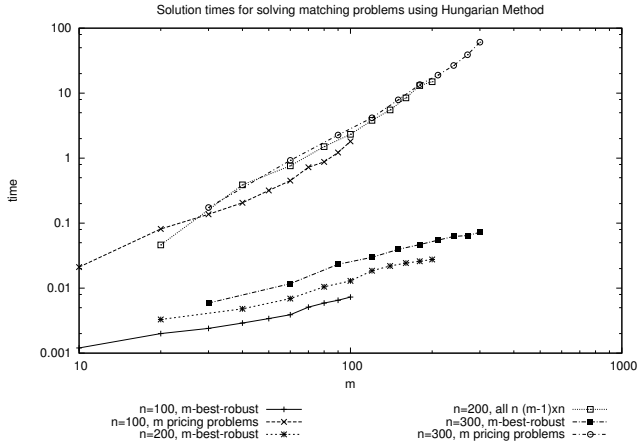


Figure 3: Comparative Results for m-best-robust algorithm versus solving all pricing problems from scratch on a logarithmic scale.

Our main interest in this graph, see Figure 3, are the three lines representing the results for the m-best-robust algorithm. The slopes of all three lines ≈ 1. This empirical evidence confirms the findings of Theorem 2, that its time complexity is $O(mn^2)$. Also notice that the naive approach of solving all pricing problems from scratch has slope ≈ 2. This confirms its complexity is $O(m^2n^2)$.

These result answer an open question of Hershberger and Suri [9] of whether a lower bound for the auctioning of m tasks amongst n providers where $m < n$, can be reduced to less than that of computing $n + 1$ optimisation problems.

6 Agenda: Constraint Programming and Truthfulness

Caseau and Laburthe [1] identified that CP is not competitive with specialised OR techniques such as the Hungarian Method for solving matching problems. CP becomes a far more plausible offering however, when searching for all optimal solutions or those within a given distance of the optimum. Recall that in our case, we are interested in solving the matching problem and then resolving the problem again minus one resource for all n resources present. The constraint algorithm performs less computations per node of the search tree, therefore it can search large trees more quickly, whereas the Hungarian Method searches narrower trees but requires more time at each node.

6.1 Additional Constraints

The Assignment Problem has a property that adding any constant value to any row of the cost matrix doesn't change the optimal solution. This property allows us to add constraints limiting upper and lower bounds for each subproblem and also permits a heuristic for ordering of subproblems so that successive subproblems are likely to have tight upper and lower bounds from previously solved problems.

Additional constraints such as these and others are easily added to the constraint programming implementation, whereas it is much more difficult in a pure OR algorithm. In [4], reduced costs of separate assignment subproblems have been used for propagation in a CLP framework. They adapted variable fixing to a CLP framework, and used it successfully for pruning domains. They created an `allDiffCost` constraint for ILOG Solver that uses OR techniques to achieve pruning with cost-based reasoning. This global constraint embeds a Minimum Sum Linear Assignment Problem at each node of the search tree, and with the resulting optimal solution facilitates constraint propagation. Caseau and Laburthe [1] also created a similar global constraint, `MinWeightAllDifferent`, that is propagated using the Hungarian Method.

6.2 Problem Ordering Heuristics

The following algorithm populates three matrices, o_{ij}, m_{ij} and d_{ij}. Values m_{ij} are the entries in a $n \times n$ matrix representing the sum of differences between bids of providers i and j when i's bids are offset by a minimal amount o_{ij}, so that provider i, p_i, is always more expensive than p_j. Values d_{ij} represent the upper bounds on $soln(P \setminus p_i)$ for Problem $P \setminus p_j$.

Algorithm 3: Problem Ordering Heuristics

input : Costs c_{ij} in a $n \times n$ matrix represent p_i's bid for task j.
output : o_{ij},m_{ij} and d_{ij}
begin
  for $i = 1 \ldots n$ **do**
    for $j = 1 \ldots n$ **do**
      if $i = j$ **then**
        $o_{ij} = 0, m_{ij} = 0, d_{ij} = 0$
      else
        $o_{ij} = \min_{k=1\ldots n}(c_{ik} - c_{jk})$
        $m_{ij} = \sum_{k=1\ldots n}(c_{ik} - c_{jk} - o_{ij})$
        $d_{ij} = \max_{k=1\ldots n}(c_{ik} - c_{jk} - o_{ij})$
end

Take an example where provider p_1s' bids are all 5 units less than p_2, therefore $o_{1,2} = 5$, $m_{1,2} = 0$ and $d_{1,2} = 0$. We can safely say that the optimal solution for the problem with p_2 missing, $soln(P \setminus p_2)$, is exactly 5 units less than $soln(P \setminus p_1)$ and providers p_1 and p_2 perform the same task. However if two of p_2s' bids are 6 greater, $o_{1,2} = 6$, $m_{1,2} = 2$ and $d_{1,2} = 1$, and we can say the upper bound for $P \setminus p_1$ is the solution of $P \setminus p_2 + 6$. Alternatively, if $soln(P \setminus p_1)$ is solved first, the upper bound on $soln(P \setminus p_2)$ is the $soln(P \setminus p_1) - 6$. Lower bounds can be similarly computed, and as the search progresses, these bounds become tighter. Also note that if $\min_{j=1..n}(c_{1j} - c_{2j}) = (c_{1t} - c_{2t})$ then $soln(P \setminus p_2) = soln(P \setminus p_1) \setminus t$ and $soln(P \setminus p_2, t) = p_1$.

During search, the values in o_{ij}, m_{ij} and d_{ij} can be reduced as the tasks are removed from possible domains. This enables further propagation and inference, hence it is well suited to CP techniques. Minimal values from d_{ij} can be used as a heuristic to select subsequent problems. We call this the min-upper-bound heuristic, whereas selecting minimal values from m_{ij} is the most-similar-bid-profile heuristic. We aim to investigate the potential benefits of these Vickrey pricing problem-ordering heuristics in future work,

6.3 Benefits of CP

One major benefit of constraint programming over operations research algorithms is its simplicity and straightforward implementation. Its compact program structure also permits more flexibility, when different input constraints are required. Constraints offer easier problem maintenance and in the case of $m \geq n$, the specialised OR matching algorithms no longer apply whereas CP can still find solutions. The OR algorithms are difficult to implement efficiently and often rely on specialised data structures for good performance, [16] describes a specialised data structure for the Hungarian Method for example.

Solutions to all the pricing sub-problems must be optimal, thus precluding non-deterministic search algorithms. Simulated annealing and genetic-based algorithms are both hill-climbing algorithms with random components that allow escape from local maxima, but aren't guaranteed to find the optimal solution. A systematic search is necessary in order to guarantee optimality, [1] and [4] provide evidence that constraints offer a promising alternative when more than one solution to a matching problem is required.

7 Related Work

Hung et al. describe in [10] an algorithm for finding what they term as *"the most vital edges of matching in a bipartite graph"*. An edge is called a most vital edge (MVE), with respect to a weighted matching, if its removal from the graph results in the largest decrease in the total weight of the maximum matching. Or alternatively, the greatest increase in cost in a minimal cost matching. The MVE algorithm finds a maximal weighted matching with each edge removed in turn, in order to find the most important edge. This is precisely what is required to calculate Vickrey prices for each provider.

Their algorithm constructs an auxiliary digraph N from an original graph G, based on the optimal matching M^*. The shortest paths in N are then found from u_i to v_i, where $i = 1 \ldots n$. For each edge $e = (u_i, v_i) \in M^*$, calculate $w(M^*) - w(M_e) = w_e + min\{0, w[P_{u_i,v_i}]$. Choose the edges that have the largest value of $w(M^*) - w(M_e) = 0$ for all $e \in M^*$, hence all edges in G are the most vital edges. Dijkstra's algorithm cannot be used because N may contain edges of negative weight, so Floyd's algorithm can be applied to find the shortest paths in $O(mn^2)$. This result provides the same time complexity as our algorithm using a different technique. One of the advantages of their technique is that any bipartite weighted matching algorithm can be used to solve the original problem, but an advantage of our method is its simplicity and straghtforward implementation.

8 Conclusions and Future Work

Ensuring truthfulness amongst self-interested agents who are bidding against one another in an auction is computationally expensive. The Vickrey-Clarke-Groves mechanism guarantees that each of the agent's dominant strategy is to tell the truth, but it requires solving $n + 1$ optimisation problems for n agents. This paper presented an algorithm for computing prices for all agents in the same asymptotic time complexity as solving just one problem in the case of n agents bidding for m tasks where $m < n$, thus solving an open problem presented by Nisan and Ronen [14] for the specific case of task scheduling. This problem was also identified by Hershberger and Suri [9].

CP algorithms can work in harmony with OR algorithms to minimise the punitive computational burden of ensuring truthfulness. Using propagation and inference techniques, it can aid the search when the conditions for solutions are complicated by additional constraints.

We aim to examine which problem ordering heuristics work best with realistic data and which constraints are most effective in reducing the search space. We also plan to investigate Branching CSPs [6], because if we are expecting important tasks we need to ensure spare capacity to cope with uncertain future demands. We will be conducting an empirical study of the benefits of using CP for task scheduling within a pricing mechanism framework. Furthermore, we will generalise the study so that m tasks are auctioned amongst n providers, where $m \geq n$. We also aim to show that constraint technology can make truthfulness more computationally feasible when allied with OR algorithms in our future work.

ACKNOWLEDGEMENTS

We wish to thank Subhash Suri for his comments on an earlier draft of this paper.

REFERENCES

[1] Yves Caseau and Francois Laburthe, 'Solving various weighted matching problems with constraints', in *Principles and Practice of Constraint Programming*, pp. 17–31, (1997).

[2] Edsger Wybe Dijkstra, 'A note on two problems in connexion with graphs', *Numerische Mathematik*, **1**, 269–271, (1959).

[3] Jack Edmonds and Richard M. Karp, 'Theoretical improvements in algorithmic efficiency for network flow problems', *Journal of the Association of Computing Machinery*, **19**(2), 248–264, (April 1972).

[4] Filippo Focacci, Andrea Lodi, and Michela Milano, 'Integration of CP and OR methods for matching problems', in *CP-AI-OR'99 Workshop on Integration of AI and OR techniques in Constraint Programming for Combinatorial Optimization Problems*, (1999).

[5] Ian Foster, 'The anatomy of the grid: Enabling scalable virtual organizations', *Lecture Notes in Computer Science*, **2150**, (2001).

[6] David Fowler and Ken Brown, 'Branching constraint satisfaction problems for solutions robust under likely changes', in *Proceedings of CP'2000*, ed., Rina Dechter, pp. 500–504. Springer Verlag, (2000).

[7] Michael L. Fredman and Robert Endre Tarjan, 'Network flow and testing graph connectivity', in *25th FOCS*, pp. 338–346, (1984).

[8] Michel Gondran and Michel Minoux, *Graphs et Algorithmes*, Wiley, English edn., 1984. Translated by Steven Vajda.

[9] John Hershberger and Subhash Suri, 'Vickrey pricing in network routing: Fast payment computation', in *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*, pp. 252–259, (2001).

[10] Chun-Nan Hung, Lih-Hsing Hsu, and Ting-Yi Sung, 'The most vital edges in a bipartite graph', *Networks*, **23**(4), 309–313, (July 1993).

[11] Donald E. Knuth, *The Stanford GraphBase*, Addison Wesley, 1993.

[12] Harold W. Kuhn, 'The Hungarian Method for the Assignment Problem', *Naval Research Quarterly*, **2**, 83, (1955).

[13] Andreu Mas-Collel, Micheal D. Whinston, and Jerry R. Green, *Microeconomic Theory*, Oxford University Press, 1995.

[14] Noam Nisan and Amir Ronen, 'Algorithmic mechanism design', in *Proceedings of the 31st Annual ACM Symposium on Theory Computing*, (1999).

[15] Noam Nisan and Amir Ronen, 'Computationally feasible VCG mechanisms', in *ACM Conference on Electronic Commerce*, pp. 242–252, (2000).

[16] Christos H. Papdimitriou and Kenneth Steiglitz, *Combinatorial Optimisation*, Prenitce-Hall, 1982.

[17] Armen S. Ratian, Tristan M.J. Denley, and Roland Haggkvist, *Bipartite Graphs and their Applications*, Cambridge University Press, 1998.

[18] W. Vickrey, 'Counterspeculation, auctions, and competitive sealed tenders', *Journal of Finance*, 8–37, (1961).

RecTree Centroid: An Accurate, Scalable Collaborative Recommender

Jerome Kelleher and **Derek Bridge**[1]

Abstract. We present a Collaborative Recommender that uses a user-based model to predict user ratings for specified items. The model comprises summary rating information derived from a hierarchical clustering of the users. We compare our algorithm with several others. We show that its accuracy is good and its coverage is maximal. We also show that the algorithm is very efficient: predictions can be made in time that grows independently of the number of ratings and items and only logarithmically in the number of users.

1 Introduction

Recommender Systems advise their users about which items (products, services or information) to consume. In *Content-Based Recommenders*, users articulate their requirements and the system matches them against item descriptions. In *Collaborative Recommenders*, which are the subject of this paper, no descriptions of requirements or items are needed; the system bases its recommendations on information about the preferences of related users. Standardly, if there are n users, $U = \{u : 1 \dots n\}$, and m items, $I = \{i : 1 \dots m\}$, preferences are represented using a $n \times m$ matrix r of ratings. User u may explicitly supply his/her rating of item i, $r_{u,i}$, or the system may obtain an implicit rating from observing user actions. A rating is typically represented by a Boolean or by a value from a numeric scale. Note that it is possible and common that $r_{u,i} = \perp$, signalling that the user has not yet rated that item. The user who is interacting with the recommender, $u_a \in U$, is called the *active user*. In the task of *prediction*, the recommender system uses r to compute a predicted rating $p_{u_a,i}$ for active user u_a and an item i for which $r_{u_a,i} = \perp$.

Collaborative Recommenders differ in the amount of work they do *off-line*, in advance of making predictions, and the amount of work they do *on-line*, when making predictions. Off-line, the system builds a *model* from r: the ratings data is mined for association rules, Bayesian networks or other structures that can capture regularities in the data. The model might be characterised as being either *user-based*, where the model captures relationships between users who have similar ratings, or *item-based*, where the model captures relationships between items that have been rated similarly. Then, on-line, predictions are made from the model.

In *model-based* systems, *most* of the work is done off-line; then, on-line, predictions are made using the model to the exclusion of the ratings matrix r. For example, Breese et al. [1] describe a system that learns a Bayesian network with a node for each item, arcs for item dependencies and decision trees at each node to encode the conditional probabilities. For on-line predictions, the items that the active user has rated are used as evidence variables, and the item-to-be-predicted, i, is the query variable.

In *memory-based* systems, little, if any, work is done off-line; predictions are made directly from r. An example of a purely memory-based system is the Exhaustive Recommender we describe in Section 3. It uses *neigbourhood-based* methods. Using the data in r, the active user's neighbourhood of recommender partners, i.e. the set of users who have rated i whose ratings are most highly correlated with the active user's ratings, is obtained. Then a prediction is computed from a weighted combination of the neighbours' ratings. Both of these computations happen on-line.

There are many ways of *combining* model-based and memory-based approaches. In Section 3, for example, we describe the possibility of an Exhaustive Recommender that pre-computes a matrix of all user-user correlations. This gives it a user-based model that speeds up the on-line process of finding neighbours. The approach still qualifies as memory-based because most prediction effort happens on-line and because, once the neighbours have been found, predictions are still computed directly from r.

In this paper, we propose a new approach that is purely model-based. A user-based model is built using clustering techniques; predictions are made using summary information about the clusters.

In Section 3, we describe a memory-based recommender. We show how it can be improved in Section 4. Section 5 explains the clustering algorithm we use in this work. Then in Section 6, we describe our new model-based algorithm, which uses summary information about the clusters for efficient, accurate predictions. Section 7 compares our results with those from related work. Before any of that, in Section 2 we explain how we have evaluated the different algorithms.

2 Evaluation

We compare the performances of the different algorithms that are described in this paper using two datasets: the MovieLens dataset (`www.grouplens.org`) and the PTV dataset[2]. Some of the characteristics of these datasets are reported in Table 1. The MovieLens dataset has been cleaned up to include only users who have rated at

Table 1. Dataset Characteristics

Dataset	No. of users	No. of items	No. of ratings
MovieLens	943	1682	100000
PTV	2341	8164	60000

[1] Department of Computer Science, University College Cork. Email: `jt.kelleher|d.bridge@cs.ucc.ie`

[2] We are grateful to ChangingWorlds for supplying us with this dataset. The original dataset contains 60666 ratings but we removed 666 at random to give a round number of ratings.

least 20 movies; the PTV dataset has not. Both are sparse, but the PTV dataset is by far the sparser.

We have converted all ratings to z-scores in advance of using them in the experiments described in this paper. This conversion gives a very small improvement in the accuracy of the Exhaustive Recommender (Section 3) and significant improvements for the systems described in Sections 4 and 6.

In each experiment, the dataset is split into two disjoint sets, the training set (80%) and the test set (20%). All results are subject to five-fold cross validation, each time using a different 80/20 split. To model the on-going use of a recommender system, experiments use different total numbers of ratings. We report the effects on the accuracy, coverage and efficiency of the algorithms.

- To measure *prediction accuracy*, we use Mean Absolute Error (MAE). MAE is calculated by averaging the absolute difference between the algorithm's predicted rating and the user's actual rating (from the test set).
- *Prediction coverage* is measured by counting the number of times the algorithm fulfils a prediction request and reporting this as a percentage of the overall number of prediction requests made.
- To give an implementation-independent measure of *prediction efficiency*, we count Total Prediction Operations (TPO). We explain exactly which operations are counted in Sections 3 and 6.

If a system receives widespread use, the numbers of users, items and ratings are likely to be continually growing and to become extremely large. Hence, we are particularly interested in the scalability of Collaborative Recommenders. It may be that improvements in, for example, accuracy are not justified if they result in substantially longer response times.

In this vein, we will also report the worst-case time complexity and space complexity (which will include the size of the model, if there is one) for the on-line components of the algorithms. And, we will also discuss how easy it is to introduce new users, new items and new ratings. Some model-based approaches have particular problems with new data. There may be no efficient, incremental way of revising the model to take account of the new data. During the period between the arrival of the new data and the periodic re-generation of the model, the value of the system to its users may be diminished.

3 Exhaustive Recommender

The Exhaustive Recommender is a memory-based system whose performance is used as a baseline in comparisons with the other algorithms. We described it briefly in the previous section. Here we present some of the details.

In [5], the Exhaustive algorithm was subjected to an extensive empirical investigation using the MovieLens dataset. The decisions that we make below about the details of the algorithm (the choice of Pearson correlation as the similarity formula, the weighting factor of 50, the choice of 20 as the size of the neighbourhood and the choice of prediction formula) are based on the best results reported in [5]. We use the exact same settings in all our other experiments using different algorithms. Because of the time it would take, we have not verified that these are the best settings for these other algorithms. These other algorithms generally out-perform the Exhaustive algorithm. So the effect of not finding the best settings is to understate the extent to which they out-perform the Exhaustive algorithm. More questionably, we use the same settings in our experiments on the PTV dataset.

The Exhaustive algorithm works as follows:

- The similarity $w_{u_a,u}$ between the active user u_a and each other user $u \neq u_a$ *who has rated* i is computed using Pearson Correlation [5].

$$w_{u_a,u} \hat{=} \frac{\sum_{i=1}^{m}(r_{u_a,i} - \bar{r}_{u_a})(r_{u,i} - \bar{r}_u)}{\sigma_{u_a}\sigma_u} \times \frac{s}{50} \quad (1)$$

  $\bar{r}$ denotes a mean value and σ denotes a standard deviation, and these are computed on co-rated items only. If a user has given the same rating to all the co-rated items, his/her standard deviation is zero and $w_{u_a,u}$ is then defined to be zero (J.Herlocker, pers. comm. 2002). Following [5], the Pearson Correlation coefficient is weighted by a factor of $\frac{s}{50}$ where s is the number of co-rated items. This decreases the similarity between users who have fewer than 50 co-rated items (who, even if their ratings are very similar, are likely to be poor predictors).
- After computing the similarity between u_a and each other user u who has rated i, the 20 nearest neighbours are selected, i.e. the 20 for whom $w_{u_a,u}$ is highest.
- The predicted rating $p_{u_a,i}$ is computed from the neighbours' ratings of i as follows:

$$p_{u_a,i} \hat{=} \bar{r}_{u_a} + \frac{\sum_{u=1}^{k}(r_{u,i} - \bar{r}_u)w_{u_a,u}}{\sum_{u=1}^{k} w_{u_a,u}} \quad (2)$$

  where k is the number of neighbours (in our case, 20). This is essentially a weighted average of the neighbours' ratings for item i (weighted by their similarities). If $p_{u_a,i}$ goes out of the range of legal ratings, it is rounded to the nearest end-point of the range (J.Herlocker, pers. comm. 2002).

In fact, we improve the prediction efficiency of this algorithm by doing some off-line computation. We pre-compute a matrix of user-user similarities (Equation 1). This means that, on-line, we need only find the 20 nearest neighbours who have rated i and compute their prediction for i using Equation 2.

The accuracy, coverage and efficiency results for running this algorithm (and our other two algorithms) on the two datasets are shown in Figures 1– 6. Figures 1 and 4 show that accuracy is good when the dataset becomes less sparse. But Figures 2 and 5 show that coverage is not maximal. The real problem with this algorithm, however, shown by Figures 3 and 6, is that it does not scale well. In these two plots, we are counting the following operations: we count as one operation the check to see whether the user has rated i; then we count all the comparisons needed to find the 20 nearest neighbours; and then we count each rating that we aggregate when making the final prediction.

The algorithm must compare the active user with all other users, of which there are n, checking each to see whether they have rated i. Finding the k nearest neighbours will require nk comparisons in the worst case. The prediction formula then requires k steps, to compute the average rating for i over the k neighbours. We are assuming here that Equations 1 and 2 are reformulated to allow incremental computation, so means and standard deviations can be computed without multiple passes through the co-rated items. The total cost is therefore $n(1+k)+k$. Taking k to be constant, the time complexity is $O(n)$. (Without the user-user matrix, it is $O(nm)$.)

The space complexity is the cost of storing matrix r, which is $O(nm)$, plus the cost of storing the user-user matrix, which is $O(n^2)$.

Irrespective of whether a user-user matrix is pre-computed or not, this algorithm easily incorporates new users, new items and new ratings. We simply update r and, if applicable, we make incremental updates to the similarities in the user-user matrix.

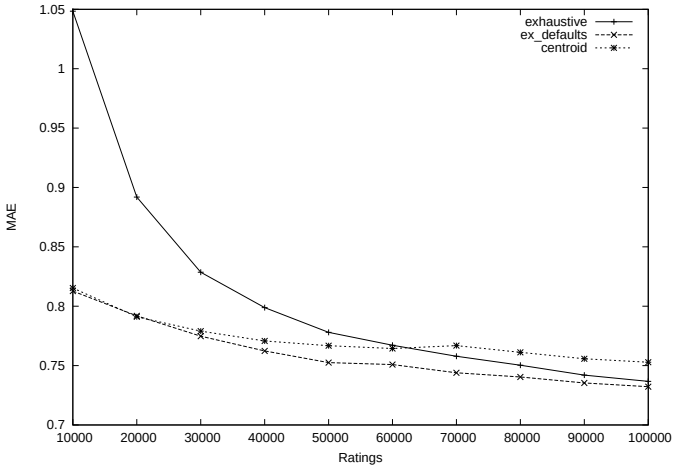


Figure 1. MovieLens Error Results

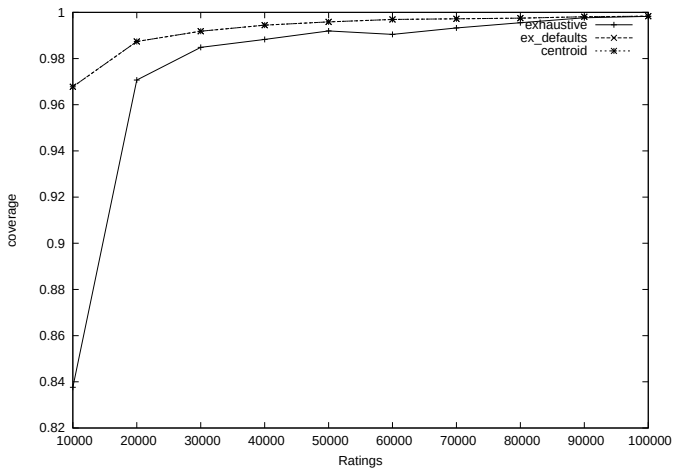


Figure 2. MovieLens Coverage Results

4 Exhaustive with Defaults Recommender

There are occasions when the Exhaustive Recommender cannot make a prediction. We can increase coverage by providing a strategy for making default predictions in such cases. A simple strategy is, when no prediction can be made, predict i's average rating in the dataset.

But, in fact, we have found that both coverage *and* accuracy can be increased if the default strategy is invoked more often. It is well-known that non-personalised predictions can be more accurate in cases where the recommender does not have enough ratings to provide an accurate personalised recommendation. Our Exhaustive with Defaults Recommender makes default predictions whenever the number of neighbours who have rated i falls below a minimum threshold. This, of course, includes the case where no neighbour has rated the item. In our experiments, the threshold we have found to be best is actually 20. In other words, if the algorithm fails

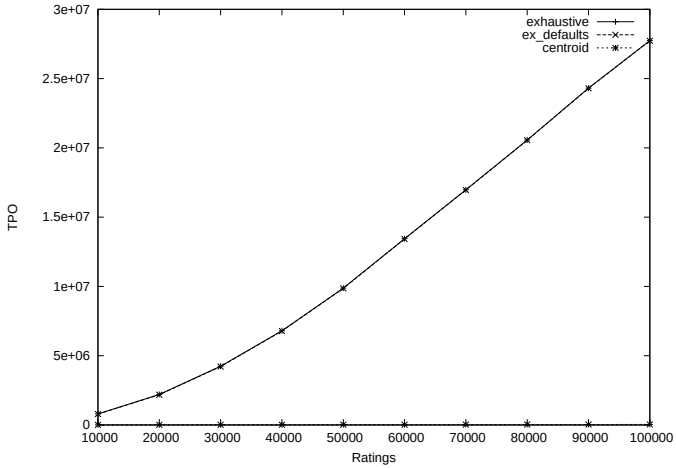


Figure 3. MovieLens Effi ciency Results

to find a full set of 20 neighbours who have rated i, it uses a default prediction.

The default prediction for i, def_i, is defined as follows:

$$def_i \hat{=} avg_i(U) \tag{3}$$

where for set of users S,

$$avg_i(S) \hat{=} \begin{cases} \frac{\sum_{\{u \in S : r_{u,i} \neq \perp\}} r_{u,i}}{|\{u \in S : r_{u,i} \neq \perp\}|} & \text{if } \exists u \in S : r_{u,i} \neq \perp \\ \perp & \text{otherwise} \end{cases} \tag{4}$$

This is simply the average rating for i over all users who have rated i.

Exhaustive with Defaults is an algorithm with *maximal* coverage. The only cases in which a prediction will still fail to be made are those where *no one* in the whole dataset has rated i.

The default predictions (averages) can be pre-computed, and are easily updated incrementally when new ratings arrive. The on-line cost of providing any default prediction is, therefore, $O(1)$. So, the overall time and space worst case complexities of Exhaustive with Defaults are no different from those of Exhaustive.

The advantage of including results for this algorithm in this paper is that this algorithm provides fairer accuracy and efficiency comparisons with our RecTree Centroid Recommender (Section 6). Both Exhaustive with Defaults and RecTree Centroid have maximal coverage. This means that accuracy and efficiency results can be compared for exactly the same sets of predictions.

We turn now to the RecTree Centroid algorithm. We first explain the way we cluster the data, and then we explain the RecTree Centroid Recommender itself.

5 The Clustering Algorithm

The clustering algorithm which we have chosen is called RecTree. It builds a binary tree of clusters; it was previously used to cluster users in [3]. RecTree recursively invokes the k-means clustering algorithm. In k-means, elements are repeatedly assigned to clusters on the basis of their similarity to the cluster centre. We use Pearson Correlation, Equation 1, to compute similarities. The centres are initially

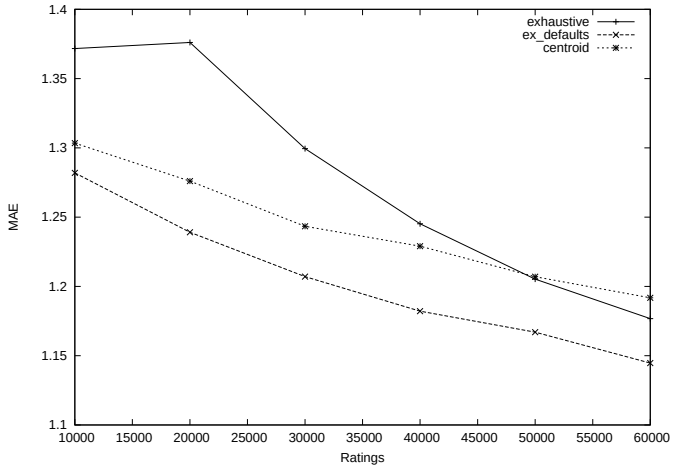


Figure 4. PTV Error Results

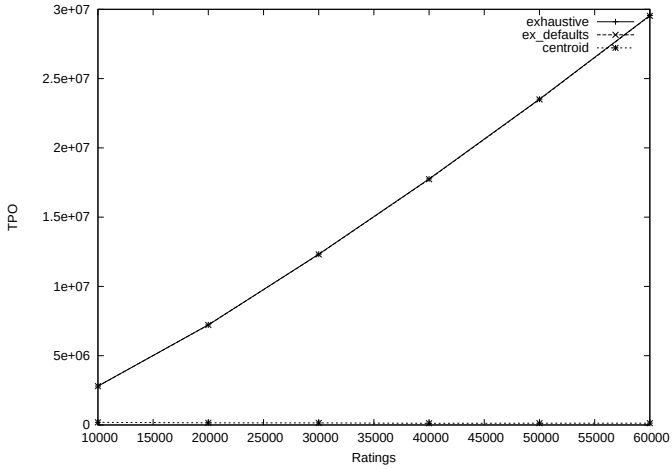


Figure 6. PTV Effi ciency Results

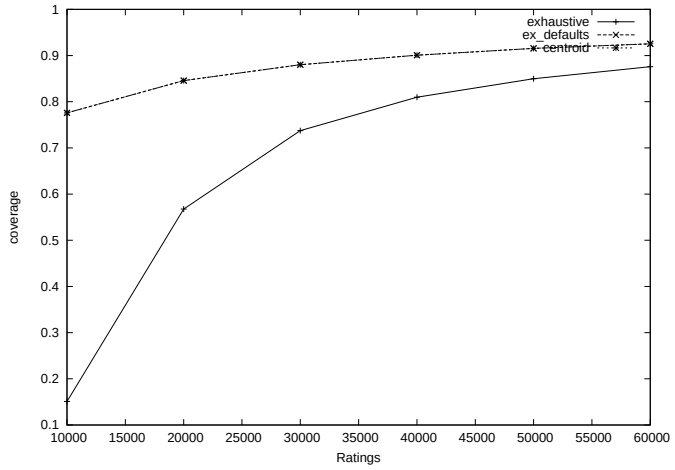


Figure 5. PTV Coverage Results

seeds, provided as parameters. As the algorithm iterates, the centres become the centroids of the existing clusters, where a centroid is a new 'dummy' element formed by averaging ratings within the cluster. Specifically, the centroid of cluster c that contains users U_c gives rating $cent_{c,i}$ to item i:

$$cent_{c,i} \hat{=} avg_i(U_c) \tag{5}$$

which uses Equation 4 to compute the average rating, but we compute this only for the users in cluster c.

We consider k-means to have converged when the size of all clusters is the same on two successive iterations. We found empirically that convergence usually takes about 15 iterations. To ensure that the algorithm always terminates, even on pathological data, we imposed a complementary stopping criterion in the form of an iteration limit of 20.

The RecTree algorithm calls k-means to split successive datasets into child clusters. It returns a binary tree of clusters, where the root represents the whole dataset. The algorithm splits the dataset if its size is greater than a pre-specified maximum leaf cluster size. It selects seeds using the 'two mutually well separated centres policy' [3]. The first seed is the element within the dataset whose distance is greatest from the dataset centroid. The second seed is the element which is most distant from the first seed.

Algorithm 1 RecTree(*parent*, *elements*) [3]

$v \leftarrow$ new node having *elements* as its contents
make v a child of *parent*
if *elements* size $>$ max. leaf cluster size and no. of internals $<$ max. no. of internals **then**
  centroid $\leftarrow$ centroid of *elements*
  seed1 $\leftarrow$ the member of *elements* with min. correlation with *centroid*
  seed2 $\leftarrow$ the member of *elements*$\backslash\{$*seed1*$\}$ with min. correlation with *seed1*
  clusters $\leftarrow$ k-means(*elements*, $\{$*seed1*, *seed2*$\}$)
  for all c in *clusters* **do**
    c-elements $\leftarrow$ elements in cluster c
    RecTree(v, *c-elements*)
  end for
end if
return v

RecTree terminates when the sizes of all leaf nodes are less than or equal to the maximum leaf cluster size and so no more subdivision is required. In some cases, however, when data is not suitably distributed, growth in the tree can be 'lopsided'. This is where a very small and a very large cluster are created at each invocation. In this case we prevent the construction algorithm from over-partitioning the data by limiting the number of internal nodes (in our case to 2000). Because of this, the algorithm makes no guarantee about the size of any leaf cluster; but if the algorithm terminates normally, leaf clusters will have at most the maximum leaf cluster size number of elements.

For each user, we store a reference to the leaf cluster to which s/he is ultimately assigned. This allows $O(1)$ access to that cluster. For each cluster (including interior clusters), we store the cluster's

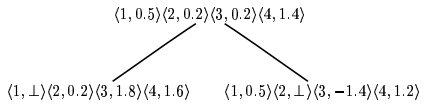


Figure 7. Extract from a RecTree

centroid (Equation 5) to give $O(1)$ access time to this information.

We can obtain some idea of the cost of walking and storing the RecTree if we make some assumptions about the shape of the tree that gets built. Let us idealise and assume that the two children of a node share equally the users of their parent, i.e. the two children are the same size, which is half their parent size. At any level of the tree, this will give clusters whose sizes differ by at most one. For n users and a maximum leaf cluster size of max, the height of the tree h is $\lceil \log_2 n - \log_2 max \rceil$, which is $O(log_2 n)$. There are $2^h + 2^{h-1} + \ldots + 1$ nodes in the tree, which, given that h is $O(\log_2 n)$, is $O(n)$. There are $\lceil n/max \rceil$ (or, equivalently, 2^h) leaf clusters.

6 RecTree Centroid Recommender

The RecTree Centroid Recommender is a model-based recommender. Predictions are obtained from the RecTree, which is used as a user-based model.

Suppose the active user u_a is a member of leaf cluster c. An initial prediction of u_a's rating for item i is the average rating of i for the users in cluster c, as given by the centroid. However, from Equations 5 and 4, we know that if no one in the cluster has rated i, then $cent_{c,i}$ will be $\perp$. In this event, the RecTree Centroid Recommender visits the parent of cluster c and returns the rating for i given by the parent's centroid (if this is not $\perp$). The algorithm climbs the RecTree until it finds a centroid whose rating for i is not $\perp$ or until it reaches the root, in which case it returns the rating contained in the centroid for the root (whether this be $\perp$ or not).

For example, suppose that the active user belongs to the right-hand leaf cluster in Figure 7. The ratings stored at this node are the average ratings from the users that belong to this cluster. In particular, for items 1, 3 and 4, the average ratings are 0.5, -1.4 and 1.2 respectively; no-one in this cluster has rated item 2. If we want to predict the active user's rating for item 4, for example, we simply return the rating stored at this node, i.e. 1.2. If we want to predict a rating for item 2, we ascend the tree until we find a node at which there is a rating for item 2 (or until we reach the root). Recall that a parent node in the RecTree represents the union of the users associated with its two child nodes. There is thus a greater likelihood that the parent's centroid will have a rating for i that is not $\perp$. In Figure 7, for example, the parent has a rating for item 2 because some of the users associated with the left-hand child have rated item 2; so the prediction is 0.2. The root node represents all users and so the root node's centroid will only have no rating for i if no one in the entire system has rated item i.

It might be argued that, while this algorithm is certainly a collaborative one (since the preferences of other users are used to construct the RecTree), it is a less *personalised* recommender: all users in the same cluster receive the same predictions for items they have not rated.

For the MovieLens dataset, we found experimentally that a maximum leaf cluster size of 300 gave the best results, and so (although it may not be best for PTV) this is what we used for both MovieLens and PTV in Figures 1– 6.

Figures 1 and 4 show that RecTree Centroid actually has lower error than the Exhaustive Recommender for the sparsest datasets and only has higher error as ratings grow beyond a certain point; it is never more accurate than Exhaustive with Defaults but it is still highly competitive. The use of z-scores makes a big difference: when we used raw ratings (not plotted), error was much higher.

Coverage (Figures 2 and 5) is maximal, the same as Exhaustive with Defaults. In both algorithms, the only circumstance in which a prediction will not be made is if no user has rated item i.

But RecTree Centroid is very efficient. In Figures 3 and 6, the plot (which counts tree node accesses) lies slightly above the x-axis. It takes constant time to access a user's leaf cluster. Then, in the worst case, the algorithm climbs the whole tree, from the leaf cluster to the root. So the worst case time is governed by the height of the tree, $\lceil \log_2 n - \log_2 max \rceil$. Importantly, then, prediction time is *independent* of the number of ratings (and items), and it is less than linear in the number of users ($O(log_2 n)$).

Space requirements are higher than for the Exhaustive algorithms, but still acceptable. We must store the RecTree ($2^h + 2^{h-1} + \ldots + 1$ nodes, where h is the height of the tree). Each node stores just its centroid, containing one rating per item, and so is of size m. We must also store each user with a reference to his/her leaf cluster.

New users, new items and new ratings can all be accommodated efficiently on-line. A new user must be inserted into the most appropriate leaf cluster. We would do this by walking down the tree from the root to a leaf, at each step selecting the child for which the correlation between the new user's ratings and the child's centroid is the greater. This will take time bounded by hm.

A new item simply requires that each centroid's vector of ratings be extended in length. A new rating (for an existing user and item) requires updates to the centroid rating for that item in the user's leaf cluster and its ancestors. These updates can be done incrementally. However, this will not have any affect on the predictions we would make for the user who supplied the new rating: s/he is still in the self-same cluster. To counter this, if a user supplies a large number of new ratings, then it may be better to delete the user from his/her leaf cluster (incrementally updating centroids in the tree appropriately), and then treat this user as a new user. This gives the possibility, if this user's ratings are now much changed, that s/he will be placed into one of the other clusters. Of course, this is still not as good as re-generation of the tree, since it assumes that the existing clusters are basically correct. Periodic re-generation will still be necessary.

7 Comparisons with Related Work

We look in particular here at work that uses clustering for Collaborative Recommenders. In [2], we claimed that clustering can be used for two different purposes: *partitioning* and *grouping*. In partitioning, a dataset is divided so that search can be confined to one of the partitions; in grouping, a dataset is divided and then a composite object (a 'super-user' or 'super-item') acts as proxy for the members of the group.

In [2], we described the Clustered Users algorithm, which is a neighbourhood-based algorithm that uses the RecTree to *partition* users. Its search for neighbours is confined to users who are in the same leaf cluster as the active user. Hence, all equations in Section 3 are used, but with u ranging only over members of c, the active user's leaf cluster.

In [2], we describe how Clustered Users makes non-personalised

predictions in certain cases. When we construct the RecTree, it is possible that some leaf clusters will have very few members; if they have fewer than a certain threshold (in our case 10), we call that node of the tree an *outlier node*. During prediction, if the user's leaf cluster is an outlier node, then we ascend the RecTree *one* level and use the centroid rating from the parent. This is similar to the recommender algorithm described by Chee [3]. Our results can be seen in [2]. Clustered Users is much less accurate, has much lower coverage but is more efficient than Exhaustive. However, Clustered Users is not competitive with RecTree Centroid: its accuracy, coverage and efficiency are worse across all dataset sizes.

We have recently investigated a variant of Clustered Users. We abandon the idea of outlier nodes, and instead we incorporate ideas from Exhaustive with Defaults and RecTree Centroid. Specifically, if the number of neighbours (still drawn from the active user's leaf cluster) who have rated i falls below a minimum threshold, we invoke the RecTree Centroid method of finding a rating, i.e. we climb the RecTree until we reach either the root or a node whose centroid includes a rating for i other than $\perp$. We have not published the results for this algorithm because, unfortunately, it is not competitive. On the positive side, its coverage is maximal. But, its accuracy is worse than Exhaustive with Defaults and RecTree Centroid for all dataset sizes. And, while it remains more efficient than Exhaustive algorithms, the fact that it is still memory-based means that it is much slower than RecTree Centroid.

Breese et al. describe a model-based approach in which users are clustered probabilistically [1]. A prediction takes the form of a probability for a rating. It is not possible to compare their results directly with ours because they use different datasets and different evaluation measures. One observation, however, is that their clustering algorithm performed relatively poorly compared with their Bayesian network and memory-based approaches. Since our clustering method does well against our memory-based method, this gives us reason to hope that we have found a better way of using clustered data.

Fisher et al. use just the k-means algorithm to *group* users [4]. They store the centroids of the clusters, but they have no tree as we do. To make predictions, their Clustered Pearson Predictor treats the centroids as super-users and seeks neighbours only among the super-users. This was the most accurate and scalable of the algorithms they used. But, their results are given for a different dataset from ours and are not plotted for different total numbers of ratings. So, again, no direct comparison can be given. One observation is that their dataset contained 60000 users for which they created 5000 clusters; to find neighbours therefore requires at least 5000 operations. If we had built a RecTree for this dataset (assuming a maximum leaf cluster size of 300 again), the height of the tree, which determines prediction effort, would have been $\lceil \log_2 60000 - \log_2 300 \rceil \approx 7$.

It is possible, of course, to cluster *items* instead of, or as well as, users. O'Connor & Herlocker *partition* items using Pearson correlation [7]. To make a prediction for item i, the Exhaustive algorithm is applied only to co-rated items from i's partition. O'Connor & Herlocker expected accuracy and coverage to be higher, but found experimentally that this was not so, irrespective of which of several clustering algorithms they used. Efficiency, of course, improves but continues to take time proportional to the number of users.

In [2], we describe an algorithm we call Clustered Items in which we *group* items, to produce super-items. This gives a denser dataset. Applying the Exhaustive algorithm to this denser dataset gives accuracy that is only a little worse than Exhaustive and coverage that is slightly higher than Exhaustive. Coverage is still not maximal although one could introduce default predictions to obtain maximal coverage. As one would expect with a denser dataset, the algorithm is far less efficient than Exhaustive.

It is possible to cluster *both* users and items. In [2], we plotted results for a Dual Clustered algorithm. We *partitioned* users first and then *grouped* items into super-items. We chose this ordering to avoid the reduction in accuracy that would occur if we were to cluster users after their ratings for items had been 'collapsed' into ratings for super-items. To make a prediction, the algorithm first finds the item's super-item and then applies neighbourhood-based methods to the user's cluster to make a prediction for the super-item. Unfortunately, this was our least efficient and least accurate algorithm.

In [8], Ungar & Foster cluster users based on items, and then items based on users; then users are clustered based on item clusters, and items based on user clusters; and then this is repeated three times. The item clustering can be done from data in the ratings matrix r but, in their experiments on real CD purchase data, item clustering was done in a content-based way, by CD artiste. Experimental prediction results are not given.

In contrast, Kohrs & Merialdo cluster users and items independently into two cluster hierarchies [6]. The hierarchies are produced in a way that is highly similar to the way we build our RecTrees. Their prediction formula is very different: they use a weighted sum of the centroids of all nodes on the path in the user's hierarchy from the user's leaf cluster to the root and all nodes on the path in the item hierarchy from the item's leaf cluster to the root. The weights are based on cluster distortion, this being the sum of the distances between ratings and the centroid rating. Results are presented for a different dataset from ours. We regard a proper comparison between their algorithm and ours to be an objective of our future work.

In conclusion, we have proposed two new algorithms, Exhaustive with Defaults and RecTree Centroid. Both have maximal coverage. Their relative usefulness, therefore, depends on the trade-off between accuracy and efficiency: Exhaustive with Defaults is marginally more accurate, while RecTree Centroid is hugely more efficient. In the domains in which Collaborative Recommenders are used, there may be tens of thousands, if not millions, of items and users. In domains with these characteristics, RecTree Centroid has the greater promise: it is likely to meet response time requirements, while still making reasonably accurate predictions.

REFERENCES

[1] J. S. Breese, D. Heckerman, and C. Kadie, 'Empirical analysis of predictive algorithms for collaborative filtering', in *Procs. of the 14th Annual Conference on Uncertainty in Artificial Intelligence*, pp. 43–52, (1998).

[2] D. Bridge and J. Kelleher, 'Experiments in sparsity reduction: Using clustering in collaborative recommenders', in *Procs. of the Thirteenth Irish Conference on Artificial Intelligence and Cognitive Science*, eds., M.O'Neill, R.F.E.Sutcliffe, C.Ryan, M.Eaton, and N.J.L.Griffith, pp. 144–149. Springer, (2002).

[3] S. H. S. Chee, *RecTree: A Linear Collaborative Filtering Algorithm*, Master's thesis, Simon Fraser University, 2000.

[4] D. Fisher, K. Hildrum, J. Hong, M. Newman, M. Thomas, and R. Vuduc, 'Swami: A framework for collaborative filtering algorithm development and evaluation', in *Procs. of SIGIR*, pp. 366–368, (2000).

[5] J. L. Herlocker, *Understanding and Improving Automated Collaborative Filtering Systems*, Ph.D. dissertation, University of Minnesota, 2000.

[6] A. Kohrs and B. Merialdo, 'Clustering for collaborative filtering applications', in *Procs. of CIMCA'99*. IOS Press, (1999).

[7] M. O'Connor and J. Herlocker, 'Clustering items for collaborative filtering', in *Procs. of the ACM SIGIR Workshop on Recommender Systems*, (1999).

[8] L.H. Ungar and D.P. Foster, 'Clustering methods for collaborative filtering', in *AAAI Workshop on Recommendation Systems*, (1998).

Evaluation-Based Semiring Meta-Constraints [1]

Jerome Kelleher and **Barry O'Sullivan** [2]

Abstract. Classical constraint satisfaction problems (CSPs) provide an expressive formalism for modelling and solving many real-world problems. However, classical CSPs prove to be restrictive in any situation where uncertainty, fuzziness, probability, optimisation or partial satisfaction are intrinsic. Soft constraints alleviate many of the restrictions imposed by classical constraint satisfaction. In particular, soft constraints provide a basis for capturing notions such as vagueness, uncertainty and cost in the CSP model. In this paper we focus on the semiring-based approach to soft constraints. We present an overview of soft constraints, and the recent functional formulation of the semiring framework in particular, which also plays a tutorial role in this paper. Furthermore, we present a new evaluation-oriented scheme for implementing meta-constraints, which can be applied to any existing implementation to improve its time and space efficiency.

1 Introduction

Classical constraint satisfaction problems (CSPs) provide an expressive formalism for modelling and solving many real-world problems. CSPs allow us to express *constraints* over variables; these constraints state the allowed combinations of instantiated values for variables. In this way we can declaratively state problems and pass the burden of finding solutions to these problems onto the constraint solver.

However, classical CSPs prove to be restrictive in any problems in which uncertainty, fuzziness, probability, optimisation or partial satisfaction are intrinsic. *Soft* constraints alleviate many of these restrictions imposed by classical constraint satisfaction. We present an overview of the semiring framework for soft constraint satisfaction, which can handle all of the above aspects of softness as well as traditional crisp constraints in a unified and elegant manner.

In this paper we discuss the specification and implementation of semiring *meta-constraints* (constraints which depend on other constraints). We show that implementing meta-constraints using the compilation-based approach is fundamentally flawed and results in any algorithm which utilises these useful abstractions having exponential time and space complexity. We show how these problems can be very simply resolved by instead adopting an *evaluation*-based approach to specifying and evaluating these constraints, which we show to be unrestrictive for the definition of more complex constraint processing algorithms.

We advocate the use of semiring meta-constraints to reduce the complexity of defining algorithms to efficiently solve soft constraint problems. Such algorithms have been defined in the system given in [2], which are unfortunately highly inefficient due to the representation of meta-constraints used.

Therefore, the contributions of this paper are as follows:

- We present an overview of soft constraints, and the recent functional formulation of the semiring framework in particular, which is useful for tutorial purposes;
- We present a new evaluation-oriented scheme for implementing meta-constraints. This scheme can be applied to any existing implementation to alleviate problems of excessive space usage - which has concomitant negative implications for the time complexity of constraint processing algorithms.

The paper is organised as follows. Section 2 introduces some of the schemes for solving soft constraint problems over different domains of interest, and shows how these all use different ideas of consistency and satisfaction. Section 3 then shows how the semiring framework of Bistarelli et al. [3, 4, 5] can unify many disparate models of constraint satisfaction by using a semiring structure to represent consistency levels and the operations needed to combine and compare those levels. We explain semiring meta-constraints and provide some pedagogical examples of *evaluation*-based meta-constraints. In Section 4 we present a brief discussion of the applicability of local-consistency techniques to soft constraint problems, and of the inherent limitations of any system which utilises these techniques. Section 6 presents our scheme for the implementation of evaluation-based meta-constraints, and Section 7 presents some basic results on the runtime efficiency that can be expected for systematic search using these abstractions over problems of different tightness. Finally, Section 8 summarises the ideas of this paper and hints at possible future lines of work.

2 Soft Constraints

In this section we informally present an overview of some of the different models for soft constraints which can be cast in the semiring framework (for a more formal and complete treatment of this subject the reader is referred to the literature [3, 4]). For each of the models we will note three separate facets of the model: the set of consistency values used in the model, the operation used to combine consistency values to determine a total consistency value, and the operation used to compare consistency values to determine which is 'better', if any.

We use the term 'consistency' (or α-consistency [5]) here to denote the degree to which a particular problem is satisfied, according to whatever criteria specified in that particular model. This is an extension of the idea of consistency from classical constraint satisfaction where a problem is consistent if it contains a solution and inconsistent otherwise. In soft constraints we do not have a simple boolean concept of consistency, which allows us to specify any number of optimisation schemes.

In the following discussion we will use the ideas of variables and constraints. Informally, a variable is any single element of a problem which can be assigned values from a fixed domain. A constraint is then a function over some set of variables which returns a consistency value when evaluated for some instantiation of its variables.

[1] This work has received support from Enterprise Ireland under their Basic Research Grant Scheme (Grant Number SC/02/289).

[2] Cork Constraint Computation Centre, Department of Computer Science, University College Cork.
Email:{jt.kelleher|b.osullivan}@cs.ucc.ie

2.1 Vagueness

Fuzzy constraints [7, 15] are a significant extension of classical constraint satisfaction. Fuzzy constraints allow us to deal with imprecision and vagueness in constraint-based reasoning by defining constraints as fuzzy set membership functions. The set of consistency values is then defined as the continuous interval $[0, 1]$, 0 denoting definite exclusion of a value from a fuzzy set, 1 denoting definite inclusion in the fuzzy set and all of the intermediate values representing an object's degree of membership of the fuzzy set in question. This is usually interpreted as the degree to which a given constraint is satisfied by a particular instantiation of its variables. This greatly enriches the expressiveness of the constraint satisfaction paradigm as we can naturally express, for example, user preferences, which are often imprecise and vague, and very rarely crisp.

Combining two fuzzy constraints is then equivalent to finding the intersection of two fuzzy sets. For this reason we use the min function, as this is the method used to define the membership function of the intersection of two fuzzy sets. To compare two consistency values to determine which is better we use the max function, as this allows us to determine which value satisfies constraints the most.

Fuzzy constraints can also model ([1]) Partial Constraint Satisfaction [9] which allows for solutions to be found even when a problem is over constrained in classical constraint satisfaction. Fuzzy constraints can also easily model prioritised constraints [6], in which constraints have associated levels of importance.

2.2 Uncertainty

Probabilistic constraint satisfaction can be used to model uncertainty in constraint-based reasoning. In this context the consistency level returned by evaluating a constraint is interpreted as the probability of the event represented by that instantiation occurring; hence the set of consistency values is drawn from the interval $[0, 1]$. Each constraint is then an independent probability function.

To compute the total probability of two independent events we use the product rule, i.e. $P(A\ and\ B) = P(A) \times P(B)$. Therefore, to combine consistency values in the probabilistic CSP model we use multiplication of reals. The best solution of a probabilistic CSP is obviously the instantiation of the variables with the highest probability. Hence, to compare consistency levels we use the max function.

2.3 Cost

Weighted CSPs are used to model situations when problems have sets of individual weighting functions associated with variables. The result of evaluating a constraint function under an instantiation is then the cost of this instantiation according to that particular function. The overall cost of a complete instantiation of the variables is then computed by summing the individual costs of each cost function. This allows us to declaratively state very complex optimisation problems using a problem decomposition approach.

To find a solution to a problem of this type we wish to find a configuration which minimises the total cost over all functions. The maximum consistency possible represents a cost of zero - the closer that we can get to this ideal the better. The minimum consistency possible represents an infinite cost, which we can use to model configurations which are either highly undesirable or indeed impossible. Therefore, the set of all possible consistency values is the reals.

This allows us to model optimisation problems in which the minimisation of overall cost (time, money, resources, etc) is paramount. This describes a large class of real-world optimisation problems.

3 Semiring Framework

The semiring framework for constraint satisfaction is based upon the central observation that a semiring (a set together with two operations which satisfy certain properties) is all that is needed to describe many constraint satisfaction schemes. The semiring set provides the levels of consistency, which can be interpreted as cost, degrees of preference, probabilities or any other criteria consistent with the requirements of the framework. The two operations then allow us to combine ($\times$) and to compare ($+$) consistency levels from this set.

In the interest of brevity we will restrict our discussion of the semiring framework under the functional formulation [5] to a brief statement of the basic ideas involved. For a more detailed and rigorous treatment of the subject the reader is referred to the literature [1, 3, 4, 5], where many key results pertaining to this framework are proved.

Semirings. A c-semiring (constraint-semiring) is a tuple $\langle A, +, \times, \mathbf{0}, \mathbf{1}\rangle$ such that:

- A is the set of all consistency values and $\mathbf{0}, \mathbf{1} \in A$. $\mathbf{0}$ is the lowest consistency value and $\mathbf{1}$ is the highest consistency value;
- $+$, the additive operator, is a closed, commutative, associative and idempotent operation such that $\mathbf{1}$ is its absorbing element and $\mathbf{0}$ is its unit element;
- $\times$, the multiplicative operator, is a closed and associative operation such that $\mathbf{0}$ is its absorbing element, $\mathbf{1}$ is its unit element and $\times$ distributes over $+$.

The c-semirings for some typical instances of the semiring framework are:

- Crisp CSP: $\langle\{false, true\}, \vee, \wedge, false, true\rangle$;
- Fuzzy CSP: $\langle\{x \mid x \in [0, 1]\}, max, min, 0, 1\rangle$;
- Probabilistic CSP: $\langle\{x \mid x \in [0, 1]\}, max, \times, 0, 1\rangle$;
- Weighted CSP: $\langle\mathcal{R}^+, min, +, +\infty, \mathbf{0}\rangle$;
- Set-based CSP: $\langle\wp(A), \cup, \cap, \emptyset, A\rangle$.

Constraint Problems. Given a semiring $S = \langle A, +, \times, \mathbf{0}, \mathbf{1}\rangle$ and an ordered set of variables V over a finite domain D, a *constraint* is a function which, given an assignment $\eta : V \to D$ of the variables, returns a value of the semiring. By using this notation we define $\mathcal{U} = \eta \to A$ as the set of all possible constraints that can be built starting from S, D and V.

In this *functional* formulation of the semiring framework each constraint is a function (as defined in [5]) and not a pair (as defined in [3, 4]). Each constraint function involves all the variables in V, but it depends on the assignment of only a finite subset of them. For example, a binary constraint $c_{x,y}$ over variables x and y, is a function $c_{x,y} : V \to D \to A$, but it depends only on the assignment of variables $\{x, y\} \subseteq V$. We call this subset the *support* of the constraint.

More formally, consider a constraint $c \in \mathcal{U}$. We define its support as $supp(c) = \{v \in V \mid \exists \eta, d_1, d_2 . c\eta[v := d_1] \neq c\eta[v := d_2]\}$, where

$$\eta[v := d]v' = \begin{cases} d & \text{if } v = v', \\ \eta v' & \text{otherwise.} \end{cases}$$

Note that $c\eta[v := d_1]$ means $c\eta'$ where η' is η modified with the assignment $v := d_1$ (that is, the operator $[\,]$ has precedence over application).

A *soft constraint satisfaction problem* is a pair $\langle C, con\rangle$ where $con \subseteq V$ and C is a set of constraints: con is the set of variables of interest for the set of constraints C, which may also concern variables not in con.

3.1 Semiring Meta-Constraints

In this paper we introduce the term *semiring meta-constraints* (or more concisely, meta-constraints) as a convenient means of referring to constraint functions defined over other constraints in the semiring framework. Several classes of meta-constraints have been defined in the literature, including *combination* constraints, *projection* constraints, *solution* constraints and $blevel$ constraints [1, 3, 4, 5]. In this paper we will focus on combination and projection meta-constraints as both *solution* and $blevel$ meta-constraints are defined in terms of these primitives.

Combination Meta-Constraints. Given the set $\mathcal{U}$, the combination function $\bigotimes$ is defined as $(\bigotimes C)\eta = \prod_{c \in C} c\eta$. This function takes a set of constraints and returns a combination meta-constraint. This definition is the straightforward extension of the $\otimes$ function [5] to sets of constraints.

Informally, combination meta-constraints represent the constraint which is equivalent to all of the constraints in C combined together. This is a very useful abstraction as it allows us to perform all reasoning over single constraints instead of cumbersome sets of constraints. To evaluate a given combination meta-constraint for an instantiation of the variables η simply involves evaluating all of its constituent constraints under η and combining the individual consistency values using the semiring $\times$ operator.

Projection Meta-Constraints. Given a constraint $c \in \mathcal{U}$ and a variable $v \in supp(c)$, the *projection* function $\Downarrow$ is defined as $(c \Downarrow_{(supp(c)-\{v\})})\eta = \sum_{d \in D} c\eta[v := d]$. This function takes a constraint and a set of variables as parameters and returns the constraint which is equivalent to the original constraint with its support reduced to the specified set of variables.

Informally, projecting a constraint c over the set of variables $(supp(c) - \{v\})$ returns a constraint c' which is equivalent to c with the variable v removed from the support. This is done by evaluating $c\eta[v := d]$ (for the instantiation of interest) for all domain values d in the domain of v, and returning the sum of all of these individual consistency values using the semiring additive operator $+$. Effectively then, the value returned from evaluating $c'\eta$ is the maximum consistency value possible for the instantiation of variables η if we can choose any value for the instantiation of v.

3.2 An Example

In this section we present an example weighted constraint problem. In this problem we have two variables, x and y defined over the domain $D = \{1, 2, 3, 4, 5\}$. As this is a weighted constraint problem we use the semiring $S = \langle \mathcal{R}^+, min, +, +\infty, 0 \rangle$. In a problem of this type we have a set of cost functions defined over the variables of interest; each individual cost function describes the cost of one specific section of a configuration under a particular instantiation of that variable or the cost of instantiations over related variables (non-unary constraints). For simplicity we define a generic cost function $cost(a, n) = (n - a)^2$ to enable us to easily demonstrate the ideas in question. In general however, any arbitrary function can be used to describe costs.

In particular, we will define three constraints denoted c_x, c_y and $c_{x,y}$ defined as follows,

$$c_x\eta = cost(2, x),$$
$$c_y\eta = cost(4, y),$$
$$c_{x,y}\eta = cost(1, y - x).$$

Unary constraints c_x and c_y are intended to represent the costs associated with an instantiation deviating from an ideal value. For instance, the ideal value for x according to c_x is 2 and any instantiation where x is not set to this value will be penalised proportional to the square of its distance from this value. Binary constraint $c_{x,y}$ is used to illustrate the idea that we can easily model complex interrelationships between variable instantiations.

The constraint problem in this example is then given by $P = \langle \{c_x, c_y, c_{x,y}\}, \{x, y\} \rangle$. To allow us to demonstrate the ideas of evaluation oriented meta-constraints introduced in this paper we will give examples of combination and projection meta-constraints over this problem.

Combination. In this example we demonstrate the evaluation of a combination meta-constraint. To evaluate a combination meta-constraint for a particular instantiation we must evaluate each of the constituent constraints under the instantiation in question and find the product of these values using the semiring multiplicative operator.

In particular, we demonstrate the evaluation of the combination of the constraints c_x, c_y and $c_{x,y}$ - i.e. $\bigotimes\{c_x, c_y, c_{x,y}\}$ - under the instantiation where x has the value 1 and y has the value 5 ($\eta[x := 1, y := 5]$), i.e.,

$$\begin{aligned}
(\bigotimes\{c_x, c_y, c_{x,y}\})\eta[x := 1, y := 5] &= \\
c_x\eta[x := 1, y := 5] &= cost(2, 1) = 1 \\
&\times_s \\
c_y\eta[x := 1, y := 5] &= cost(4, 5) = 1 \\
&\times_s \\
c_{x,y}\eta[x := 1, y := 5] &= cost(1, 4) = 9.
\end{aligned}$$

As the semiring multiplicative operator in this case is addition over reals, the overall cost associated with this instantiation of the variables $\eta[x := 1, y := 5]$ is 11.

Projection. In this example we demonstrate a projection meta-constraint evaluation. In particular we will demonstrate the evaluation of the projection of the constraint $c_{x,y}$ over the set $\{x\}$, i.e. the meta-constraint where we remove y from the support of $c_{x,y}$. Specifically then, we will evaluate $c_{x,y} \Downarrow_{\{x\}}$ under the instantiation $\eta[x := 1, y := 5]$, i.e., the instantiation where x has the value 1 and y has the value 5.

To evaluate the projection of a constraint over a particular subset of its support for a given instantiation we must evaluate the constraint in question for all domain values of variables which have been removed from the support. We then find the sum of all of the individual consistency values using the semiring additive operator, $+_s$.

In this particular example we are projecting the constraint $c_{x,y}$ over the set $\{x\}$. This means we are removing the variable y from the support of the constraint. When we remove a variable from the support of a constraint we are effectively saying that we are not interested in this variable, and we therefore wish to find highest consistency value possible if we can assign any value from D to this variable. To do this we must evaluate the constraint in question for all possible instantiations of y, i.e.,

$$\begin{aligned}
(c_{x,y} \Downarrow_{\{x\}})\eta[x := 1, y := 5] &= \\
c_{x,y}\eta[x := 1, y := 1] &= cost(1, 0) = 1 \\
&+_s \\
c_{x,y}\eta[x := 1, y := 2] &= cost(1, 1) = 0 \\
&+_s \\
c_{x,y}\eta[x := 1, y := 3] &= cost(1, 2) = 1 \\
&+_s \\
c_{x,y}\eta[x := 1, y := 4] &= cost(1, 3) = 9 \\
&+_s \\
c_{x,y}\eta[x := 1, y := 5] &= cost(1, 4) = 16.
\end{aligned}$$

As the semiring additive operator for the weighted semiring is the min function over reals, the result of evaluating this constraint is 0.

One important idea illustrated in this example is the concept of the support of a constraint. In this example, the support of $c_{x,y} \Downarrow_{\{x\}}$ is $\{x\}$. This means that this constraint depends only on the assignment of values to variable x. This is demonstrated in the example when we evaluate the constraint $c_{x,y} \Downarrow_{\{x\}}$ under the instantiation $\eta[x := 1, y := 5]$, but we evaluate the constraint that it depends on ($c_{x,y}$) for all instantiations where $x := 1$ and $y := d$.

4 Local Consistency in Soft Constraints

The application of local-consistency techniques has been highly successful in classical constraints. Local-consistency techniques choose sub-problems in which to eliminate local inconsistency and then iterate this elimination in all chosen sub-problems until stability. The most widely used local-consistency techniques are arc-consistency algorithms, in which sub-problems contain only one constraint. These techniques have been used to increase search efficiency in crisp CSPs very successfully.

In [4, 3] the authors explored the applicability of local-consistency techniques to instances of the semiring framework, and found that an extra restriction must be placed on the framework to allow for the application of these methods[3]. This extra restriction is the requirement that the $\times$ semiring operation be idempotent for local-consistency techniques to be meaningful. Specifically, if the $\times$ operator is not idempotent, then, in general, any local consistency algorithm cannot be guaranteed to:

1. terminate;
2. return a problem which is equivalent to the original one;
3. be independent of non-deterministic choices made during the algorithm.

Requiring the $\times$ operator to be idempotent is a severe limitation on the expressiveness of the semiring framework. The operation $\times : A \times A \rightarrow A$ is idempotent if for all $a \in A$, $a \times a = a$. As the multiplicative operation is used for combining consistency values this is an unnatural restriction, as this disallows the cumulative aggregation of consistency values. For instance, the standard addition (weighted semiring) and multiplication (probabilistic semiring) operations defined over the reals do not satisfy this property.

On the other hand, in [3, 4] the authors showed that the concept of Dynamic Programming can be usefully applied to *any* instance of the semiring framework, without requiring the idempotency of the $\times$ operation. This may be a fruitful direction for research into more efficient soft constraint solving algorithms.

5 Existing Implementations

In this section we discuss the published implementations of the semiring framework. There are a number of issues with these implementations: these range from limitations on the types of semirings that can be handled, to runtime efficiency issues.

5.1 `clp(FD,s)`

In [11] the authors present an extension of the `clp(FD)` [8] system, `clp(FD,s)`. This system provides an efficient means of solving constraint problems defined over a subset of the semirings in the semiring framework. However, no implementation of the combination and projection meta-constraints is provided.

In this system, the authors explicitly restrict the scope of the solver to those semirings in which $\times$ is idempotent, and hence do not support the full generality of the semiring framework. Many of the techniques used to gain efficiency utilise properties only present in semirings where the multiplicative operation is idempotent. This may seem like a reasonable compromise; however, this design decision prevents problems defined over the Probabilistic and Weighted semirings from being solved using this system.

[3] Subject to certain caveats.

5.2 Soft CHR

In [2] the authors present an implementation of the semiring framework based on CHRs [10]. CHRs allow for the simplification and propagation of constraints and have been successfully deployed in dozens of projects to implement various crisp solvers. However, as propagation cannot be applied to instantiations where the multiplicative operation is not idempotent, the usefulness of CHRs is limited in this context.

However, the system does provide several algorithms which can be used over all instances of the semiring framework, including Branch & Bound algorithms with both variable and constraint labelling, as well as a Dynamic Programming search algorithm. Unfortunately, the implementation of meta-constraints in this system severely limits the applicability of these useful algorithms.

In this system all meta-constraints are represented *extensionally* as a list of tuple-consistency pairs, using the compilation-based scheme (see Section 6). Savings in space usage are attained by not storing tuples with consistency of zero. However, in general, a k-ary meta-constraint will require exponential time and space to compile and store. Moreover, many of the more complex operations for this system - such as the dynamic-programming solver - use this operation heavily, ensuring that these operations require exponential time and space also.

In the next section we present a simple method to solve this problem of exponential time and storage. Hopefully, this new method can be integrated into the system, which may allow the useful general purpose algorithms provided in the system be applied to non-trivial problems.

6 A New Approach to Implementing Meta-Constraints

While a large amount of work has been published on the theoretical aspects of soft constraints, apart from the two implementations mentioned in Section 5 very little has been published on the subject of practical implementation of soft constraints. We advocate the use of semiring meta-constraints as a useful abstraction to reduce the difficulty of implementing efficient algorithms to solve soft constraint problems in general.

However, currently meta-constraints are not viable as they are both specified and implemented using a compilation-based approach. By compilation-based we mean that when a meta-constraint function is created a lookup table of all possible input values and their corresponding output results is computed and stored. This approach is extraordinarily wasteful of both computing time and space. For instance, if we had a ternary meta-constraint function over variables with domains of size twenty, we would need to compile a lookup table with 8000 entries. In general, if we have a compilation based meta-constraint function over a set of variables V with domain D, then we will require a lookup table with $|D|^{|V|}$ entries to fully encode the function. This means we need *exponential* time and space to construct these functions.

For example, consider the function $f(x, y)$ shown in Table 1. In this example we show a function which is composed of two functions over different variables with their respective results added to-

Table 1. Compilation-based function $f(x, y) = x^2 + y^3$ defined over domain $\{1, \ldots, 5\}$.

x	y	$x^2 + y^3$
1	1	2
1	2	9
1	3	28
⋮	⋮	⋮
5	5	150

gether. This is analogous to a combination meta-constraint, which is a function composed of a number of separate functions over different variables with their results combined together using some simple operation. The variables in this function x and y are defined over the domain $D = \{1, \ldots, 5\}$. Even with this tiny domain, it is necessary to truncate the lookup table which we are using for explanatory purposes.

6.1 The Evaluation-Oriented Approach

A far more economical (and indeed simpler) method of implementing meta-constraint functions is to simply store the original constraint functions that are involved and evaluate these *as required* with the instantiation of interest. This approach is a direct implementation of the definition of the combination function, $(\bigotimes C)\eta = \prod_{c \in C} c\eta$. In this way we can create a new meta-constraint function in constant time and with space linear in the number of constraints involved. This is the *evaluation*-based method of implementing meta-constraints.

One possible criticism of this evaluation-based approach is that there may be situations where we need know the value of all possible instantiations for a particular meta-constraint, and furthermore, we may need to find out the value of a particular instantiation many times. However, these situations are hard to imagine and still do not warrant the *storage* of all possible instantiations. If we wish to find the value of every possible instantiation for a given meta-constraint we can simply iterate through all possible instantiations and evaluate the constraint for that instantiation. If the value of an instantiation will be needed many times, it is the responsibility of the specific algorithm which requires this property to determine if it is worthwhile caching the value, *not* the function which calculates it.

Furthermore, if we make the not-unreasonable assumption that in the majority of constraint processing algorithms that we define, we will want to find the value of the least number of instantiations possible, the compilation scheme is highly undesirable. To sum up, *any* algorithm that we define in terms of compilation based meta-constraints will have exponential time and space complexity, regardless of the semantics of the algorithm itself.

6.2 Combination Evaluation

Combination meta-constraints are an extremely useful abstraction as they allow us to treat a set of constraints as a single constraint. Thus, any reasoning or operations that deal with constraints can be defined over a single constraint as we can refer to any set of constraints by their combination as a single constraint. This simplifies both theoretical and practical work with constraints.

Combination is a universal operation in constraint satisfaction. Any form of constraint processing which deals with distinct sets of constraints can all be expressed in terms of this operation. Therefore any improvements we make in the time or space efficiency of this operation will have knock-on effects on any other more sophisticated constraint processing that we perform.

To evaluate a combination meta-constraint defined over the set of constraints C at runtime for a given instantiation η we use Algorithm 1. In this algorithm we simply iterate through all of the constraints in C and evaluate each one under the instantiation in question. To prevent unnecessary computation, we use the fact that $\mathbf{0}$ is the absorbing element of the $\times$ operation. In this way, we know that if any single function evaluates to $\mathbf{0}$ then the entire combination constraint will also evaluate to $\mathbf{0}$ and we can therefore immediately return $\mathbf{0}$.

Algorithm 1 CombinationEvaluate(η)

```
a ← 1
for all c ∈ C do
  a ← a × cη
  if a = 0 then
    return 0
  end if
end for
return a
```

As this lazy-evaluation leverages the full generality of the semiring framework, it applies to *all* instances. For example, in the crisp semiring, this optimisation reduces to the lazy evaluation of the boolean AND operation; over the fuzzy semiring, it reduces to the lazy evaluation of the min function defined over the interval $[0, 1]$.

7 Experimental Evaluation

In this section we present some results on a prototype branch and bound solver developed using combination meta-constraints. To quantitatively determine the performance implications of using an evaluation-oriented method for representing meta-constraints, combination meta-constraints in this system are implemented using both the compilation and evaluation scheme. Specifically, we tested the number of constraint evaluations required to find the set of best solutions to random soft constraint problems using compilation and evaluation based combination meta-constraints.

To generate random soft constraint problems we follow the methodology adopted in [14], in which random Fuzzy CSPs are generated with four specific properties: the number of variables n, the number of domain values per variable m, the density d and tightness t. The tightness of a problem is defined as the number of instantiations which evaluate to 0 over the total number of instantiations for each binary constraint. The remaining instantiations are then assigned a consistency value from the interval $(0, 1]$, which is randomly generated with a uniform distribution. To ensure that anomalous results are not reported over random problems, we performed 50-fold cross validation over results obtained, i.e., we generated 50 random Fuzzy problems with the required specifications and report the average results over these instances.

To implement compilation-based meta-constraints as efficiently as possible we used Algorithm 1 to compute the associated semiring value for each instantiation of the variables in the support of the meta-constraint in question, thus minimising the number of constraint evaluations required. To minimise space requirements, we follow the methodology used in [2] and store only associations between instantiations with non-zero semiring values.

For each problem instance we count the number of constraint evaluations required to find the set of best solutions to the problem using a branch and bound algorithm. This is a vital operation for any soft constraint solver. Figure 1 shows the results of these experiments. Plotted on the x-axis is the problem tightness, or the proportion of instantiations in each constraint which evaluate to zero and on the x-axis the number of constraint evaluations required to find the set of best solutions.

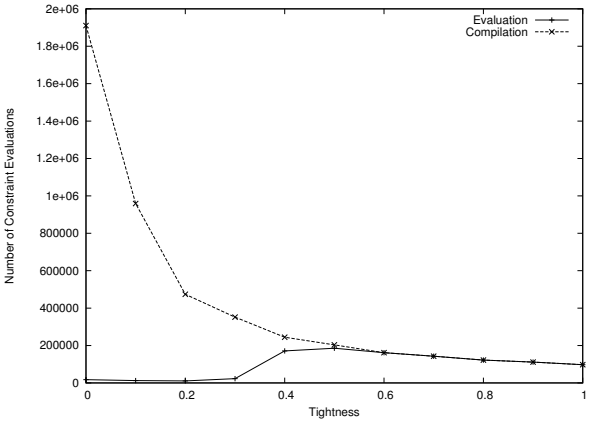


Figure 1. Branch and Bound search using evaluation-based and compilation-based combination meta-constraints. Random problems have 7 variables, 5 domain values and a density of 1.0.

With low tightness values the evaluation oriented approach vastly outperforms the compilation-based approach. This is because we only evaluate constraints on demand in the evaluation-based approach and, because very few constraints evaluate to zero, the bound can be quickly updated and significant branch pruning occurs. In the compilation-based approach however, the entire state-space must be first generated, a value corresponding to the instantiation in question computed, and the association between these two stored *before* any search occurs, resulting in exponential space and time complexity.

At higher levels of problem tightness the performance in terms of constraint evaluations of the two schemes converges. This is because as the difficulty of the problem increases the bound will not be updated and in general we will have to explore a larger proportion of the state-space. The total number of constraint checks required to find the set of best solutions decreases (counter-intuitively) for the compilation-based scheme as tightness increases. This is due to the lazy evaluation given in Algorithm 1, which allows values to be calculated without requiring all of the constraints to be evaluated when compiling the meta-constraint.

To sum up, the compilation-based approach to implementing meta-constraints is fundamentally flawed as it results in the computation *and* storage of a great deal of information which may not be required to solve a particular problem. The evaluation-based approach is a far simpler and more efficient means of implementing these meta-constraints.

8 Conclusions and Future Work

Classical constraint satisfaction problems (CSPs) provide an expressive formalism for expressing and solving many real-world problems in a declarative fashion. However, classical CSPs prove to be restrictive in any situation where uncertainty, fuzziness, probability, optimisation or partial satisfaction are intrinsic. Soft constraints alleviate many of the restrictions which classical constraint satisfaction imposes. In particular, soft constraints provide a basis for capturing notions such as vagueness, uncertainty and cost into the CSP model.

In this paper we have focused on the semiring-based approach to soft constraints. We presented an overview of soft constraints, and the recent functional formulation of the semiring framework in particular, which also provides a tutorial to the reader who may be unfamiliar with the literature on this subject.

Furthermore, we focused on some critical issues related to the implementation of semiring-based constraint solvers. We presented a new *evaluation-oriented* scheme for implementing meta-constraints, which can be applied to any existing implementation to significantly improve its performance in terms of both space and time complexity.

In future work we intend to investigate the viability of building a general purpose soft constraint solver using an object-oriented language. In particular, we intend to develop an object-oriented soft constraint solver modelled in part after the benchmark crisp constraint system, ILOG Solver [12, 13].

Acknowledgements

We would like to thank Stefano Bistarelli for fruitful discussion and comments.

REFERENCES

[1] S. Bistarelli, *Soft Constraint Solving and programming: a general framework.*, Ph.D. dissertation, Dipartimento di Informatica, Università di Pisa, Italy, mar 2001. TD-2/01.

[2] S. Bistarelli, T. Fruehwirth, M. Marte, and F. Rossi, 'Soft constraint propagation and solving in constraint handling rules', in *Proc. of the ACM Symp. on Applied Computing*, pp. 1–5, (2002).

[3] S. Bistarelli, U. Montanari, and F. Rossi, 'Constraint Solving over Semirings', in *Proc. IJCAI95*, pp. 624–630, San Francisco, CA, USA, (1995). Morgan Kaufman.

[4] S. Bistarelli, U. Montanari, and F. Rossi, 'Semiring-based Constraint Solving and Optimization', *Journal of the ACM*, **44**(2), 201–236, (Mar 1997).

[5] S. Bistarelli, U. Montanari, and F. Rossi, 'Soft concurrent constraint programming', in *Proc. ESOP, April 6 - 14, 2002, Grenoble, France*, LNCS, pp. 53–67, Heidelberg, Germany, (2002). Springer-Verlag.

[6] Alan Borning, Michael Maher, Amy Martindale, and Molly Wilson, 'Constraint hierarchies and logic programming', in *Proceedings 6th Intl. Conference on Logic Programming*, eds., Giorgio Levi and Maurizio Martelli, 149–164, The MIT Press, Cambridge, MA, (1991).

[7] James Bowen, Robert Lai, and Dennis Bahler, 'Lexical imprecision and fuzzy constraint networks', in *Proceedings of AAAI-92*, pp. 616–621, (July 1992).

[8] Philippe Codognet and Daniel Diaz, 'Compiling constraints in clp(FD)', *Journal of Logic Programming*, **27**(3), 185–226, (1996).

[9] Eugene C. Freuder and Richard J. Wallace, 'Partial Constraint Satisfaction', in *Artificial Intelligence vol. 58*, pp. 21–70, (1992).

[10] Thom Frühwirth, 'Theory and practice of constraint handling rules', *Journal of Logic Programming, Special Issue on Constraint Logic Programming*, **37**(1-3), 95–138, (October 1998).

[11] Y. Georget and P. Codognet, 'Compiling semiring-based constraints with `clp(FD,S)`', in *Proc. of CP-98*, LNCS 1520, pp. 205–219, (1998).

[12] Jean-Francois Puget, 'A C++ implementation of CLP', in *Proceedings of the Second Singapore International Conference on Intelligent Systems*, Singapore, (1994).

[13] Jean-Francois Puget and Michel Leconte, 'Beyond the glass box: Constraints as objects', in *International Logic Programming Symposium*, pp. 513–527, (1995).

[14] F. Rossi and I. Pilan, 'Abstracting soft constraints: Some experimental results', in *Proceedings of the Joint Annual Workshop of the ERCIM Working Group on Constraints and the CoLogNET area on Constraint and Logic Programming*, (2003).

[15] Zsofi Ruttkay, 'Fuzzy constraint satisfaction', in *Proceedings 1st IEEE Conference on Evolutionary Computing*, pp. 542–547, Orlando, (1994).

A False Colouring Real Time Visual Saliency Algorithm for Reference Resolution in Simulated 3-D Environments

John Kelleher[1] and **Josef van Genabith**[2]

Abstract. In this paper we present a novel false colouring visual saliency algorithm and illustrate how it is used in the Situated Language Interpreter system to resolve natural language references.

1 Introduction

The focus of the Situated Language Interpreter (SLI)[3] project is to develop a natural language interpretive framework to underpin the development of natural language virtual reality (NLVR) systems. An NLVR system is a computer system that allows a user to interact with simulated 3-D environments through a natural language interface. The central tenet of this work is that the interpretation of natural language (NL) in 3-D simulated environments should be based on a model of the user's knowledge of the environment. In the context of an NLVR system, one of the user's primary information sources is the visual context supplied by the 3-D simulation. In order to model the flow of information to the user from the visual context, we have developed and implemented a visual saliency algorithm that works in real-time and across different simulated environments. Unlike previous NLVR systems [19, 1, 6, 4, 5, 8] salience in particular visual salience is a crucial component in reference resolutions in the SLI system. This paper describes this algorithm and illustrates how it is used to resolve references

2 Perception and Attention

Although visual perception seems effortless, "psychophysical experiments show that the brain is severely limited in the amount of visual information it can process at any moment in time" [15, pg. 1]. In effect, there is more information perceived than can be processed. The human faculty of attention is the "selective aspect of processing" [9, pg. 84]. Attention regulates the processing of perceived visual stimuli by selecting a region within the visual buffer for detailed processing. Our knowledge of the human attention process is not complete, "but it appears to consist of a set of mechanisms that exhibit different, sometimes opposing effects" [7, pg. 9]. For example, [11] lists: visual familiarity, intentionality, an object's physical characteristics, and the structure of the scene. This multiplicity makes the modelling of visual perception difficult. A priori, one of the major functions of visual attention is object identification. With this in mind, an important factor when considering modelling visual attention is the difference between foveal and peripheral vision. The fovea is a shallow pit in the retina which is located directly opposite the pupil, consisting of cones and is the site of highest visual acuity, the ability to recognise detail. It "drops 50 percent when an object is located only 1° from the centre of the fovea and an additional 35 percent when it is 8° from the centre" [3, pg. 228]. Identifying an object requires the use of foveal vision, occurring when a person looks directly at the object, causing the image of the object falling on the retina to be centred on the fovea. The dependence of object identification on foveal vision implies a relationship between foveal vision and attention. Moreover, this gradation across visual acuity is congruent with the gradation of attention theory. This theory posits that "attention is greatest at a single point, and drops of gradually from that point" [9, pg. 90]. Following this, the more central a location is with respect to the centre of an eye fixation the higher the location's salience. Indeed, the most common computational mechanism for modelling visual attention is a filtering of visual data by removing portions of the input located outside a spatial focus of attention [7].

3 Previous Computational Work

Section 2 examined some of the aspects of perception that pertain to modelling vision, in particular how attention affects the human awareness of what people perceive. It was noted that spatial attention is the most commonly used visual filtering mechanism. There are many computational models of vision that use this abstraction; most have been developed for robot navigation.

[7] reviews several of the robotic attention systems. However, there are two reasons why the models of vision created for robotic systems are not suitable for NLVR systems. First, nearly all of these systems have a connectionist or neural net architecture. This form of system requires training. As a result, these models are restricted to the domains described by or sufficiently similar to the training set given to the system. For example, connectionist navigational systems trained with images from the inside of a factory would need to be retrained to handle a forest environment. A system that requires retraining when shifting from one visual domain to another is not suitable as a model of rendered environments which may change drastically from program to program or even within the one application. Second, the major difficulties facing robotic vision (pattern recognition, distance detection, and the binding problem [14]) do not impact on NLVR systems because the visual scene is already modeled.

There have been several models of visual perception developed that use 3-D graphics techniques. These models can be classified based on the graphics techniques they use: ray casting and false colouring. [17, 18] implemented a realistic virtual marine world inhabited by autonomous artificial fish. The model used a graphics technique called ray casting to determine if an object met the visibility conditions. Ray casting can be functionally described as drawing an invisible line from one point in a 3-D simulation in a certain direction, and then reporting back all the 3-D object meshes this line

[1] School of Computing, Dublin City University, email: john.kelleher@computing.dcu.ie
[2] School of Computing, Dublin City University
[3] See: http://www.computing.dcu.ie/ jkellehe/SLI.htm

intersected and the coordinates of these intersections. It is widely used in offline rendering of graphics; however it is computationally expensive and for this reason is not used in real-time rendering.

Another graphics based approach to modelling vision was proposed in [12]. This model was used as a navigation system for animated characters. The vision module was comprised of a modified version of the world being fed into the system's graphics engine and scanning the resulting image. In brief, each object in the world is assigned a unique colour or "vision-id" [12, pg. 149]. This colour differs from the normal colours used to render the object in the world; hence the term false colouring. An object's false colour is only used when rendering the object in the visibility image off-screen, and does not affect the renderings of the object seen by the user, which may be multi-coloured and fully textured. Then, at a specified time interval, a model of the character's view of the world, using the false colours, is rendered. Once this rendering is finished, the viewport[4] is copied into a 2-D array along with the z-buffer[5] values. By scanning the array and extracting the pixel colour information, a list of the objects currently visible to the actor can be obtained. [12] used this vision model as part of a navigation system for animated characters. Another navigation behavioral system that used false colouring synthetic vision was proposed by [10]. [13] also used a false-colouring approach to modelling vision, however they integrated their vision model as part of a goal driven memory and attention model which directed the gaze of autonomous virtual humans.

4 The SLI Visual Saliency Algorithm

The basic assumption underpinning the SLI visual saliency algorithm is that an object's prominence in a scene is dependent on both its centrality within the scene and its size. The algorithm is based on the false colouring approach introduced in Section 3. Each object is assigned a unique ID. In the current implementation, the ID number given to an object is simply 1 + the number of elements in the world when the object is created. A colour table is initialised to represent a one-to-one mapping between object IDs and colours. Currently, in the implementation this table contains 256 entries. Although this restricts the number of objects that can be added to the world, this restriction is more a matter of convenience than necessity as the colour table can be extended without affecting the rest of the system. Each frame is rendered twice: firstly using the objects' normal colours, textures and normal shading. This is the version that the user sees. The second rendering is off-screen. This rendering uses the unique false colours for each object and flat shading. The size of the second rendering does not need to match the first. Indeed, scaling the image down increases the speed of the algorithm as it reduces the number of pixels that are scanned. In the SLI system the false colour rendering is 200 x 150 pixels, a size that yields sufficient detail. After each frame is rendered, a bitmap image of the false colour rendering is created. The bitmap image is then scanned and the visual salience information extracted.

To model the size and centrality of the objects in the scene, the SLI system assigns a weighting to each pixel using Equation 1. In this equation, P equals the distance between the pixel being weighted and the centre of the image, and M equals the maximum distance between the centre of the image and the point on the border of the image furthest from the centre; i.e., in a rectangular or square image, M is equal to the distance between the centre of the image and one of the corners of the image.

$$Weighting = 1 - \left(\frac{P}{M+1}\right) \quad (1)$$

This equation normalises the pixel weightings between 0 and 1. The closer a pixel is to the centre of the image, the higher its weighting. After weighting the pixels, the SLI system scans the image and, for each object in the scene, sums the weightings of all pixels that are coloured using that object's unique colour. This algorithm ascribes larger objects a higher saliency than smaller objects since they cover more pixels and objects which are more central to the view will be rated higher than objects at the periphery of the scene as the pixels the former cover will have a higher weighting. This simple algorithm results in a list of the currently visible objects, each with an associated saliency rating.

It is important to note that the scanning process in the SLI visual salience algorithm differs from those in the previous false colour synthetic vision models [12, 10, 13]. The previous false colouring algorithms simply recorded whether the object had been rendered or not. The SLI algorithm records whether an object has been rendered and ascribes each object a relative prominence within the scene. It is this difference that allows the SLI system to rank the objects based on their visual salience. We do not claim that this algorithm accommodates all the perceptual factors that impact on visual salience. However, it does define a reasonable model of visual saliency that operates fast enough for real-time systems.

In the SLI system, we have integrated the information created by this visual salience algorithm with a model of user input discourse. Using this information the SLI system is able to define a local context for the interpretation of deictic reference; i.e., when a reference is made to an object in the visual environment the system is able to restrict the set of objects it considers as candidate referents to those that the are currently in the view frustrum or that the user has seen. A further advantage of this approach is that the visual salience scores associated with the objects in the context model allows the system to adjudicate between candidate referents when resolving some ambiguous references. In Section 5 we will discuss this application of the visual saliency algorithm in more detail.

5 Using Visual Salience to Resolve Ambiguous References

Since Russell [16], there has been a debate concerning the singularity constraint associated with definite descriptions. The singularity constraint is: given the use of a definite description there should be one, and only one, candidate referent in the context of the utterance. An ambiguous or undetermined reference is a reference that breaks the singularity constraint; i.e., there is more than one candidate referent. It has been shown, however, in psycholinguistic experiments that people can easily resolve ambiguous or underdetermined references [2]. "In order to identify the intended referent under these circumstances, subjects rely on perceptual salience as well as on pragmatic assumptions about the speaker's communicative goals" [2, pg. 6].

An advantage of using a visual saliency model as an input to an NLVR system's context model is that the visual salience scores associated with the objects in the context model allows the system, in some instances, to adjudicate between candidate referents when resolving underspecified or linguistically ambiguous references, as illustrated below. Given Figure 1 as the visual context, the referring expression *the house* in *make the house wider*, is an example of an

[4] A viewport is the rectangular area of the display window. It can be conceptualised as a window onto the 3-D simulation.

[5] The z-buffer stores for each pixel in the viewport the depth value of the object rendered at that pixel

ambiguous visible situation use of a definite description. This is because there is more than one object in the context that fulfills the linguistic description of the expression's referent.

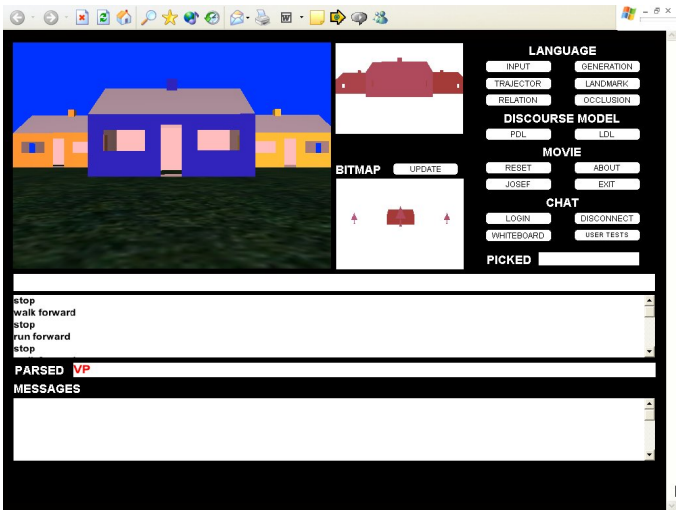


Figure 1. A scene containing three houses.

However, in this instance the SLI system can utilise the visual saliency scores associated with each of the candidates as a probability of the candidate being the referent for the expression. In this case, the SLI system ascribes the blue house in the foreground a normalised computed visual salience of 1.0000 and each of the houses in the background a normalised visual salience of 0.0117. Based on these visual saliency scores, the system decides that the user is referring to the blue house in the foreground and updates the simulation accordingly. Figure 2 illustrates the state of the system after this user input has been interpreted.

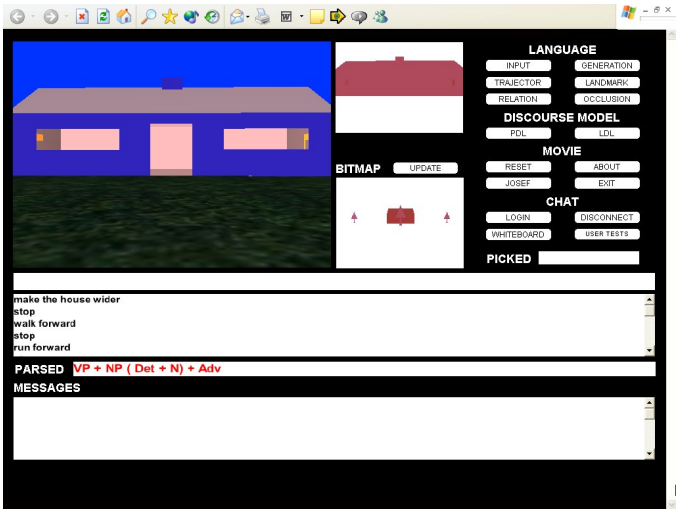


Figure 2. The state of the simulation after the SLI system has interpreted the underdetermined reference *the house* and processed the input *make the house wider*.

Clearly, however, not all ambiguous references can be resolved based on visual salience. In some instances, the difference in the visual saliency scores associated with each of the candidate referents is not sufficient to allow the selection of a referent. Accordingly, as part of the interpretation process for resolving ambiguous references, the SLI system compares the saliency of the primary candidate referent and the other candidates. If the saliency difference does not exceed a predefined confidence interval, the system outputs a message to the user explaining that it is unable to resolve the reference. In SLI scenarios, it is found that when comparing normalised saliency scores, ranging from 0 to 1, a confidence interval of .4 works well. This of course can be adjusted to model a more or less stringent interpretation. Figure 3 illustrates a scene with two houses that have equal visual saliency scores. In this instance, both houses have a visual saliency rating of 1.0000.

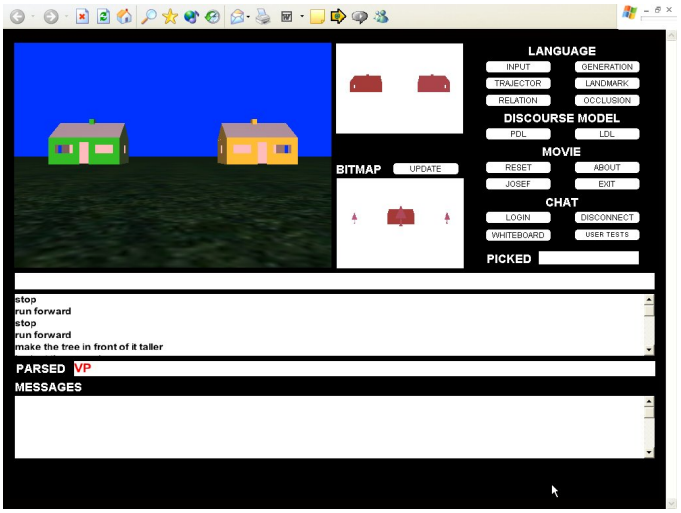


Figure 3. A scene with two houses that have equal visual saliency scores.

Taking Figure 3 as the visual context, if the user inputs an ambiguous referring expression, make the house taller, the system would be unable to resolve the reference. Figure 4 illustrates the state of the system after this command has been interpreted.

Note that in Figure 4 the visual scene has not changed and the message text box contains a message to the user explaining why the system was unable to resolve the reference, as well as listing the candidate referents the system restricted its search to: *Required Saliency Interval Not Reached, Primary Candidates Saliency Confidence Insufficient, I'm not sure which house you mean, I think you mean: house 18 Normalised Salience = 1.0000 or house 17 Normalised Salience = 1.0000.*

6 Conclusions

In this paper, a computational algorithm for modelling the visual salience of objects in the view volume was developed. This model of visual attention is a novel application and extension of a synthetic model of vision that uses a graphics technique called false colouring [12]. In the SLI project, the function of this visual attention model is to try to capture the perceptual information flowing from the visual simulation to the user. For an NLVR system, the advantages of using this visual salience algorithm are that the information created by the algorithm can be used to define a local interpretive context for a given referring expression and the visual salience scores associated with the objects in the context model allows the system, in some

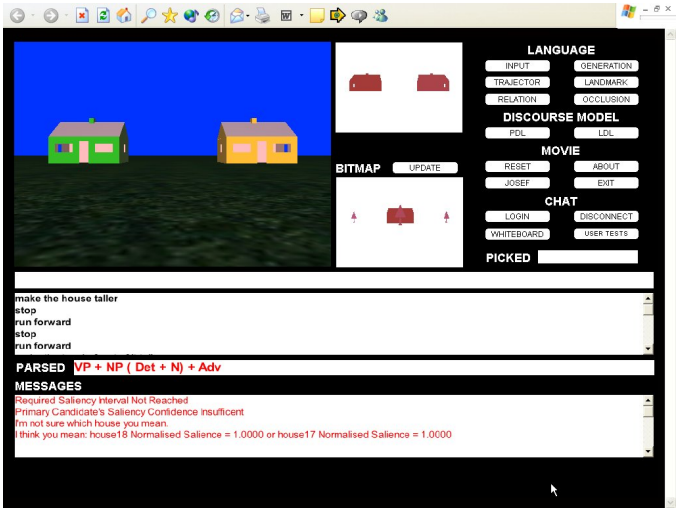


Figure 4. The state of the SLI system after the system has output a message to the user stating that the saliency differences between the candidate referents of an undetermined expression did not permit the system to resolve the reference.

instances, to adjudicate between candidate referents when resolving underspecified or linguistically ambiguous references.

REFERENCES

[1] E. Andre, G. Herzog, and T. Rist, 'On the simultaneous interpretation of real world image sequences and their natural language description: The system soccer', in *In Proceedings of the 8th European Conference on Artificial Intelligence (ECAI-88)*, pp. 449–454. Pitmann, (1988).

[2] I. Duwe and H. Strohner, 'Towards a cognitive model of linguistic reference', Report: 97/1 - Situierte Kunstlicher Kommunikatoren 97/1, Univeristat Bielefeld, (1997).

[3] R.H. Forgus and L.E. Melamed, *Perception A Cognitive Stage Approach*, McGraw-Hill, 1976.

[4] T. Fuhr, G. Socher, C. Scheering, and G. Sagerer, 'A three-dimensional spatial model for the interpretation of image data', in *Representation and Processing of Spatial Expressions*, eds., P. Olivier and K.P. Gapp, 103–118, Lawrence Erlbaum Associates, (1998).

[5] S. J. Goldwater, E.O. Bratt, J.M. Gawron, and J Dowding, 'Building a robust dialogue system with limited data', in *Proceedings of the Workshop on Conversational Systems at the First Meeting of the North American Chapter of the Association of Computational Linguistics*, Seattle, WA, (2000).

[6] G Herzog, 'Connecting vision and natural language systems', Technical Report SFB 314 Project VITRA, Universitt des Saarlandes, (1997).

[7] M.S. Hewitt, *Computational Perceptual Attention*, Ph.D. dissertation, University of Texas, Texas, 2001.

[8] T. Jording and I. Wachsmuth, 'An anthropomorphic agent for the use of spatial language', in *Spatial Language: Cognitive and Computational Aspects*, eds., K.R. Coventry and P. Olivier, 69–86, Kluwer Academic Publishers, Dordrecht, (2002).

[9] S.M. Kosslyn, *Image and Brain*, The MIT Press, 1994.

[10] J. Kuffner and J.C. Latombe, 'Fast synthetic vision, memory, and learning models for virtual humans.', in *Proceedings of Computer Animation Conference (CA-99)*, pp. 118–127, Geneva, Switzerland, (1999). IEEE Computer Society.

[11] F Landragin, N Bellalem, and L Romary, 'Visual salience and perceptual grouping in multimodal interactivity', in *Proceeding of the International Workshop on Information Presentation and Natural Multimodal Dialogue (IPNMD)*, Verona, Italy, (2001).

[12] H. Noser, O. Renault, D. Thalmann, and N Magnenat-Thalmann, 'Navigation for digital actors based on synthetic vision, memory and learning', *Computer Graphics*, **19**(1), 7–9, (1995).

[13] C. Peter and C. O'Sullivan, 'A memory model for autonomous virtual humans', in *Proceedings of Europraphics Irish Chapter Workshop (EGIreland-02)*, pp. 21–26, Dublin, (2002).

[14] O Renault, N Magnenat-Thalmann, and D Thalmann, 'A vision-based approach to behavioural animation', *Visualization and Computer Animation*, **1**(1), 18–21, (1990).

[15] J. Reynolds, 'Visual salience, competition, neuronal responce synchrony and selective attention', in *Sloan/Swartz Centers for Theoretical Neurobiology Annual Summer meeting*. The Swartz Foundation, (2001).

[16] B. Russell, 'On denoting', *Mind*, **14**, 479–493, (1905). Reprinted Logix and Knowledge (1956), pp. 39-56, R.C. Marsh ed.

[17] X Tu and D. Terzopoulos, 'Artificial fishes: Physics, locomotion, perception, behaviour', in *Proceedings of ACM SIGGRAPH*, pp. 43–50, Orlando, FL, (1994).

[18] X. Tu and D. Terzopoulos, 'Perceptual modelling for behavioural animation of fishes', in *Proceedings of the Second Pacific Conference on Computer Graphics and Applications*, pp. 185–200, Beijing, China, (1994).

[19] T. Winograd, 'A procedural model of language understanding', in *Computer Models of Thought and Language*, eds., R.C. Schank and K.M. Colby, 152–186, W. H. Freeman and Company, (1973).

Personalised Web Navigation using Combination Filtering

Liadh Kelly[1] and **John Dunnion**[2]

Abstract. It is becoming increasingly difficult for a user to find the information they are looking for when searching the web. A great deal of artificial intelligence research is dedicated to alleviating this problem using ideas and techniques from user modelling and information filtering. In this work, we present INVAID, a personalised navigational aid that integrates these approaches in order to track a user's navigation on the web and subsequently recommend pages of potential interest to the user. The novel approach to collaborative filtering used in INVAID is highlighted, along with the 'live' and database driven experimental results, which validate the approach but also show the need for further refinement of the system.

1 Introduction

Web users are currently faced with a significant information overload problem. A great deal of research has looked at developing tools to help users locate the information they are looking for on the web. Examples here include *Personal WebWatcher*[16], *Letizia*[10], *GroupLens*[8]. Recent advances in artificial intelligence, user modelling, and information filtering promise to provide practical and effective solutions to this information overload problem.

In relation to user modelling, a model of a user's areas of interest can be constructed from previous interactions with the navigational aid system. Various techniques can be used for information filtering. Two such approaches are *collaborative filtering* and *content-based filtering*. Collaborative filtering organises users with similar interests into peer groups, thus enabling the recommendation of documents viewed as interesting by some of the peer group to other members of that group. Content-based filtering, on the other hand, compares the contents of a web page with a user profile and selects documents whose contents match the user profile.

In this paper we describe the technology behind the INVAID (Intelligent NaVigational AID for the world wide web) system, which was created to address the problem faced by users in finding relevant information on the web. Very briefly, INVAID combines a standard weighted feature list content-based filtering module with a novel collaborative filtering module to suggest web pages of potential interest to individual users. Combining content-based filtering and collaborative filtering in a system allows for the shortcomings of one filtering technique to be over come by the other [2, 18].

INVAID does not attempt to solve all problems associated with web navigation and certainly not for the entire vast web. Rather the goal is to track users on a given relatively static web site, by noting the pages the user visits and the text contained within these pages, and to subsequently highlight relevant unvisited pages.

The remainder of this paper is organised as follows. Section 2 describes the INVAID system design, focusing on the user modelling and information filtering components. Section 3 presents the experimental evaluation of the system, and finally in section 4 we discuss issues relating to future research on this project before concluding.

2 System Design

INVAID is a personalised navigational aid that tracks a user's navigation on the web and subsequently recommends pages of potential interest to the user. The full design [7] of the INVAID system is beyond the scope of this paper. This paper focuses on the user modelling and information filtering components of INVAID, see Figure 1. The user modelling component consists of two modules, each of which provide user data to the appropriate information filtering module. The information filtering modules use this data to form web page suggestions for an individual.

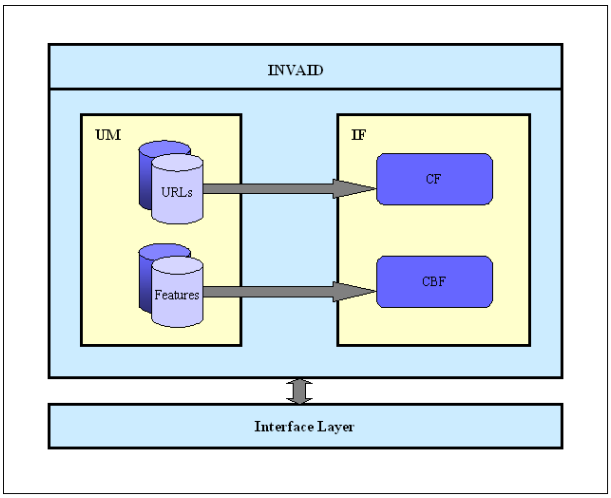


Figure 1. System Design; UM=User Modelling, IF=Information Filtering, CBF=Content-based Filtering, CF=Collaborative Filtering.

2.1 User Modelling

A user modelling system models a user in some way, using various techniques to 'learn' relevant information about a user. Different ap-

[1] Athlone Institute of Technology, Dublin Road, Athlone, Co. Westmeath, Ireland
[2] Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland

proaches are taken to user modelling depending on the individual requirements of the system, examples include [3, 13].

The user modelling component in INVAID works online in the system's proxy server, and for each user two user profiles are maintained, see Figure 1. A URL list user profile is built up from web requests made by the user. A feature list user profile is built up from the web pages viewed by the user.

2.1.1 Feature List Profile

This profile adopts a simple weighted feature list methodology. For every page a user visits, the page is represented as a list of weighted features. Then for every word already in the user's list the current weight is augmented to include the weight of the word in the page currently being viewed by the user. Words not already in the user's list are added to this list with the weight they hold in the page the user is currently viewing.

Luhn [12] suggests that the highest discrimination power is associated with the middle frequency of occurrence terms. The INVAID system adopts this approach, thus maintaining the n middle frequency of occurrence keywords in a user's profile and removing the keywords at either extreme.

2.1.2 Visited URLs List Profile

Each URL visited by the user is stored along with associated information for the URL in order to establish weights of interest for the user in the URL. For each URL visited by a user the following information is stored:

1. The actual URL.
2. The method by which the URL was accessed, either by explicit entry of URL or by means of following a hyperlink (in which case the URL that this hyperlink came from is stored).
3. The most recent date of visitation of the URL by the user.
4. The number of times the user has previously visited the URL.
5. A user rating for the URL, if any was given.

The zero-input information from 2, 3 and 4 above is combined to obtain an implicit relevance feedback rating representing the user's interest in the URL. The only form of explicit relevance feedback for a URL is the user rating, listed as item point 5 above. In our system we are aware that users will not always give ratings for pages they visit. With this in mind our primary focus is to obtain implicit relevance feedback from the user. The formula concentrates on implicit feedback to derive a weight of interest for a user on a URL and incorporates the explicit user rating where such a rating has been given. Little work has been done on automatic collaborative filtering using implicit ratings although one technique that has been considered for supporting implicit ratings is to map implicit measures such as time-spent-reading into an explicit rating scale and then use previously proved collaborative filtering algorithms [5, 6, 17]. The INVAID system carries out such a conversion of implicit ratings into explicit ratings, using Equation 1, where $W_{URL,user}$ is the overall relevance weighting of a URL for a given user (defined such that the maximum weight for a URL is 1), and where nw = number of visits weight; ni = number of times the user visited the URL; $nmax$ = maximum number of times any URL was visited by the user; dw = date weight; lt = age of oldest URL in the user profile; di = age of the URL; aw = access method weight; ai = 1 if direct access, 0 otherwise; ew = explicit relevance feedback weight; ri = user rating for the URL (in the absence of a user rating, a default rating is given); α+β+γ=1 (the values for α,β,γ can be set to give different importance to the implicit values); δ = max rating (in this system, this is currently set to be 5).

$$W_{URL,user} = ew(nw + dw + aw) \tag{1}$$

$$nw = \alpha(\frac{ni}{nmax}) \tag{2}$$

$$dw = \beta(\frac{lt - di}{lt}) \tag{3}$$

$$aw = \gamma(ai) \tag{4}$$

$$ew = (\frac{ri}{\delta}) \tag{5}$$

In this work we make the assumption that web pages the user has most recently looked at represent the user's current interests more accurately than other web pages in their history profile. Therefore, when the user profile reaches saturation point, all but the n most recently visited URLs are removed from their profile.

2.2 Information Filtering

Information filtering systems are designed to examine a stream of dynamically generated articles and display only those which are relevant to a user's interests.

The information filtering component in INVAID operates off-line, on a nightly basis. In this way the filtering system receives daily updates to user profiles. This component consists of two modules, the content-based filtering module and the collaborative filtering module, see Figure 1. Each component separately compiles a list of URLs to suggest to a user, these lists are then combined to arrive at the final suggestion list for the user.

2.2.1 Content-based Filtering

The content-based filtering system uses a simple weighted feature list representation of all web pages in the site and compares this to a user's feature list profile to arrive at a list of web pages to suggest to a user. The similarity between a page and user profile is measured using the overlap coefficient shown by Equation 6 [19], where X is the user's feature list profile and Y is the feature list representation of a web page.

$$sim(X, Y) = \frac{\mid X \cap Y \mid}{Min(\mid X \mid, \mid Y \mid)} \tag{6}$$

2.2.2 Collaborative Filtering

We implemented a novel collaborative filtering module, with the assumption that all users of a peer group should have a certain amount of commonality for the group as a whole to be of maximum benefit to its members (The rationale behind this choice of peer group formation is discussed at length in [7]). All possible peer groups, of maximum size, are thus created using the algorithm given in Figure 2.

$$w(a, i) = \frac{\Sigma_j(v_{a,j} - \bar{v}_a)(v_{i,j} - \bar{v}_i)}{\sqrt{\Sigma_j(v_{a,j} - \bar{v}_a)^2 \Sigma_j(v_{i,j} - \bar{v}_i)^2}} \tag{7}$$

```
1.  define PeerGroupFormulation (Users U, PeerGroups C, Threshold t)
2.  begin
3.     C := user1 in U
4.     for each user a in U
5.         for each Peergroup PG in C
6.              newPG ← MakeNewPG(PG, a, t)
7.         endFor
8.         C := C + newPG
9.         C ← UpdatePeerGroups(C)
10.    endFor
11.    return C
12.  end

13. define MakeNewPG(PG, a, t)
14. begin
15.    newPG := {}
16.    for all user i in PG
17.        if similiarity(a,i) > t
18.           newPG := newPG + i
19.    endFor
20.    newPG := newPG + a
21.    return newPG
22. end

23. define UpdatePeerGroups(C)
24. begin
25.    C := C - duplicate PGs
26.    C := C - subset PGs
27.    return C
28. end
```

Figure 2. Peer group formulation algorithm.

In line 17 of the algorithm in Figure 2 the weight of similarity, $w(a,i)$, is calculated using the Pearson correlation coefficient (see Equation 7), where $w(a,i) =$ the weight of similarity between user a and each user, $\bar{v}_i$ = mean vote for user (see Equation 8),

$$\bar{v}_i = \frac{1}{I_i} \sum_{j \in I_i} v_{i,j} \tag{8}$$

I_i = set of items on which user has voted, and $v_{i,j}$ = weight associated with user i for URL j. This metric was chosen as a substantial amount of research has found it performed reasonably well and in a lot of instances out-performed other methods.

When determining which URLs to suggest to a user, the URLs in the peer groups, of which the active user is a member, are consulted and a predicted weight of interest for the user in the URLs, she has not viewed, is computed using Equation 9, where $p_{a,j}$, the predicted vote of the active user for item j, is a weighted sum of the votes of the other users. The URLs with the highest predicted weight of interest are then chosen for suggestion to the user.

$$p_{a,j} = \bar{v}_a + \kappa \sum_{i=1}^{n} w(a,i)(v_{i,j} - \bar{v}_i) \tag{9}$$

3 Experimental Evaluation

Experiments were carried out to test the effectiveness of the user modelling and information filtering components of INVAID. The experimentational process consists of two parts. In part one the system went 'live' for a limited period to address the overall workings of the system. Part two of the experiments used a readily available testbed [1] to address the novel URL weight assignment and peergroup formation methods of the collaborative filtering component.

3.1 Trial One : 'Live Experiments'

A small group of 20 users used the system for a trial period of two weeks on the 'Lonely Planet' [11] web site. At the time of experimentation the 'Lonely Planet' site was reasonably large, containing almost 8000 URLs, of which 5000 are (text-based) HTML pages. Although the site also contained over 2500 external links, for the 'live' experiments the stipulation was made for users to remain within the 'Lonely Planet' web site itself. In cases where users followed a link out of the 'Lonely Planet' web site, all such external navigation was ignored by the INVAID system. (Statistics on the 'Lonely Planet' site were generated by Custo application [4].)

3.1.1 Testing Method

When each person logged on to the INVAID system they were presented with a personalised list of six web pages to grade from 0 to 5, reflecting their interest in each. Users then proceeded to browse the 'Lonely Planet' web site, grading pages from 0 to 5 if they so desired (each day an average of 50% of users graded pages). As the users browsed the site, the system was able to build up profiles for each user.

3.1.2 Results

The average daily user rating for week 1 and week 2 of the experiment can be seen in Figures 3 and 4 respectively. While these results did not prove to be as positive as expected, the general user feedback from using the system was positive. This positive feedback shows that people want help in navigating the web and that they are very receptive to it, despite the fact that the help might be limited or not always what the person is looking for.

An average user rating of 2.3 was obtained for the first week's ratings. The suggestions for this week, which began on day 3, were gathered from the collaborative filtering system only. This is a positive average rating result given the earlier rater problem associated with collaborative filtering, and the limited number of people that were using the system. The average user rating for the second week was somewhat lower at 2.0. The pages suggested to the user for the second week were obtained from both the content-based and collaborative filtering systems. This lower average suggests that the content-based filtering component is the weaker element of the system.

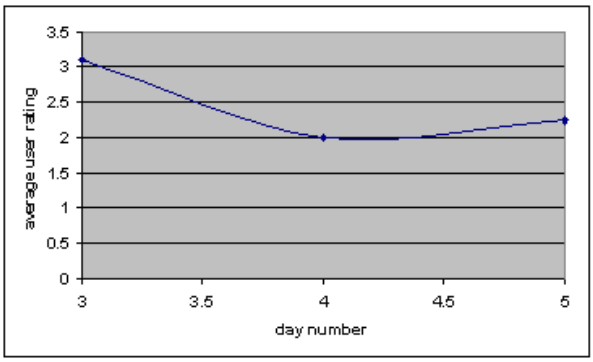


Figure 3. Week 1 average daily user ratings.

The average user rating over the two week period is only 2.0, which at first glance seems very low. However, a number of factors can contribute to this rather low average. First of all, the system was only tested over a short period of time. This meant that user profiles didn't have time to become very established. On top of this the small

numbers of users testing the system, meant that the pool of web pages viewed by people was small. These facts go some way to explain the higher quality of suggestions made by the collaborative filtering system at the beginning of the experiments. After the initial suggestions were made and viewed by the test users, individual profiles did not increase significantly enough as to greatly increase the pool of pages viewed by people. This resulted in a lower quality of suggestions being made by the collaborative filtering system on subsequent days. To exacerbate this problem even further, a lot of people did not use the system on a very regular basis. This, added to the problem of the small pool of web pages, in the sense that maximum benefit could not be obtained from the collaborative filtering system.

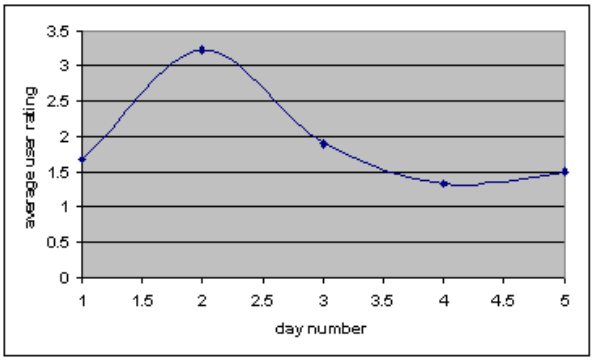


Figure 4. Week 2 average daily user ratings.

The content-based filtering module would have been expected to perform better. In the current implementation, the top n most interesting pages in the possible suggestion list might not necessarily be presented to a user. It is probable that this affected the results generated by the content-based filtering system, as higher quality suggestions could have been made if the top n ranked pages were presented to users, instead of suggesting pages at random. With content-based filtering, there is the opportunity for new pages to be added to the pool of pages in the peer groups of the collaborative filtering system. However, if the quality of the pages suggested by the content-based filtering system isn't high, the full benefit of using the content-based system is not exploited.

3.2 Trial Two: Using EachMovie Test-Bed

Collaborative filtering is the more novel of the two filtering module in the system. With this in mind, further experiments were carried out on the collaborative filtering module using a larger, readily available DEC SRC's EachMovie dataset [1]. The EachMovie dataset contains a total of 2811983 numeric ratings for 1628 different movies, entered by 72916 users over an 18 month period.

A subset of 800 users of the EachMovie data set was used for these experiments. For each user in this subset their profiles were split into a training set and a test set, in the ratio 6:4. The test set comprised the most recently accessed pages of a user, and the training set the remaining pages. It was found that many of the profiles in this subset were very small. Subsequently, a further experiment was carried out using a subset of 500 users of the EachMovie data set. This subset held an extra stipulation that the profiles in the training set had to be of size 20 or greater.

3.2.1 Testing Method

The INVAID collaborative filtering algorithm was run using each of the training sets in turn. As described earlier in this paper, the weight of importance associated with each URL in a user profile is calculated from the user rating, the method of URL access, the number of visits to the URL and the most recent date of visitation. The EachMovie data set does not record the method of URL access and the number of visits. These two values are set to 0 when calculating the relevance weights for URLs in these experiments. Peer groups using each of the training sets were formed. For the two separate sets of experiments involving the training sets of 800 users and 500 users respectively, ordered suggestion lists were then compiled for each user. These ordered suggestion lists were then compared to the users' test sets. Comparisons were carried out using precision, recall and the f-measure. The f-measure proposed by Lewis and Gale [9], which gives equal importance to precision and recall, was used here and is defined by Equation 10.

$$f - measure = \frac{2.precision.recall}{precision + recall} \tag{10}$$

Precision, recall and the f-measure are calculated for increments of 5 suggestions, i.e. they are calculated for 5 suggestions, 10 suggestions, etc.

3.2.2 Results for Experimental Set 1: 800 Users

In Figure 5 it can be seen that as the number of suggestions increases the precision decreases, as would be expected. Starting out at a low of just over 1%, recall steadily increases to obtain an eventual maximum of 100%, Figure 6. Positive results are expected from the f-measure however the accuracy here does not go over 13% until the suggestion list contains 125 elements, see Figure 7.

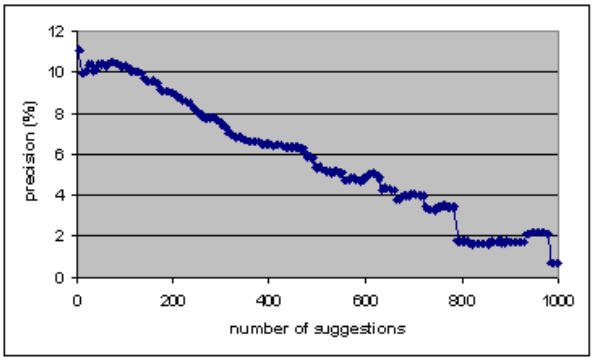


Figure 5. Graph showing number of suggestions vs precision.

3.2.3 Results for Experiment Set 2: 500 Users

The second set of experiments ran over 500 users showed some improvement in results by only using profiles above a certain minimum size. This improvement was obtained despite the fact that the number of users was significantly reduced, from 800 users to 500 users. The results obtained here are depicted in Figures 8, 9, and 10.

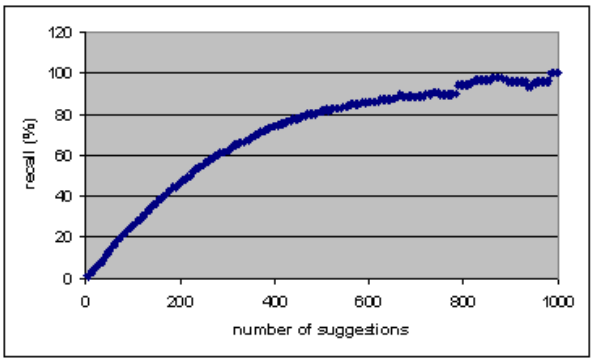


Figure 6. Graph showing number of suggestions vs recall.

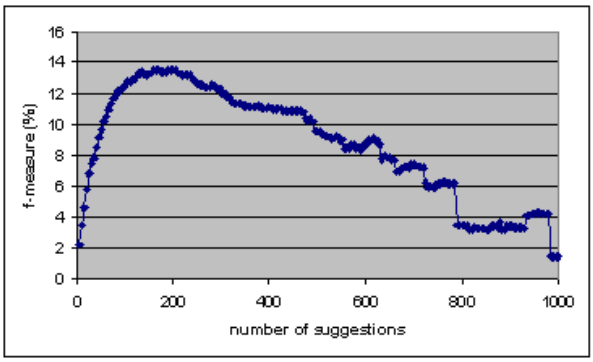


Figure 7. Graph showing number of suggestions vs f-measure.

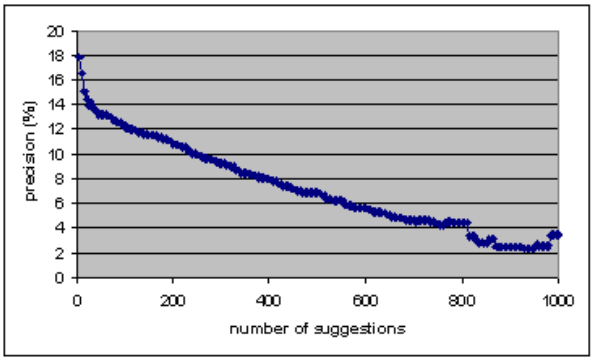


Figure 8. Graph showing number of suggestions vs precision.

Figure 8 illustrates that at as few as 5 suggestions precision was 18% and did not fall below 10% until well over 200 suggestions were made. This compares favourably with the results obtained in the previous run of this experiment, where precision was 11% for 5 suggestions. Recall also improved slightly in this experiment, see Figure 9, reaching 100% with marginally fewer suggestions than in the previous run of the experiment. However, for 5 suggestions it would have been hoped to notice more than the marginal improvement that was obtained for recall. More notable however, is the improvement in the f-measure, Figure 10, where results above 16% were obtained at under 200 suggestions, compared to maximum results below 14% in the previous run of the experiment. Also at 1000 suggestions precision and the f-measure were higher for this run of the experiments, where precision was 4% and the f-measure was above 6%. At the same time though recall had fallen below 100% for 1000 suggestions.

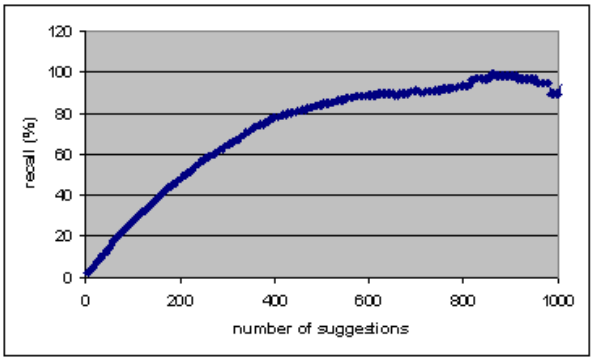


Figure 9. Graph showing number of suggestions vs recall.

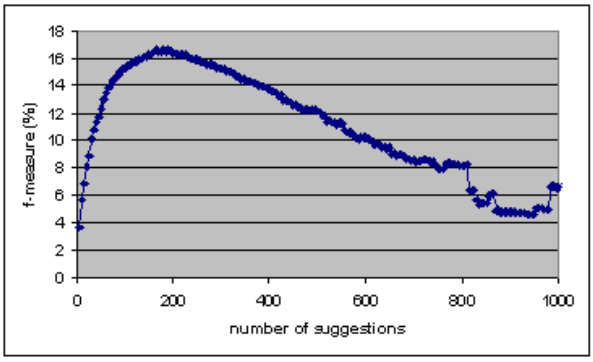


Figure 10. Graph showing number of suggestions vs f-measure.

4 Conclusions and Future Work

The ever-increasing vastness of the web makes personalisation increasingly important for the web. This paper presents a combined content-based and collaborative filtering Web page suggestion method.

The INVAID system is still in its infancy, there is great scope for further development of the system. For example, the suggestion process currently works off-line, investigation should be carried out into the practicalities of carrying out on-line immediate suggestions as the user navigates. Also, to date the backend system of INVAID has been of primary importance, a dynamic interface to the system that works with the user should also be created. Additionally, much more extensive experimentation is required. A problem we encountered during experimentation was the unavailability of sufficiently large suitable datasets. While other methods of testing are possible there is clearly a need for more large datasets to be made available for testing in the combination filtering area.

From our 'live' evaluation studies we found that people respond positively to navigational assistance, despite the fact that the average user grading of URLs was relatively low. This suggests that there is great need for systems such as INVAID, however as already mentioned in the case of INVAID further system refinement is required. The low average user rating was caused by a number of factors, as discussed in the paper. With further investigation into these factors and more extensive 'live' evaluation, the results should be improved.

Certain levels of success were found in the testbed driven evaluation, which tested the effectiveness of the collaborative filtering module. Only certain elements of our collaborative filtering algorithm could be used in this evaluation, due to the testbed used, thus restricting its effectiveness. Further *tweaking* and experimentation should be carried out to the algorithm to determine the optimal weights for implicit feedback factors, to establish if the Pearson coefficient is the optimal similarity measure for the system and to refine the algorithm.

Further refinements of the INVAID system are also possible. When suggesting pages the user's current context in the web could be taken into consideration. More weight could be given to pages recently visited when formulating lists of pages to suggest to the user. In this way a user's current interests or search goals could be more accurately suggested, but at the same time still taking their overall interests into consideration by use of their user profiles.

It would then be desirable to once more run the experiments shown in this paper. Additional experimentation should also be carried out in the testing of the system, such as those carried out in [14, 15]. In future experiments the INVAID collaborative filtering method will be compared with existing collaborative filtering method, such as the nearest neighbour technique.

Moving forward this combination filtering technique has the potential to aptly suggest web pages of interest to individuals and impact on the development of increasing effective filtering systems.

ACKNOWLEDGEMENTS

This work was funded by the Advanced Software Technologies Initiative (ASTI), which was funded by Enterprise Ireland's Software Programme in Advanced Technology (PAT) and the National Software Directorate. Thanks are also due to Compaq Equipment Corporation who provided the test bed for the Trial 2 experiments.

REFERENCES

[1] EachMovie data set, System Research Center, Digital Equipment Corporation, 1997. http://www.research.compaq.com/SRC/eachmovie/.

[2] M. Balabanovic and Y. Shoham, 'FAB: Content-Based Collaborative Recommender.', *Communications of the ACM*, **40(3)**, 66–72, (1997).

[3] L. Chittaro and R. Ranon, 'Adding Adaptive Features to Virtual Reality Interfaces for E-Commerce', in *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, eds., O. Stock P. Brusilovsky and C. Strapparava, pp. 86–97. Springer-Verlag, (2000).

[4] http://www.netwu.com/.

[5] B.J. Dahlen, J.A. Konstan, J.L. Herlocker, N. Good, A. Borchers, and J. Riedl, 'Jump-starting movieLens: User benefits of starting a collaborative filtering system with "dead-data"', Technical Report TR 98-017, University of Minnesota, (1998).

[6] J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl, 'An Algorithmic Framework for Performing Collaborative Filtering', in *Proceedings of 1999 Conference on Research and Development in Information Retrieval*, (1999).

[7] L. Kelly, *The Design and Implementation of an Intelligent Personalised Navigational Aid for the World Wide Web*, Master's thesis, University College Dublin, Belfield, Dublin 4, Ireland, 2001.

[8] J.A. Konstan, B.N Miller, D. Maltz, J.L. Herlocker, L.R. Gorgan, and J. Riedl, 'GroupLens: Applying collaborative filtering to Usenet news', *Communications of the ACM*, **40(3)**, 77–87, (1997).

[9] D. Lewis and W.A. Gale, 'A sequential algorithm for training text classifiers', in *Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pp. 3–12. Springer-Verlag, (1994).

[10] H. Lieberman, 'Letizia: An Agent That Assists Web Browsing', in *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI'95*, ed., C. Mellish, pp. 924–929. Morgan Kaufman Publishers, (1995). Montreal, Canada.

[11] http://www.lonelyplanet.com/.

[12] H.P. Luhn, 'The automatic creation of literature abstracts', *IBM Journal of Research and Development*, **2**, 159–165, (1958).

[13] P.P. Maglio and R. Barrett, 'How to Build Modeling Agents to Support Web Searchers', in *Proceedings of the Sixth International Conference on User Modeling*, pp. 5–16, (1997).

[14] P. Melville, R.J. Mooney, and R. Nagarajan, 'Content-Boosted Collaborative Filtering', in *Proceedings of the SIGIR-2001 Workshop on Recommender Systems*, (2001). New Orleans, LA.

[15] O. Meshar. MovieMatch - a collaborative filtering system for movie recommendations, 2002. http://citeseer.nj.nec.com/meshar02moviematch.html.

[16] D. Mladenic, 'Personal WebWatcher: Implementation and Design', Technical Report IJS-DP-7472, Department for Intelligent Systems, J. Stefan Institute, (1996).

[17] R. Rafter, K. Bradley, and B. Smyth, 'Automatic Collaborative Filtering Applications for Online Recruitment Services', in *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, eds., O. Stock P. Brusilovsky and C. Strapparava, pp. 363–368. Springer-Verlag, (2000).

[18] B. Smyth and P. Cotter, 'A Personalized TV Listings Service for the Digital TV Age', *Journal of Knowledge-Based Systems*, **13(2-3)**, 53–59, (2000).

[19] C.J. van Rijsbergen. Information Retrieval. http://www.dcs.gla.ac.uk/Keith/Preface.html.

Examining the Scalability of K-means on High Dimensional Datasets

Sharon Kennedy and **Dr. M-Tahar Kechadi**[1]

Abstract. The capacity of digital data storage has doubled every nine months during the last ten years. This data explosion is at twice the rate predicted by Moore's Law for the growth of computing power for the same time frame. This has raised many data mining and analysis issues and it means that our ability to store data surpasses our ability to process it. With this in mind, our research is aimed at investigating new approaches to data mining, in particular clustering, and investigating the scalability problem with respect to existing algorithms for use with highly multi-dimensional large datasets. Within this paper, the initial phase of our research is discussed. Here we examine the scalability of three of the several data mining techniques that we have worked with. These are k-means, and two different hierarchical clustering methods, utilising the cosine similarity function and Pearson's correlation co-efficient. We compare each of these methods and report on the scalability of each. The preliminary work that has been undertaken is presented here and the results prove to be encouraging.

1 INTRODUCTION

Data mining techniques allow discovery of knowledge from massive amounts of data. When we consider a data dependent company or organisation the importance of research becomes critical. Clearly efficient and accurate data mining solutions are imperative. One of the most common techniques used to manipulate these datasets is clustering [10, 11]. The idea is to divide a large collection of objects into sets having some common characteristics. This method of representing a dataset achieves an overall simplification of the data, thereby enabling the discovery of hidden patterns and relationships within the dataset. Many techniques and algorithms were proposed in the literature [13] to cluster data of different types. Usually these techniques perform reasonably well on small datasets sizes. However, most of them do not scale well as these datasets become very large. This problem is crucial as, nowadays, large and growing datasets are commonplace within the scientific community, governments and business organisations. For example, NASA's Earth Observing System (EOS), collates one terabyte of data from orbiting satellites and other stellar instruments every day [12]. Prior to the digital data revolution much of the world's data was kept as hard copy. This presented obvious analysis drawbacks. However, without efficient and effective algorithms that can be scaled to the terabyte level, all of this digital data available will remain as unexploited as before.

Previous research efforts on scaling data mining algorithms to accommodate vast datasets have taken several avenues of approach. General scaling principles include the use of summary statistics, data compression, heuristic state space pruning and incremental computation [6]. These principles have contributed to addressing the scaling problem, however, as datasets continue to grow, when a sequential data-mining algorithm cannot be further optimised or when the fastest serial machine available cannot deliver results in a reasonable time, new solutions must be investigated. It is our aim in this research to find new ways of scaling traditional serial data mining algorithms to cope with very large datasets and thus to further contribute to this area of research.

[1] Department of Computer Science, University College Dublin, Belfield, Dublin 4.

2 PROBLEM DESCRIPTION

In order to address this problem we need to work with a suitably large dataset to investigate the scalability of clustering algorithms. Simple, linear data, although easier to work with, is not typically the type of data that requires analysis so, with this in mind, a highly multi-dimensional dataset was chosen.

2.1 The Corpus

One such dataset is a textual corpus called the Topic Detection and Tracking (TDT2) corpus [2]. One of the main contributions of the TDT Pilot study was the production of the TDT Study Corpus. The Linguistic Data Consortiumcreated the corpus to support the research of their projects and other data mining initiatives.

The TDT2 English Corpus has been designed to include six months of material drawn on a daily basis from six English news sources. The period of time covered is from January 4 to June 30, 1998. The six sources are the New York Times News Service, the Associated Press Worldstream News Service, CNN "Headline News", ABC "World News Tonight", Public Radio International's "The World", and the Voice of America. It contains approximately twenty seven thousand stories with a feature set over sixty six thousand. This dataset is therefore sufficiently complex for our research and is used in the experiments described later.

2.2 Data Mining Techniques

The system (K)luster, uses the algorithms to cluster together documents from our highly multi-dimensional text dataset. The system is composed of several sub-processes, each of which has a specific task. The stages of data development are illustrated in Figure 2.1.

Pre-processing

In order to cluster effectively, the data must be reduced to an appropriate level of complexity so that it can be readily processed. As discussed previously, the documents were taken from either Reuters or a manual transcription of the broadcast news speech from CNN. Both of these sources are therefore, a narrative. For comparison purposes, a narrative would be extremely difficult, if not impossible, to work with. For clustering, it is the terms within a document and the frequencies of those terms that contain the most information about the content of the story. The task was then to reduce the stories to data files containing term frequency pairs. During

this process, stop words, i.e. prepositions, conjunctions and commonly occurring words were eliminated using a stop word list. This has the effect of reducing the noise within the dataset and thereby improving the overall accuracy of the results. Terms were also stemmed using Porter's Stemming algorithm [5]. This pre-processed data is now in the form of a file containing, for each document, a list of the terms within it accompanied by the frequency with which they occur in the document. Conceptually, this is a sparse feature representation of the data. This means that only when a feature, i.e. a term, is present in a document, is it included in the representation of that document. If a feature is not present, it is not included in that document representation. If we had decided on a dense feature representation, every feature would be included in every document representation and a frequency of zero would be associated with a feature that does not occur in a given document. Our dataset benefits from a sparse representation because of its large feature set. This pre-processed data was then fed into the clustering engine, (K)luster.

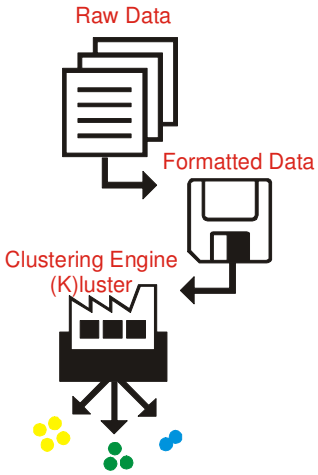


Figure 2.1. The stages of data preparation and clustering in (K)luster.

In this paper we present three of the different data mining techniques that were used in the system. Each algorithm was used to cluster the dataset and the scalability, accuracy and efficiency of each was compared. The algorithms used were k-means and two different hierarchical clustering methods, utilising the cosine similarity function and Pearson's correlation co-efficient.

2.3 Hierarchical Clustering

Hierarchical clustering does not partition each data point into a particular cluster in a single step. Instead, a series of partitions take place, which may run from a single cluster containing all objects to *n* clusters each containing a single object. Hierarchical Clustering is subdivided into agglomerative methods, which proceed by a series of fusions of the *n* objects into groups, and divisive methods, that separate a group of *n* objects. Agglomerative techniques are more commonly used, and this is the method implemented here. An agglomerative hierarchical clustering procedure produces a series of partitions of the data, P_n, P_{n-1}, ..., P_1. The first P_n consists of *n* single object, the last P_1, consists of single group containing all *n* cases. At each particular stage the method joins together the two clusters which are closest together or most similar. The similarity score of two clusters is determined by two different methods, the cosine similarity and Pearson's product moment correlation coefficient.

Cosine Similarity

The cosine similarity measure is based on the inner or dot product of two vectors and can be formulated as follows:

$$\text{cosine\_sim} = \frac{A \cdot B}{|A||B|}$$

$$\text{cosine\_sim} = \frac{\sum_{i=1}^{N} w_t, a_i \cdot w_t, b_i}{\sqrt{\sum_{i=1}^{N} w_t^2, a_i} \cdot \sqrt{\sum_{i=1}^{N} w_t^2, b_i}}$$

where, *A* and *B* are the two vectors used to represent the documents , wt,a_1 is the weighting of the term a_1 in document *A* and wt,b_2 is the weighting of the term b_2 in document *B* and *N* is the total number of terms in the document. The weightings used in our system are the frequencies of the terms. This measure becomes 1 if the documents are identical, and 0 if there is nothing in common between them i.e., if the documents are identical then the angle between them will be 0, the cosine of which is 1. If the documents have nothing in common the vectors will be orthogonal to each other and have an angle of 90 degrees, the cosine of which is 0.

Pearson's Correlation Coefficient

The correlation between two variables reflects the degree to which the variables are related. The most common measure of correlation is the Pearson Product Moment Correlation [16].

$$r = \frac{\sum_{i=1}^{N} ((x_i - \bar{x})(y_i - \bar{y}))}{\sqrt{\sum_{i=1}^{N} (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^{N} (y_i - \bar{y})^2}}$$

where *r* is Pearson's correlation value and reflects the degree of linear relationship between two variables. It ranges from +1 to -1. A correlation of +1 means that there is a perfect positive linear relationship between variables. N is the number of terms in a document, x_i is the frequency of the matching term in the document, which is being clustered, $\bar{x}$ is the average frequency in the document vector being clustered. y_i is the frequency of the matching term in the cluster to which the document is being correlated to, and $\bar{y}$ is the average frequency in the cluster vector.

2.4 K-Means

The third algorithm used in the system was the k-means algorithm [1, 3] and is by far the most popular clustering tool used in scientific and industrial applications. The name comes from representing each k cluster by the mean of its members. This mean structure is referred to as the cluster centroid and is usually incrementally updated, as new members are absorbed into the cluster. A data point is added to the cluster that achieves the maximum similarity score, if this score is higher than the set threshold value. The similarity score is calculated as the Normalised Euclidean distance between the data point and the cluster centroid, and is defined as follows:

$$dist = \frac{\sqrt{\sum_{i=0}^{N} \delta_i \left(freq\ (x_i) - freq\ (\mu_i) \right)^2}}{N}$$

where N is the number of objects in the data point, the number of terms in a document in this case. x_i is term i in the current document. μ_i is term i in the current cluster centroid. $freq(x_i)$ is the frequency of x_i and $freq(\mu_i)$ is the frequency of μ_i. The purpose of the denominator is to normalise the distance figure. The additional value δ_i, has the affect of including the frequencies of matching terms and discarding the frequencies of terms that don't match as stated below.

$$\delta_i = \begin{cases} 1 \text{ if } x_i = \mu_i \\ 0 \text{ otherwise} \end{cases}$$

Initial Cluster Formation

The k-means algorithm dictates that a number of initial clusters are formed. It is known that iterative clustering techniques are especially sensitive to initial starting conditions, and so a question that plagues all hill-climbing procedures is the choice of the starting point. Unfortunately, there is no simple, universally good solution to this problem. A recursive method for initializing the means by running K clustering problems is mentioned in [8]. A variant of this method consists of taking the mean of the entire data and then randomly perturbing it K times [9]. This method, however, does not appear to be better than random initialization. This seems to be the case for many attempts at creating good initial clusters. At this early stage we have, therefore decided to adopt the random approach that has proven itself to be no worse than many heuristic methods.

Cluster Mean

Each cluster object contains a mean or centroid, which is a structure within the cluster that represents all the members of the cluster at a particular time. It is on this mean that entry of new members into the cluster is based. As mentioned previously, terms that appear more frequently are more significant to the document's content, so the mean of a cluster should contain those terms that appear most frequently among its members. The mean of each cluster in our system satisfies the above requirement by containing a list of the terms that appear most frequently within the documents in a cluster. For example, say initially that a cluster contains mostly stories about the World Trade Centre bombing. Obviously we want any additional documents that we consider for membership into this cluster to be related to this topic. Intuitively, we can say that there are a set of terms that will be common to all stories which discuss this topic and also that these terms will frequently appear in any such document. The mean of a cluster should contain this set of terms. A statistical analysis of the corpus revealed that the average length of a document, excluding stop words, was approximately eighty-four terms. As we are only concerned with the most frequent terms it not necessary to maintain a mean of this size so a compromised size of fifty of the most frequent terms was reached. This is approximately sixty percent of the average length so encompasses the necessary information level to accurately summarize a document's content. By altering the length of this feature set, we found that clusters of lower quality tended to form, so this length was kept constant. When updating a cluster mean, only terms that have higher frequencies than terms already existing in the mean are included.

3 EXPERIMENTAL RESULTS

In this section the experiments, which have been carried out are presented and conclusions are given.

The performance of data mining systems is generally characterised in terms of two trade-off measures. These are recall and precision. Precision is the percentage of actual relevant documents out of all those that the system decided were relevant. Recall is the percentage of actual relevant documents out of all those that are actually relevant. But how is a relevancy judged? The corpus being used has been fully annotated so that each story within it has been reviewed by human experts and classified as discussing a specific topic. This means that after clustering has been preformed, we can assign a topic to each cluster. This assignment is based on what the majority of documents within it discuss and also on the most frequently occurring terms within the current mean of the cluster. With this information we can then assess the performance of the system. As well as precision and recall, performance for event classification can also be judged by two further measures. These are error measures, misses and false alarms. Misses are referred to as type I errors and are caused when a relevant event is not included in a cluster. False alarms are referred to as type II errors and are caused when a non-relevant event is included in a cluster. Both of these types of error are inversely proportional to the overall accuracy of the system. The methods for calculating them are summarised below, see Table 3.1(a), (b).

	Relevant	Not Relevant
Retrieved	a	b
Not Retrieved	c	d

Table 3.1(a). A General Contingency Table

	Event x is True	Event (S – x) is True
Predicted: discusses x	a	b
Predicted: discussing (S-x)	c	d

Table 3.1(b). An Event Detection Contingency Table, where, x is an event and S is the set of all events.

The retrieved documents are those that are classified by the system as being about an event. The relevant documents are those that were manually judged to be about a specific event.

$$recall = \frac{a}{(a+c)} \qquad precision = \frac{a}{(a+b)}$$

$$misses = \frac{c}{(a+c)} \qquad false\_alarms = \frac{b}{(a+b)}$$

The tests performed to measure the system's effectiveness involved clustering the data set using the k-means algorithm with several different similarity thresholds. The values for precision and recall were extracted from the results. Using

this information, a table was created displaying the information see Figures 3.2(a), (b) below.

Threshold	Precision	Recall
0.05	0.41	0.27
0.1	0.49	0.37
0.15	0.71	0.4
0.2	0.76	0.49
0.25	**0.81**	**0.41**
0.3	0.77	0.38
0.35	0.75	0.22
0.4	0.68	0.24
0.45	0.69	0.24

Table 3.2(a). The average precision and average recall values obtained at various thresholds.

Threshold	Precision	Recall
0.23	0.78	0.44
0.24	0.78	0.42
0.25	**0.81**	**0.41**
0.26	0.82	0.41
0.27	0.79	0.40

Table 3.2(b). The expanded range experiments to determine most accurate threshold.

There is a definite trend in these results. The precision values obtained rise as the threshold increases until it reaches a peak at the threshold level of 0.26. The level of precision obtained here is 0.82. Recall exhibits a similar trend but peaks at the smaller threshold value of 0.20. The recall value obtained at the threshold level of 0.26 is still high, however, leading to the conclusion that it is at this threshold that the most accurate results are achieved. A similar conclusion can be reached when the error measures are examined, see Table 3.3(a), (b). In this Table, the values for false alarms and misses were extracted from the results. Here it can be seen that the false alarm value obtained at the threshold of 0.25 is the lowest and hence the best. Similarly, the miss rate achieved at this threshold remains quite low leading to the same conclusion as above, namely the results obtained at the similarity threshold of ~0.25 are the most accurate.

Threshold	False Alarms	Misses
0.05	0.59	0.74
0.1	0.51	0.63
0.15	0.29	0.6
0.2	0.24	0.51
0.25	0.17	0.59
0.3	0.23	0.62
0.35	0.24	0.78
0.4	0.32	0.76
0.45	0.31	0.76

Table 3.3. A table of the average false alarm and average miss values obtained at various similarity thresholds.

In order to fully assess the performance of our system, we must compare the results obtained above with results obtained using different algorithms. Plotted below are the results of our system, (K)luster compared with clustering results obtained by using the other algorithms outlined in section two, Figure 3.4.

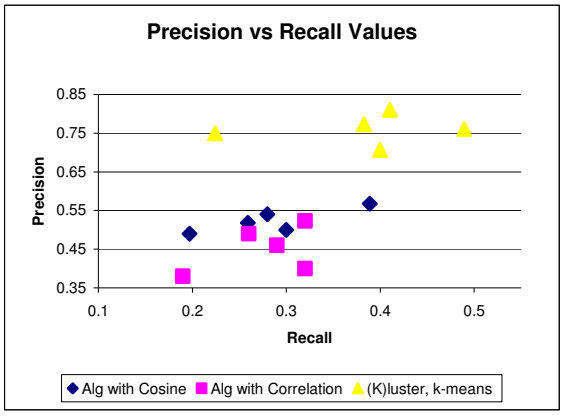


Figure 3.4. A graph of precision versus recall for this system using the three clustering methods.

As can be clearly seen from the above graph the precision and recall values for k-means are significantly higher than those for the cosine and correlation similarities.
False alarm and miss rates for the three clustering methods were also compared, see figure 3.5 below. Here the false alarm versus miss rate for k-means is well below the cosine and correlation similarity rates. This indicates the performance of k-means is a lot better than the other two methods and also that the system is very accurate displaying both high precision and recall values and low false alarm and miss rates.

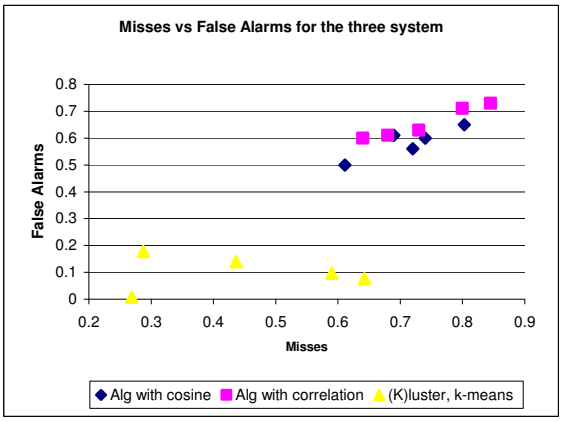


Figure 3.5. A graph of false alarms versus misses for each of the clustering experiments, the cosine similarity, the correlation co-efficient and k-means.

Scalability

Data Mining is traditionally a very computationally intensive task so the goal of looking for very efficient data mining algorithms is vital. In order to completely evaluate our system, we must look at the scalability of our algorithms as the dataset grows. Observing the performance with different data sets tested the scalability of our system. These tests for system efficiency involved clustering different sized subsets of the original data set and comparing system effectiveness. In order to do this we will use the F-measure (Van Rijsbergen, 1979) that combines recall and precision into a single efficiency measure. This is the harmonic mean of the two measures, precision and recall.

$$HarMean = \frac{2 * (recall * precision)}{recall + precision}$$

The magnitude of F varies from 0, when almost no relevant documents are retrieved, to 1, when all and only the relevant documents are retrieved. Moreover F is strongly weighted towards the lower of the two values P and R; therefore this measure can only be high when both P and R are high. This is why the harmonic mean is a useful performance measure.

In the set experiments to examine scalability, four different sized subsets of the dataset where extracted and clustered using the algorithm that obtained the best results from the earlier experiments, k-means. In order to ensure that none of the randomly generated subsets contained bias towards better clusters, several samplings were extracted and clustered. An average of the obtained results for each subset was then used as the final result. The similarity threshold was also varied for each subset. Precision and recall values where calculated and the harmonic mean was obtained. These results were then graphed along with results from the cosine and correlation similarity measures. Again, it can be seen k-means performs very well. The highest value for the harmonic mean was obtained from clustering with the smallest of the four subsets. This is what we would expect. The interesting question is weather the accuracy of the system decreases significantly as the complexity of the dataset increases. An increase in complexity occurs when the feature set being used increases. This is the case when there is an increase in the number of documents being clustered. We can see from Figure 3.6 below, that the results are less accurate for the second subset. Again this is what we expect as this second dataset contains twice as many documents as the first so the complexity of the system has increased. We see for the third dataset, which has over three times the number of documents than the second subset, that the overall accuracy is actually higher. This may be attributed to the formation of the mean within the clusters. With few documents, as in dataset 1, the complexity of the dataset is smaller so the task of clustering is easier. With the larger subset, dataset 3, the mean of the cluster becomes well formed during several iterations and so can form a better basis from which to build the clusters. This leads to better cluster formation and explains the higher values obtained for harmonic mean. The forth subset is four times larger than the third subset and so the complexity of this dataset has greatly increased. As we would expect, the harmonic mean values obtained here are lower. When we compare these results with those of the other two methods, however, we see that the results are better at some thresholds even though this dataset is significantly more complex than that used during the cosine and correlation similarity measure experiments. This is again good news for our system as although the results indicate a drop in accuracy as the data complexity increases, the overall accuracy is still higher than the cosine and correlation similarity measures, which uses a much smaller dataset.

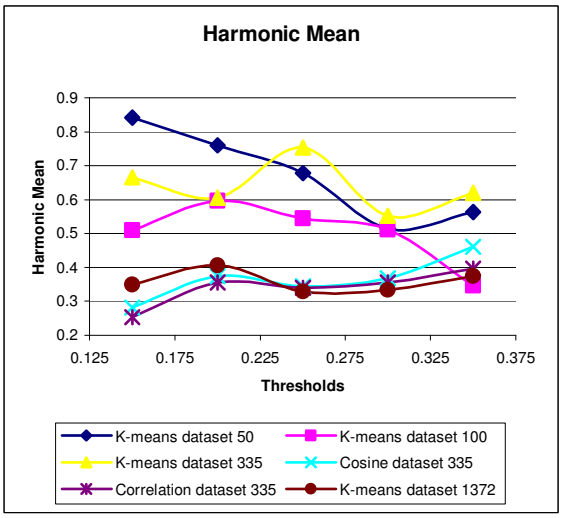


Figure 3.6. A graph of the harmonic mean values obtained for the four different data subsets compared with the cosine and correlation similarities.

Values for false alarm and miss rates were also extracted from the results for the four different datasets, see Figure 3.7. The results here showed that the error measures remained low for all the first three datasets compared with the cosine and correlation similarities but also that the best results were for the dataset 3. Of the three datasets, we see again that dataset 2 performed slightly worse than the other two. Again, as anticipated, the false alarm and miss rates for the forth and largest dataset are higher. This is expected and not unacceptable for very highly multi-dimensional data.

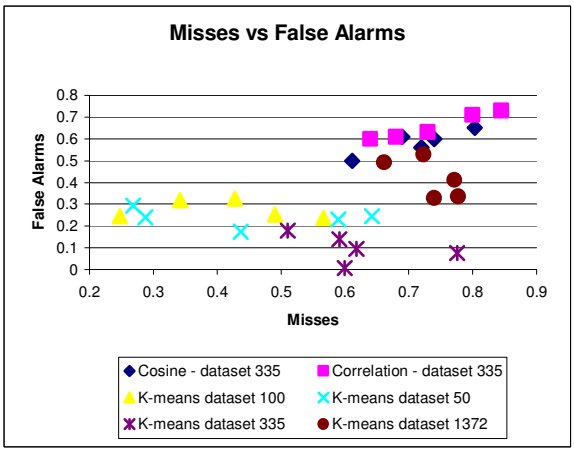


Figure 3.7. A graph of the false alarm and miss rates obtained for the four different data subsets and Clustifier.

In order to further assess the scalability of our system, execution times were recorded during each of the above experiments. The times recorded were graphed in Figure 3.8. The results show that for smaller datasets, our system remains efficient, with only slight execution time increases as the dataset rises from fifty documents through to approximately three hundred and forty documents. For the next dataset we see a clear jump in execution time. We observe that in this dataset at a threshold value of 0.2, which obtained the highest accuracy, the execution time is just over one minuet. Again we consider this an acceptable execution time for this size dataset.

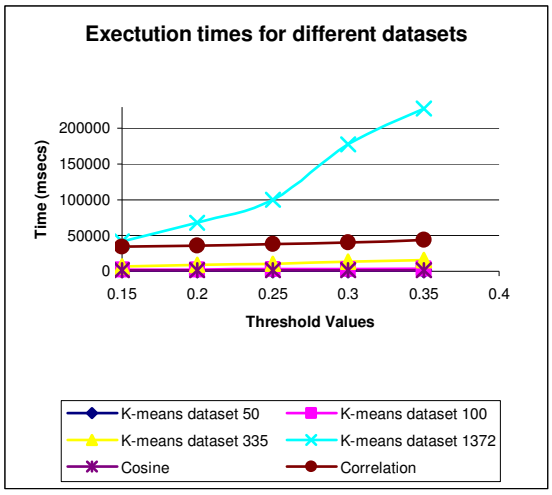


Figure 3.8. The execution times obtained during the clustering experiments.

4 CONCLUSIONS & FUTURE WORK

The above analysis reveals a very identifiable pattern. In terms of system effectiveness, it can be clearly seen that k-means performs very well, giving high precision versus recall values against both the cosine similarity and the correlation coefficient similarity functions.

In terms of false alarms versus misses, we see our system using k-means demonstrates excellent results when compared with the other algorithms we experimented with and also that the rates themselves are very low.

We also found that the k-means algorithm used in (K)luster, is scalable. For small, highly multi-dimensional datasets, the execution times remained low. As the complexity of the dataset greatly increased, however, so too did the execution time.

The dataset used with our system was approximately four hundred megabytes in size. Consider the example given previously of NASA's Earth Observing System dataset, which gathers one terabyte of data every day. It is only when considering datasets of this size that scalability truly becomes an issue for concern. So although (K)luster using k-means proved to be scalable in our experiments, it is clear that future work should take the direction of investigation into techniques that prove to be highly scalable for all size datasets.

Future Work

In response to this need, we intend to complete our work by adapting our current system so that it can effectively and efficiently scale to very large datasets. We intend to do this by not only adapting current centralised data mining algorithms but also taking into account that, no matter how fast an algorithm can process a large dataset, further problems exist. For instance, very often these datasets due to storage limitations or geographical location cannot reside in one centralised system. We propose that by truly pooling the resources of disparate systems an individual can overcome storage and processor concerns and can accomplish computationally very intensive tasks in a reasonable time. Phase two of this project will involve the further investigation of scalable data mining algorithms. In this next phase of research we will also migrate our existing system to a distributed environment in an attempt to contribute to solving the issues of distributivity, heterogeneity and data mining time complexity on large data sets. Through this work we will also be addressing the ever-increasing need for distributed solutions to mining naturally distributed data.

ACKNOWLEDGEMENTS

We would like to thank the referees for their comments, which helped improve this paper.

5 REFERENCES

1. Haitian, J. *Clustering Algorithms*, John Wiley & Sons, New York, NY, 1975.
2. http://www.nist.gov/speech/tests/tdt/
3. Hartigan, J. and Wong, M. *Algorithm AS136: A k-means clustering algorithm. Applied Statistics, 28, 100-108*, 1979.
4. Porter, J. *Disk Trend Report*, 1998.
5. Porter, M.F. *An algorithm for suffix stripping*, 14, 130-137, 1980.
6. Fayyad, U. and Uthurusamy, R. *Evolving Data Mining into Solutions for Insights*, Communications of the ACM, 2002.
7. Retrospective Event Detection using Clustering Algorithms, Report 2002.
8. R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, New York, John Wiley and Sons, 1973.
9. B. Thiesson, C. Meek, D. Chickering, and D. Heckerman. *Learning Mixtures of Bayesian Networks*, Microsoft Research Technical Report TR-97-*30*, Redmond, WA, 1997.
10. R. Ng and J. Han, *Efficient and effective clustering methods for spatial data mining*, 20th International Conference on Very Large Data Bases, Santiago, Chile, 1994.
11. J.F. Roddick and M. Spiliopoulou, *A Survey of Temporal knowledge discovery paradigms and methods*, IEEE Transactions on Knowledge and Data Engineering, 2001.
12. http:// eospso.gsfc.nasa.gov/
13. P. Adriaans and D. Zantinge, Data Mining, Addison-Wesley, 1996.
14. Raymond T. Ng and Jiawei Han, *CLARANS: A Method for Clustering Objects for Spatial Data Mining*, IEEE Transactions on Knowledge and Data Engineering, Vol. 14, 2002
15. Huang, Zhexue. *A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining*, Cooperative Research Centre for Advanced Computational Systems, 2002.
16. Peck, D. *Statistics, The Exploration and Analysis of Data*, 100 – 156, Duxbury, 1997.

Knowledge-based intelligent error feedback in a Spanish ICALL system

Thomas Koller[1]

Abstract. This paper describes the Spanish ICALL system ESPADA which helps language learners to improve their syntactical knowledge. The most important parts of ESPADA for the learner are a Demonstration Module and an Analysis Module. The Demonstration Module provides animated presentation of selected grammatical information. The Analysis Module is able to parse ill-formed sentences and to give adequate feedback on 28 different error types from different levels of language use (syntax, semantics, agreement). It contains a robust chart-based island parser which uses a combination of mal-rules and constraint relaxation to ensure that learner input can be analysed and appropriate error feedback can be generated.

1 INTRODUCTION

Many CALL packages fall short when it comes to providing the learner with individualised teaching and flexible feedback [4]. ICALL is very useful in this context as it can automatically generate detailed feedback and provide an individualised environment for each learner. Through this personalisation the learner is more involved in the own language learning process.

The aim of ESPADA is the development and implementation of a fully functional syntax teaching system for German learners of Spanish.[2] In Germany, the teaching of grammar plays an important role in language teaching, but the teaching of syntax in particular is often neglected. Literature pertaining to teaching Spanish syntax in Germany is very sparse. ESPADA aims to provide the learner with a curriculum-independent resource for learning and practising syntactical structures. The project was pedagogically driven and incorporated recent findings on the *real* use of feedback by learners [1]. Currently, ESPADA is directed towards adult learners with basic linguistic knowledge about grammar and syntactic patterns.

The RECALL system (see [4]) is a system that teaches syntax using ICALL resources. It has a Learner Module that contains a model of the individual learner so that the exercises can be tailored specifically to each learner. The Tutoring Module contains an Exercise and a Test Library in order to provide the learner with varied training resources.

ESPADA shares some of the ideas of the RECALL system. However, it is much smaller in scale. For example, it does not have a Learner Module. The Demonstration Module (which is similar to the Tutoring Module of RECALL) of ESPADA provides animated grammars which differ from traditional methods to illustrate grammatical properties. The Analysis Module is able to properly recognise sentences with the wrong type and/or number of sentence components.

The main feature of ESPADA is the multi-faceted preparation and feedback offered to the learner. It consists of three components and their interaction: a Demonstration Module (DM), a Lexicon Module (LM) and an Analysis Module (AM). The Demonstration Module provides animated presentation of selected grammatical information; the Lexicon Module stores and selectively displays lexical information and the Analysis Module dynamically analyses learner input and generates appropriate feedback.

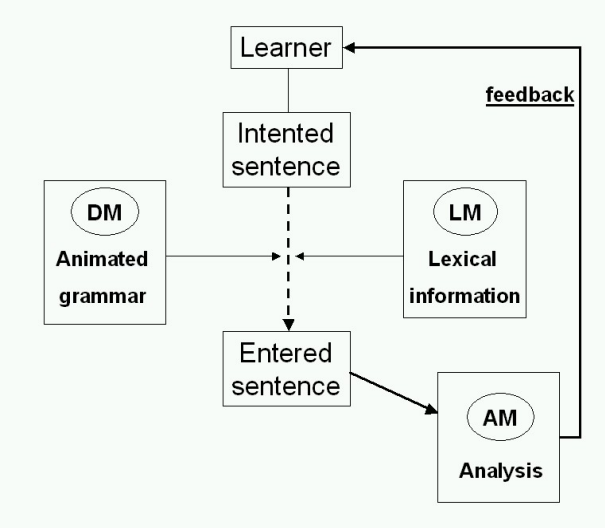


Figure 1. Interaction of Modules

ESPADA allows the learner to enter more than a word or phrase. The learner can enter a simple Spanish sentence, which is then analysed and flexible feedback is generated. This feedback can be useful and is usually lacking in commercial language learning systems. The learner can gather further information about syntactic and semantic properties with the help of the Lexicon Module and the Demonstration Module. All the modules can be accessed at the same time.

In Figure 2, the Lexicon Module displays the sentence patterns and verbal forms of *beber* (*to drink*), the Demonstration Module explains the characteristics of nominal groups and the Analysis Module displays the result of analysing the (correct) sentence *El padre bebió un café.* (*The father drank coffee.*)

[1] School of Computing, Dublin City University, Dublin, Ireland, email: tkoller@computing.dcu.ie

[2] Therefore the basic interface language is German. The system has now been enhanced to also have English as the medium of instruction.

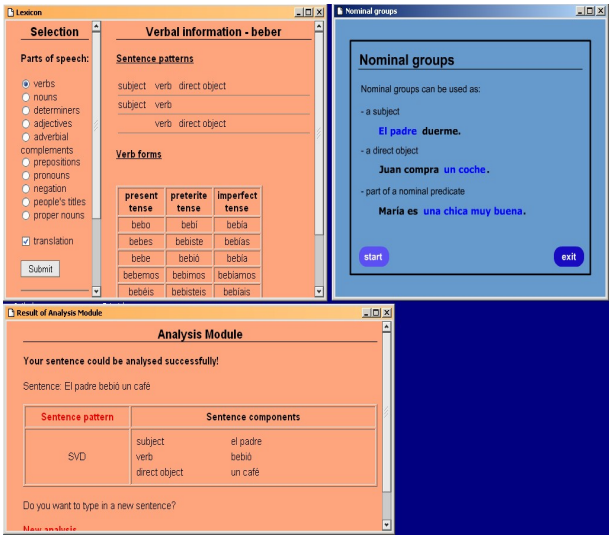


Figure 2. Simultaneous use of modules

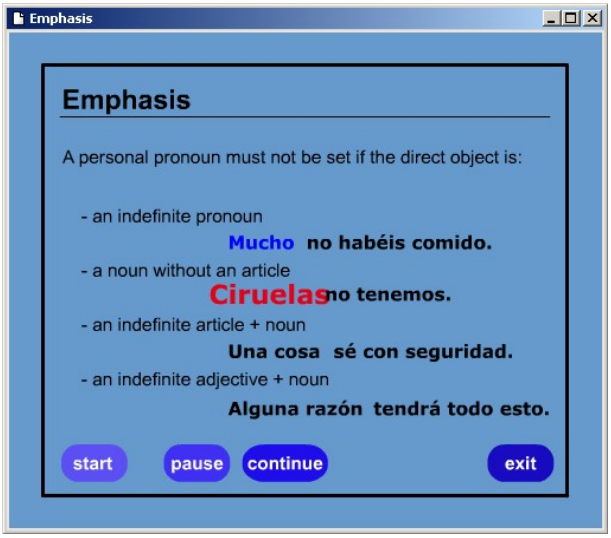


Figure 3. Demonstration Module - Explaining topicalisation

ESPADA is based on a detailed knowledge base of linguistic description of syntactic features. It has a modular design with a web-based implementation using HTML and JavaScript. The full form lexicon is coded in XML (based on a XML schema file) and contains 300 lemmas and 1900 full forms. We use a chart-based island parser with mal-rules (see [6], [4]) and constraint relaxation (see [2]).

2 DEMONSTRATION MODULE

The Demonstration Module - together with the Analysis Module - represents an animated grammar. The Demonstration Module offers the learner a dynamic representation of grammatical information, which is in contrast with traditional ways of displaying such information. It explains syntactic structures with the help of short Flash animations. These animations can be viewed repeatedly and stopped during rendering, so that the learner has full control of the topics presented (see Figure 3). The Demonstration Module contains 14 different animations, which are classified into 8 main topics (placement of the subject, agreement, types of complements, properties of complements, negations, personal pronouns, *ser* vs. *estar*, topicalisations).

Without the Demonstration Module, the feedback of the Analysis Module would have to be much more extensive. Heift [1] recommends avoiding extensive feedback stating that feedback exceeding three lines was not read by learners.

3 LEXICON MODULE

The Lexicon Module (see Figure 4) is interactive and selectively displays the following data in the lexicon:

- all available lemmas of a selected POS
- morphological, syntactic and semantic properties of a chosen verb (allowed sentence patterns, semantic features of required complements and conjugated verb forms)
- different sets of semantic features
- all words of non-verbal POS having a particular set of semantic features

With the information available in the Lexicon Module, the learner can build up a sentence before using the Analysis Module: after choosing an appropriate verb according to the communicative intentions, the learner finds out the required sentence patterns, the semantic type of the complements and the conjugated forms of this verb. Then the learner can see a list of all non-verbal words that correspond to the semantic features required by the verb.

The verb *beber* (*to drink*), for instance, has a possible sentence pattern SVD (subject - verb - direct object) with the required semantic features *+humano* (human) for the subject and *+bebida* (drinkable) for the direct object. Subsequently, the learner can find in the lexicon several personal pronouns and nouns bearing the feature *+humano* and the nouns *café* (*coffee*), *leche* (*milk*) and *vino* (*wine*) for the direct object.

4 ANALYSIS MODULE

The Analysis Module is able to perform a detailed analysis of sentences submitted by the learner and to return selective and appropriate feedback to the learner. It can recognize and generate feedback on 28 different error types from different levels of language use (syntax, semantics, agreement). Mistyped words and wrong morphological forms are just handled as errors so that a high degree of robustness of analysis can be ensured. According to [3], very few programs have the ability to give adequate feedback to the learner if an ill-formed sentence is encountered.

The Analysis Module uses a combination of constraint relaxation, different types of mal-rules and a number of pre- and post-parsing tests to ensure that ill-formed input can be analysed and errors are detected.

4.1 Range of analysis

The Analysis Module is able to analyse simple Spanish sentences. The range of analysis comprises unmarked sentences as well as sentences in which a nominal phrase or a prepositional phrase appears in

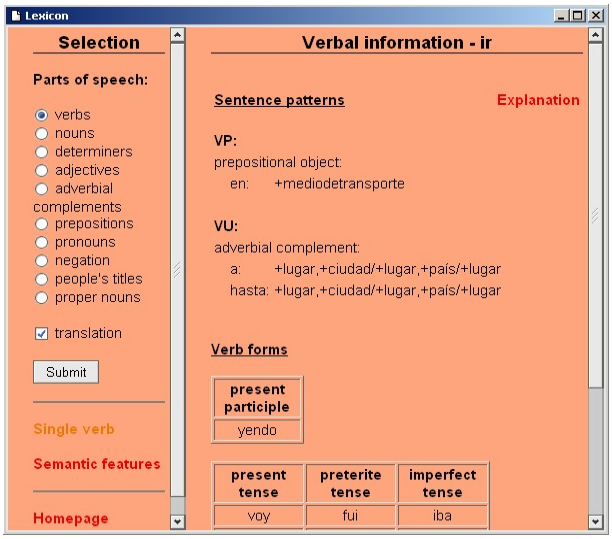


Figure 4. Lexicon Module

first position in order to put special emphasis on that part of the statement[3]. The Analysis Module is also able to control the correctness of object pronouns in connection with a topicalized object. It can handle sentences with the simple negation adverb *no*, with complex verbal groups and with up to two adverbial complements.

4.2 Main steps of analysis

- Word forms: Syntactic and semantic information of the word forms in the entered sentence are retrieved from the lexical knowledge base.
- Variants: If any word forms of the sentence can belong to several POS, the learner is asked to choose the intended POS for each ambiguous word form.
- POS filtering: With the help of the previously gained POS information, the position of the word forms representing a negation adverb or an adverbial complement are saved. Then these word forms are temporarily taken out of the sentence. This drastically reduces the complexity of the parsing process because different types of adverbial complements can appear at very different places of the sentence. Without the negation adverb or any adverbial complement, any remaining word form of the sentence has to belong to the subject, the verbal group or an object.
- Verbal group: The verbal group is checked through various if-then-tests. It is tested, for instance, if the verbal group starts with a finite verb, if the verbal form of an infinite verb (i. e. past participle, gerund or infinitive) meets the requirements of the preceding verb (e. g. the verb *querer* requires a following verb in infinitive) and if the last verb in the verbal group is a full verb.
- Negation adverbs and object pronouns: The number and position of these POS are checked simultaneously as their positions depend on each other. If any adverbial complements of time were used, their temporal agreement with each other as well as with the finite verb is controlled.
- Determining sentence patterns: If no errors are encountered up to this stage, the basic sentence patterns (e. g. SVD, SVP) of the main verb are extracted from the lexicon. As topicalised sentence patterns (with an object put in first position) are supported, the basic sentence patterns are permutated (here obtaining the sentence patterns DSV, PSV)). Then all these patterns are translated into phrase patterns (e. g. SVD translates into NP|NP and into NP|PP[4]). The sentence patterns and its phrase patterns represent the basic elements for the parsing process.
  If any errors are found, the analysis process is stopped immediately and adequate feedback is given to the learner. The learner can then modify the input and restart the analysis process.
- Parsing: The parsing process is done twice. First, the part of the sentence which precedes the verbal group is parsed against those parts of all phrase patterns which are on the left side of the separator `|`.
  Second, the part of the sentence which follows the verbal group is parsed against those phrase patterns which were parsed successfully in the first parsing phase. It is matched against the part of every phrase pattern on the right side of the separator `|`.
  For instance, given the sentence *El padre dio un libro a su hijo.* (*The father gave a book to his son*) and the phrase pattern `NP|NP;PP`, the phrase *El padre* will be parsed against `NP`. If this parsing process is completed successfully, *un libro a su hijo* will be parsed against `NP;PP`.
- Tests: If any regular sentence patterns of the main verb have been parsed successfully, a broad range of syntactic and semantic checks as well as agreement tests are executed in order to detect corresponding errors. After all the tests have been done, a ranking of sentence patterns is performed according to the number of found errors.
- Displaying results: The sentence pattern(s) with the lowest number of errors are displayed.

4.3 Grammar rules

The grammar shows two particular features:

- It has no entries on sentence level (like S → NP VP). The parser only has to analyse single parts of the entered sentence (the parts before and after the verbal group).
- It contains mal-rules in order to recognise faulty nominal phrases representing ungrammatical structures of Spanish (for instance, the mal-rule `NP_Y|Titel;PNn` recognises a nominal phrase which lacks a determiner in connection with a person's title (*señor, señora, señorita*)). As a result, the process does not fail if defective structures of a certain type are encountered. The phrase recognised by the mal-rule will be added to the list of completed phrases (regular nominal phrases are labelled NP_1, NP_2, etc., whereas anticipated faulty NPs are named NP_X and NP_Y). After parsing has been completed, these faulty phrases are identified by a subroutine.

4.4 Parser

The parser is a chart-based island parser making use of the Earley algorithm. It only needs to analyse the non-verbal parts of the sentence. The verbal group in simple Spanish sentences has a fixed posi-

[3] Unmarked sentence: *El padre dio un libro a su hijo.* (*The father gave a book to his son.*) Marked sentence: *A su hijo el padre le dio un libro.* (**To his son the father him gave a book.*)

[4] The | sign symbolises the verbal group and is used as a separator by the parser

tion and can only contain certain POS (verbs and prepositions[5]). For that reason it can be examined easily with if-then tests. If an error is encountered in the verbal group, the analysis stops immediately and the user gets an appropriate feedback. This approach helps to avoid as many as error sources as possible before starting the actual parsing process. Using this approach, a broad range of error feedback can be given very fast because the parsing process and the following tests on syntax, semantics and agreement are only due to start if these simple tests have been concluded without any faults.

4.5 Mal-rules

The Analysis Module contains two different kinds of mal-rules. These mal-rules recognise faulty structures both at the level of sentence components and within nominal phrases.

The latter type is able to recognize ill-formed input within NPs either concerning wrong word order or missing words. New mal-rules of this type can be easily integrated at a later stage to adapt the grammar to L1-specific learner mistakes or to differing NP structures in other languages.

The type of mal-rules at sentence level is used to parse sentences where the type and/or number of recognised complements does not correspond to the required type and number of complements in sentence patterns of the main verb.

For simple sentences there is a finite amount of possible sentence patterns. This set of sentence patterns is equivalent to all meaningful combinations of the set of complements comprising subject, direct object, indirect object, prepositional object, necessary adverbial complement, for example SV, SVD, SVI, SVP, SVU, SVDI.

After extracting the regular sentence patterns of a given main verb in learner input, all other possible sentence patterns are added to the chart explicitly labelled as mal-rules. Both the lexically induced and the mal-rule sentence patterns are then parsed in the same way. If any regular sentence patterns have been parsed successfully, only these patterns are then checked thoroughly for syntactic and semantic errors as well as errors of agreement. If only mal-rule based sentence patterns have been parsed successfully, the analysis process stops immediately and the recognised sentence components are displayed.

4.6 Constraint relaxation

Constraint relaxation (see [5]) represents another important means to properly recognize ill-formed input. In this case it is implemented in terms of successive, layered constraint application and constraint violation bookkeeping. No constraints are applied at all during initial parsing. After parsing has been completed, syntactic and semantic adequacy as well as agreement requirements are controlled via multiple tests. After these tests have been done, the constraint violations are counted for each successfully parsed sentence pattern. The sentence pattern(s) with the lowest number[6] of constraint violations are then displayed.

5 GENERATING FEEDBACK

Heift [1] cited the results of a survey about acceptance of feedback as follows:

- Students did not try to correct themselves if no feedback about the type of error was provided.

[5] The only exception to the rule is the conjunction *que* in the expression *tener que*.

[6] This is always a definite number, not a range of numbers.

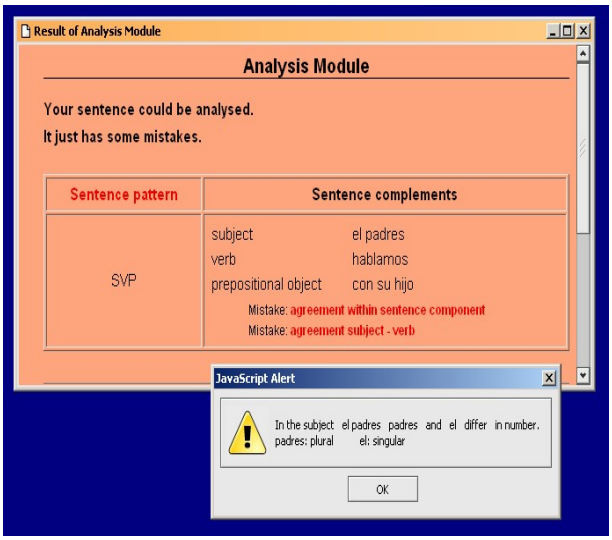


Figure 5. Analysis Module

- Feedback exceeding three lines was not read.
- If more than one error was displayed at a time the correction was felt to be too complex.

These results had significant influence of how the feedback in ESPADA was designed. A cascaded feedback system provides two levels of information. If any errors are encountered after parsing has been done, they are all displayed, but only in a shortened form merely giving a hint about the type of each error. Dynamically generated links provide more detailed feedback (which nevertheless does not exceed two lines). This feedback informs the learner of the precise type of error and the values which do not match. Besides that, further general information can be obtained via additional links. (In Figure 5, the browser window displays the result of analysing a sentence with minor errors. The Analysis Module gives information about the recognised parts of the sentence and provides for every error a short explanation combined with a dynamic link. After clicking on the link a JavaScript alert-window opens and gives further information about the error in question.)

There are four different types of feedback after parsing and all constraint checks have been done:

- The sentence could be parsed successfully with at least one regular sentence pattern and no constraint violations were detected: The feedback consists of the successfully parsed sentence pattern(s) and the corresponding sentence components.
- The sentence could be parsed successfully with at least one regular sentence pattern, but constraint violations were detected: The sentence pattern(s) with the lowest number of constraint violations are displayed. The errors are displayed with the help of the cascaded feedback system previously described.
  The errors are not weighted due to the following reason: With the available information the errors could only be organized according to the linguistic type of error (syntactic, semantic, agreement). We doubt that such categories would be of much use for non-linguists.
- The sentence could only be parsed successfully with a mal-rule on sentence level. It contains correctly build phrases, but does not coincide with any regular sentence pattern of the main verb. The

feedback shows the recognised phrases and offers further general information.

- It was not possible to parse the sentence with any sentence pattern (either regular or mal-rule based). The learner gets feedback about this failure of analysis as well as general information about possible reasons.

6 CONCLUSIONS & FURTHER WORK

ESPADA offers the learner a comprehensive curriculum-independent means to learn about syntactic structures of simple Spanish sentences. It provides intelligent knowledge-based error analysis and feedback.

A combination of constraint relaxation and mal-rules at different levels has been implemented to ensure that ill-formed sentences can be parsed successfully. Unlike previous approaches using constraint relaxation, this system is implemented in terms of successive, layered constraint application and constraint violation bookkeeping.

The special structural properties of simple Spanish sentences (i. e. easy separable verbal group, fixed position of negation adverb and object pronouns) are exploited in an optimal manner to reduce the input to the parser. The extracted sentence parts are checked on correctness with simple if-then-tests before parsing starts. As a result, the analysis process is speeded up considerably.

ESPADA represents a valuable starting-point for a multilingual ICALL system, which will be developed for several Romance languages (French, Spanish and Italian). Extended versions of the parser and the grammar (containing language-specific rules for several languages) will be very useful to teach contrastive features of Romance languages on the sentence level. Weighted constraints will be deployed to tailor the feedback of the ICALL system more tightly to each individual learner. Priorities can be set either through a database of learner errors or a learner module (see [4]) (saving the previous sessions of a learner).

The feedback of the Analysis Module will be linked directly to corresponding animated grammars in order to increase the learning effect. Animated grammars seem to be rarely used (at least in a systematic manner) and scientific documentation about whether and how to use this kind of grammars is very sparse (see [7]). Therefore the animations developed here may be an interesting starting point for systematic basic research into the design and use of animated grammars.

Currently an online evaluation platform is being created for empirical evaluation of this system. The questions of this platform are tightly linked to the expected features of the PhD work so that the results of this evaluation can be exploited directly for future work.

ACKNOWLEDGEMENTS

We would like to thank the referees for their comments which helped improve this paper.

REFERENCES

[1] Trude Heift, 'Error-specific and individualized feedback in a web-based language tutoring system: Do they read it?', *ReCALL*, **13(2)**, 129–142, (2001).

[2] Wolfgang Menzel, *Modellbasierte Fehlerdiagnose in Sprachlehrsystemen*, Niemeyer, Tübingen, 1992.

[3] Wolfgang Menzel and Ingo Schröder, *Constraint-based diagnosis for intelligent language tutoring systems*, Fachbereich Informatik, Universität Hamburg, 1998. Report Nr. FBI-HH-B-208-98.

[4] Maureen Murphy, Anja Krüger, and Andrea Grieszl, *RECALL - Providing an Individualized CALL Environment*, 62–73, Language teaching and language technology, Swets and Zeitlinger, Lisse, 1998.

[5] Veit Reuer, 'Error recognition and parsing of syntactically mildly ill-formed natural language', in *Proceedings of the LFG00 Conference*, pp. 215–225, (2000).

[6] David Schneider and Kathleen F. McCoy, 'Recognizing syntactic errors in the writing of second language learners', in *Proceedings of the 17th International Conference on Computational Linguistics (COLING)*, pp. 1–2, (1998).

[7] Paul A. Sundberg. Animation in CALL: Learning to think in the fourth dimension. Paper Presentation at the CALICO '98 Symposium. online: http://lrs.ed.uiuc.edu/students/p-sundb/homepage/animCALL.html.

PUNC: A Model of Conceptual Combination

Dermot Lynott and Mark T. Keane [1]

Abstract. We present a model of conceptual combination, PUNC, based on the Constraint Theory of conceptual combination ([1], [2]). This model incorporates the primary constraints of the Constraint Theory; Informativeness, Diagnosticity and Plausibility in an integrated fashion, creating a cognitively plausible mechanism of interpreting novel noun-noun phrases. We detail the model, including knowledge representation, interpretation production mechanisms and the ranking of interpretations in terms of their overall goodness. We also discuss how the PUNC model improves on existing models of conceptual combination.

1 INTRODUCTION

Nominal compounds like *holiday drug mule*, *soccer mom*, *laptop computer* and *trash cookies* are pervasive and illustrate the creativity of everyday language use. Conceptual combination has long been viewed as a microcosm of the creative and generative nature of language, with new meanings being created constantly by the re-combination of words in syntactically well-formed phrases. It is the need to understand this generativity of language that has motivated several decades of cognitive science research into nominal compounds (e.g., [3], [2], [4], [5], [6], [7], [8]).

This research has thrown up a number of competing models of conceptual combination, each of which tries to capture the main empirical phenomena in the field. We briefly outline these models before concentrating on the Constraint Theory and a new algorithmic instantiation of that theory: PUNC (Producing and Understanding Novel Compounds). PUNC uses the primary constraints of the constraint theory (diagnosticity, informativeness and plausibility) to generate a set of interpretations for any noun-noun compound, ranked by their overall goodness.

2 Other Theories and Models

There are three main theoretical models of conceptual combination in the cognitive Science literature: Dual Process Theory ([8], [9]), CARIN (Competition among Relations in Nominals, [10], [11]) and the Constraint Theory ([1], [2], [12]). Each of these theories proposes different mechanisms and different interactions between the modifier and head concepts of a novel compound (i.e., the first and second words in a compound). We give an overview of these models, before giving more detail on the Constraint Theory in the next section.

The *Dual Process Theory* [9] proposes two mechanisms for understanding novel, noun-noun compounds. The first, scenario creation, gives rise to interpretations that are linked by a relation (e.g., a *robin snake* is a snake that *eats* robins). The second, comparison and alignment, gives rise to interpretations that involve transferring a property from the modifier to the head concept (e.g., a *robin snake* is a snake with a red breast). Using these two mechanisms, the dual-process view accounts for two broad classes of interpretations; relation-based and property-based. Relational interpretations involve the use of a thematic relation connecting the two words in the compound (e.g., *dawn flight* is a "flight that *takes off at* dawn"). Property interpretations assert a property of one concept of the other concept (e.g., *bullet train* is "a very fast train", asserting the property of bullet of the train).

The *CARIN* model posits that property-based interpretations are rare, and that people are far more likely to produce interpretations that use relations. As such, CARIN predicts what interpretations people will produce based on what it terms "Modifier Relation Frequency"; in other words, how a modifier has been combined in other compounds in the past. For example, most compounds of the form a "chocolate X" mean, "an X made from chocolate" (e.g., *chocolate egg*, *chocolate coin*, *chocolate bar*). Therefore, future compounds of this form will be more likely to give rise to this kind of interpretation. In this way CARIN accounts for relational interpretations, while remaining silent on the origins of property interpretations.

However, both Dual-Process and CARIN accounts have drawbacks. Dual-Process requires multiple mechanisms to account for various phenomena, while the CARIN theory largely ignores an entire class of interpretation. The Constraint Theory of conceptual combination overcomes these obstacles by proposing a unitary unification mechanism to account for a wide range of data.

3 The Constraint Theory

The Constraint Theory ([1], [2]) says people understand novel, noun-noun combinations by using three primary constraints – diagnosticity, informativeness and plausibility – whose interactions dictate what

[1] Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland. Email dermot.lynott@ucd.ie

interpretations are produced. The theory has been realised in a computational model, called C3.

Diagnosticity relates to the salience of the features of a concept. A concept's diagnostic features best distinguish that concept from other concepts. For example, "has wings" is a more diagnostic feature of birds than "has legs", since many creatures have legs, but far fewer have wings. Diagnostic features facilitate the activation of the concepts involving these features. For example, using the feature "has a trunk" would call to mind "elephants" more easily than using the feature "is grey" ([13]). For this reason diagnostic features are often employed in people's interpretations for novel compounds. For example, "a beetle that is spiky" is often produced as an interpretation for the compound *cactus beetle* since it uses a highly diagnostic feature of the concept CACTUS. Constraint theory proposes that interpretations of novel, noun-noun compounds will utilise diagnostic features of both the head and modifier concepts.

Plausibility requires that interpretations utilise features that have already co-occurred in past-experiences. For example, of two interpretations for the compound *angel pig*, "a pig with wings on its torso" is more plausible than "a pig with wings on it tail" for the simple reason that "wings on torsos" has been encountered in the past, whereas "wings on tails" has not. In this way, plausibility overlaps produced interpretations with prior experience (i.e., specific instances stored in memory) to judge its overall goodness.

Informativeness specifies that an interpretation must contain a certain amount of new information. Constraint theory predicts that uninformative interpretations will not be produced. For example, a possible interpretation for the compound *pencil bed* might be "a bed that is made of wood". However, since pencils are traditionally made of wood such an interpretation would be rejected as uninformative.

3.1 The C3 Model

The C3 model ([1]) implements the three constraints of diagnosticity, plausibility and informativeness. C3 calculates the diagnosticity of a feature by comparing it to all other concepts in memory (see [1] for details). For instance, if the feature *prickly* occurred only once, in the concept CACTUS, then it would have a high diagnosticity score for that concept. C3 uses the resulting diagnosticity scores to construct partial interpretations for a compound using the diagnostic features of both concepts. To compute plausibility, these partial interpretations are then compared to existing concepts, using their level of overlapping semantic features as a measure of plausibility. An interpretation is scored as being completely plausible if it overlaps entirely with some stored instance in the knowledge base. During this stage these partial interpretations are also elaborated with additional properties from relevant concepts in the knowledge base, but only if these additions increase the plausibility of the interpretation.

Finally, *informativeness* is implemented as a post-hoc filtering process in C3. Once interpretations have been produced they are examined to ascertain whether they are informative or not. Interpretations that do not contain a requisite amount of new information are rejected. Informativeness is therefore a binary affair with new information either being present or absent.

For a given compound C3 generates approximately 4,000 unique interpretations ([1]) with the entire process taking from several hours to days. Obviously it would not be feasible to examine every single interpretation generated, so for simulation purposes a threshold on the overall goodness is usually set so that only the top 10 interpretations are output once processing is complete.

4 The PUNC Model

The PUNC model is an implementation of the Constraint Theory that seeks to improve upon C3 by reducing the amount of processing required, while still producing interpretations that parallel those produced by people. PUNC retains the central constraints of diagnosticity, informativeness and plausibility, but implements them differently to the C3 model. We detail the knowledge represented in PUNC and how this knowledge can be meshed to generate interpretations. We also discuss how these changes improve PUNC's performance compared to existing models.

4.1 Knowledge Representations in PUNC

PUNC uses a simply hierarchy to represent the most diagnostic knowledge associated with each concept. This knowledge encoded includes diverse information such as diagnostic features, functions, roles, and relations to other concepts. For example, the concept CACTUS is represented by diagnostic features such as "has spikes", "grows in the desert", "can conserve water". Since CACTUS also inherits features from PLANT, the feature "can grow", or "can be eaten" is also represented. Similarly, any concept that inherits from CREATURE will have the feature "can eat things", so specific instances of creatures e.g., BEETLE, DOG, SNAKE, will also be able to eat things. Each feature is weighted by its importance relative to the concept, so "has spikes" is weighted as being the most important feature of CACTUS (e.g., a weighting of 1), while "grows in the desert" is weighted as being of slightly less importance (e.g., a weighting of 3). The PUNC knowledge base was coded blind (i.e., without reference to specific compounds or interpretations) so current weightings currently represent intuitive values for each feature of a concept.

Features that are inherited from a parent concept are not as diagnostic to the child concept (e.g., "can photosynthesise" is less diagnostic of CACTUS than it is of PLANT). Therefore, inherited features are always

weighted relative to the diagnostic features already present in the child concept. For example, if the feature "can conserve water" had a diagnosticity score of 4, then any feature inherited from PLANT would have a weighting of 5 or more.

Since PUNC merely represents features that are important to each concept, the knowledge represented is quite diverse, but it is all treated in the same fashion. For example, the feature "has spikes" (a property) is important for CACTUS, whereas the concept VET might have the feature "treats animals" (a role), or the concept CHESS might have "strategic" or "tactical" (a more subjective property). Of course the importance of a feature for a particular concept can change from one situation to another or from one person's perspective to another's, but this knowledge base gives a snapshot of this knowledge and how these concepts relate to one another.

4.2 How PUNC Produces Interpretations

The input to PUNC is a noun-noun combination and the output is a list of possible interpretations for that combination, ranked in terms of their overall goodness. The model produces multiple interpretations for each combination; interpretations that parallel those produced by people. PUNC provides a diverse selection of interpretations, using a parsimonious mechanism that reflects both the speed and efficiency with which people understand these compounds, and the diversity of the interpretations they produce.

The two most common types of interpretation produced by people are described as property-based and relation-based. Property-based interpretations occur when a property of the modifier is transferred to the head (e.g., *bee beetle* as "a yellow and black striped beetle"). Relation-based interpretations occur when some relation links the head and modifier concepts (e.g., *cookery magazine* as "a magazine that is about cookery"). Property-based interpretations account for 30-50% of interpretations produced, with relation-based interpretations accounting for the same amount again ([1]). Other types of interpretations that occur with less frequency include conjunctions (e.g., *pet fish* as "something which is both a pet and a fish") and known-concept interpretations, where the interpretation refers to an object that already exists (e.g., *pencil bed* as "a pencil case"). Through the mechanism described below, PUNC is able to generate various types of interpretation, in a manner that reflects how people perform the process of conceptual combination.

4.2.1 Meshing Knowledge to Produce Interpretations

PUNC generates different types of interpretations through meshing the available knowledge from modifier into the knowledge of the head. When knowledge is meshed it can give rise to possible interactions between the head and modifier concepts, or to the transfer of some aspects of the modifier concept to the head. Each concept consists of a collection of features (e.g., properties, roles, etc.) that describes it and how it relates to other concepts. PUNC compares the features of the modifier concept to those of the head to establish whether elements of both can be meshed.

A simplified description of the knowledge represented for the concepts CACTUS and BEETLE is given in (i) and (ii). We will use these descriptions to illustrate how knowledge is meshed to create interpretations.

i CACTUS: is spiky, is found in deserts, can conserve water, is green, can be eaten, can photosynthesise

ii BEETLE: is black, has 6 legs, has antennae, can eat things, can be eaten.

Since each feature encoded has an associated diagnosticity weighting, PUNC processes features in descending order of diagnosticity. Firstly, PUNC examines each feature in the modifier to see if it can be meshed with the head's features to create interpretations. Secondly, it examines if the modifier itself can be used to fulfil some role in the knowledge represented in the head concept. As a starting point, PUNC uses the knowledge of the head concept as the core for each interpretation that is produced. This core is then altered or augmented by whatever new features are introduced from the modifier.

For the compound *cactus beetle*, PUNC firstly looks at the modifier's representation (see i) and attempts to create a unique interpretation using each piece of knowledge. For example, the feature "is spiky" from CACTUS is taken and compared to the features of the head concept. If this feature is considered informative (i.e., if it does not already exist in the head concept), then it can be used to create an interpretation. The "is spiky" knowledge from CACTUS is then meshed with the existing representation of BEETLE forming a representation for the interpretation, which contains all of the diagnostic features of the head concept. The newly incorporated information ("is spiky") is elevated to being the most diagnostic feature of this new representation, and as such can be used to create a gloss of the new representation of "a beetle that is spiky". This interpretation distinguishes this beetle from other types of beetle. In this way PUNC outputs both a representation of the interpretation and a text description of the interpretation. In the same way that PUNC produces "a beetle that is spiky", it will also generate the interpretations "a beetle that is found in deserts" and "a beetle that can conserve water".

In some cases the knowledge being meshed from the modifier can conflict with knowledge that already exists in the head's representation. For example, when PUNC compares the knowledge "is green" from CACTUS to the knowledge of BEETLE, there is a clash, because "is black" is represented as being the colour of beetles. In incorporating this knowledge from the modifier PUNC overrides the information that was in the head to create a representation for "a beetle that is green".

When the feature "can be eaten" from the modifier is compared to the head, PUNC meshes this with the knowledge from BEETLE that it "can eat things", as there is a reciprocal relationship between these two pieces of knowledge. This produces a representation for the interpretation "a beetle that eats cactus". On the other hand, when PUNC encounters the knowledge "can be eaten" in the both concepts, it does not create the interpretation "a beetle that can be eaten"; as that feature is already present in BEETLE and so the resulting interpretation would be uninformative.

At this point, PUNC examines whether there are features in the head concept that can incorporate the modifier itself to form an interpretation (as opposed to specific features of the modifier concept, as above). For example, there may be actions that the head concept can perform on other concepts (e.g., eats, plays, hunts etc.). For *cactus beetle* there are no features that allow this (see Table 1 for a list of interpretations produced by PUNC for the compound *cactus beetle*). However, if the compound were *cactus magazine*, MAGAZINE contains the feature "can be about something". PUNC meshes this feature with the modifier itself to produce a representation for "a magazine that is about cacti". Similarly, cactus *magazine* could also be "a magazine that is made from cactus" as magazines can be made from different things. Incorporating the modifier itself in this way gives rise to a variety of interpretations depending on the features present in the head.

Each interpretation that is produced by PUNC has an overall goodness score calculated, based on the diagnosticity of the features used to produce the interpretation, and the level of plausibility of the interpretation (as was used by the C3 model). This is discussed further in section 4.3.

Using the above mechanism that meshes knowledge of the modifier and head concepts, PUNC can produce a variety of interpretation types that have been described in the literature, such as property-based (e.g., "a hat that has yellow and black stripes"), relation-based (e.g., "a beetle that eats sugar") and conjunctions (e.g., "a bird that is also a pet"). These interpretations account for the vast majority of interpretations produced by people and PUNC provides an efficient mechanism for generating them.

4.3 Improvements on other models

Overall, PUNC is more efficient than C3 as it reduces the amount of processing necessary to produce a set of psychologically plausible interpretations.

4.3.1 Diagnosticity

[1] specified that employing differential levels of knowledge accessibility might improve model performance by limiting the amount of knowledge that needs to be drawn into the interpretation process. However, the C3 model specifies that all knowledge is equally and directly accessible from memory, whereas PUNC encodes diagnostic information of concepts in descending order of importance for each concept. It has been shown that diagnostic features are more available to people than non-diagnostic features ([16], [17]). By processing the highly diagnostic features of the head and modifier first, PUNC produces "better" interpretations first, with poorer interpretations generally occurring later in the processing stages. This more closely reflects how people produce compounds, since communicative goals generally require people to produce good interpretations first [18].

4.3.2 Informativeness

C3 generates all possible interpretations and then decides whether they are informative or not. PUNC only generates interpretations that are considered informative in the first place. It does this by considering whether information is informative before creating an interpretation, rather than performing a post-hoc test as C3 does. For example, in the compound *beetle tar* the concepts of TAR and BEETLE both contain the information that they are black. PUNC checks if this information is already present in the head concept and if so, rejects the use of this information in specifying an interpretation. While the end results may be the same for PUNC and C3 in this case, the means of getting there is not. PUNC offers a much more succinct solution to the pragmatic concern of informativeness.

4.3.3 Plausibility

PUNC looks at plausible interactions between the concepts involved, which contributes to the overall goodness of an interpretation. For example, in compound *ballet mother*, BALLET is represented as a dance, and dances can be performed. The concept MOTHER inherits from both WOMAN and PERSON and so "performs" is one possible feature that a person can have. Because these features of the two concepts mesh perfectly, the interpretation of "a mother who performs ballet" is deemed highly plausible. On the other hand, if the compound being interpreted was *ballet dog*; "dog that performs ballet" would not receive a high a plausibility scoring since dogs do not typically perform ballet, although the interpretation is still possible. Different interactions between the concepts give rise to different levels of plausibility for an interpretation.

When it comes to plausibility, PUNC address the plausibility level of an interpretation when it is being produced (as explained above). In contrast, C3 takes the less economical approach of producing a partial representation for an interpretation and calculating its plausibility based on the extent of its overlap with all stored instances in memory. It then attempts to enhance the overall plausibility of an interpretation by incorporating extra features from the head and modifier concepts, so as to increase the size of the overlap with stored instances. PUNC avoids retroactively assessing the plausibility in two ways. Firstly, PUNC considers the type of interaction between the two concepts at the

point of producing an interpretation. Secondly, it uses the knowledge of the head concept as the core for each interpretation produced. This means that its overlap with existing concepts is maximised from the outset.

In obviating the need for post-hoc checks, PUNC provides an efficient mechanism for integrating plausibility into the interpretation process.

4.3.4 Order of produced interpretation

PUNC processes the information of the head and modifier concepts in descending order of diagnosticity. This means that PUNC is more likely to produce better interpretations first. By the time the least diagnostic features of a concept are used to form an interpretation it is liable that these interpretations will not be considered as good as previous ones, although a high plausibility score may elevate their overall goodness score. On the other hand, C3 generates all possible interpretations and so does not prioritise the order in which they are produced. PUNC's approach is therefore a closer reflection of how people produce compounds, who following pragmatic considerations ([18]), tend to produce better interpretations first.

4.3.5 Knowledge Represented

The knowledge represented in PUNC is very different from other models that employ a set of finite relations that can be used to combine a compound's concepts. For example, the CARIN model ([4], [10]), uses a finite set of relations such as LOCATION, USE, MADE OF. This not only restricts such a model to relation-based interpretations, but meanings that are in fact quite different are formed using the same underlying relation. For example, the "for" relation can be used in a variety of interpretations that have very different meanings (e.g., compare "a magazine *for* workers" to "a treatment *for* backs"). PUNC's use of diverse knowledge allows for more specific interpretations, yet with greater variation in the types of interpretation produced.

4.3.6 Model Performance for Produced Interpretations

By integrating the constraints of diagnosticity, informativeness and plausibility PUNC efficiently generates a list of possible interpretations for compounds. While a model such as C3 produces thousands of interpretations for a single compound, even the interpretations that receive the highest goodness scores may not reflect the interpretations produced by people [1]. This can be partly attributed to limitations of the size of the knowledge base, but this problem applies to any computational model. In PUNC, however, the set of interpretations produced for a given compound generally seems quite sensible and informative. For example, Table 1 shows interpretations produced for the compound *cactus beetle*. While the most highly ranked interpretations seem to be better candidates (e.g., "a beetle that is spiky"), the lower ranked interpretations, while still possible, appear less good (e.g., "a beetle that can photosynthesis"). In PUNC the overall ranking of an interpretation is calculated using both the diagnosticity of the information being incorporated, from the head and modifier, and the plausibility of the interaction between the compound's constituents[2]. This calculation gives slightly more importance to the modifier's information, as this has been shown to have a greater bearing on resultant interpretations ([10], [11]).

Table 1 Interpretations for *Cactus Beetle* produced by the PUNC model

Rank	Interpretation
1	A beetle that is spiky
2	A beetle that is green
3	A beetle that is found in the desert
4	A beetle that can conserve water
5	A beetle that has the shape of a cactus
6	A beetle that eats cactus
7	A beetle that can photosynthesise

Preliminary testing has been carried out to establish whether the interpretations that PUNC produces do in fact reflect those produced by people, and whether the interpretations most frequently produced by people are considered better interpretations by PUNC's ranking.

To examine PUNC's performance we used two sets of interpretations for novel, noun-noun compounds; one from [1] (e.g., *whale* seal, *viper slug)* and one set from [19] (e.g., *carrot bomb*, *bee hat*). In both of these studies, people were asked to provide what they thought would be plausible interpretations for a list of novel compounds. Participants' responses were collated and ranked by their frequency of production.

We took the same set of compounds that were presented to those participants (39 in total) and input them into the PUNC model. For each compound PUNC returned an ordered set of interpretations and representations for each interpretation, which we could then compare to participants' responses.

We found that in 77% of cases the most frequently produced interpretation by people was produced by PUNC as the highest ranking interpretation. For example, for the compound *plate paper*, the interpretation that was ranked highest by PUNC was "paper that is used to make plates", which matched the most frequently produced interpretation by the participants. For some compounds, PUNC's highest ranking interpretation was not the interpretation produced most often by people, but was still among the set of interpretations people produced. For example, *bee*

[2] Goodness Score = (Plausibility / 10) * ((Head Diagnosticity * 0.55) + (Modifier Diagnosticity * 0.45))

hat meaning "a hat that is worn by a bee" was considered the best interpretation by PUNC, but it was only the third most frequently produced interpretation by participants. If we compare all of the interpretations produced by PUNC that were also produced by people, there is a strong correlation between PUNC's goodness score and the frequency of production of participants' interpretations (r =-0.74, N =261)[3].

These initial findings are promising, but more stringent testing is required to establish whether PUNC can be used as a viable model of conceptual combination.

5. DISCUSSION

Overall, the PUNC model offers an efficient model of conceptual combination, and implementation of the Constraint Theory, while still reflecting a diversity of interpretation types that other models lack (see [4]). By meshing diagnostic features of concepts in a way that obviates the need to consider all possible permutations of information, together with the constraints of informativeness and plausibility, PUNC constructs sets of interpretations that reflect not only those produced by people, but also their relative goodness. This increased efficiency seems to better parallel the speed people manifest in their interpretation of novel word combinations. Additionally, [20] have incorporated the additional constraint of compound familiarity to PUNC's existing constraints, which has been shown to be an important factor in explaining people's response times to noun-noun compounds.

As an implementation of the constraint theory of conceptual combination, PUNC makes an important leap from a largely computational level implementation to a more algorithmic-level consideration of conceptual combination. Future work will also consider how the PUNC system can be used to predict the compounds that people produce given particular object descriptions.

ACKNOWLEDGMENTS

The work presented here has been part-funded by a grant from the Irish Research Council for Science, Engineering and Technology.

REFERENCES

[1] F. J. Costello & Keane, M. T. Efficient creativity: Constraints on conceptual combination. Cognitive Science, 24, 2, 299-349, (2000).

[2] F. J. Costello & Keane, M. T. Testing two theories of conceptual combination: Alignment versus diagnosticity in the comprehension and production of combined concepts. Journal of Experimental Psychology: Learning, Memory & Cognition, 27(1): 255-271, (2001).

[3] E. V. Clark & B. F. Hecht. Learning to coin agent and instrument nouns. Cognition, 12, 1-24, (1982).

[4] C. L. Gagné & E. J. Shoben. Influence of thematic relations on the comprehension of modifier-noun combinations. Journal of Experimental Psychology: Learning, Memory and Cognition, 23, 71-87, (1997).

[5] R. J. Gerrig & G. L. Murphy. Contextual influences on the comprehension of complex concepts. Language and Cognitive Processes, 7, 205-230, (1992).

[6] J. A. Hampton. Inheritance of attributes in natural concept conjunctions. Memory and Cognition, 15(1), 55-71, (1987).

[7] J. N. Levi. The Syntax and Semantics of Complex Nominals. New York: Academic Press, (1978).

[8] E. J. Wisniewski. Construal and Similarity in conceptual combination. Journal of Memory and Language, 35, 434-453, (1996).

[9] E. J. Wisniewski. When concepts combine. Psychonomic Bulletin and Review, 4, 167-183, (1997).

[10] C. L. Gagné. Relation-based versus property based combinations: A test of the CARIN theory and dual-process theory of conceptual combination. Journal of Memory and Language, 42, 365-389, (2000).

[11] C. L. Gagné. Relation and lexical priming during the interpretation of noun-noun combinations. Journal of Experimental Psychology: Learning, Memory, and Cognition, 27, 236-254, (2001).

[12] M. T. Keane & F. J. Costello. Setting limits on analogy: Why conceptual combination is not structural alignment. In D. Gentner, K. J. Holyoak, & B. Kokinov (Eds.), The Analogical Mind: A Cognitive Science Perspective. Cambridge, MA: MIT Press, (2001).

[13] F. J. Costello. Investigating creative language: People's choice of words in the production of noun-noun compounds. *Proceedings of the Twenty-Fourth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum, (2002).

[14] F. J. Costello & M. T. Keane. Polysemy in conceptual combination: Testing the Constraint Theory of Combination. Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society, M. G. Shafto, P Langley (Eds.). Hillsdale, N.J.: Erlbaum. pp 137-142, (1997).

[15] Z. Estes & S. Glucksberg. Relevance and feature accessibility in combined concepts. Proceedings of the Twenty-First Annual Conference of the Cognitive Science Society, M. Hahn and S. C. Stoness (Eds.). Hillsdale, N.J.: Erlbaum. pp 149-154, (1999)

[16] E. Rosch. Family resemblances: Studies in the internal structure of categories. Cognitive Psychology, 7, 573-605, (1975).

[17] H. P. Grice. Logic and Conversation. In P. Cole and J. L. Morgan (Eds.), Syntax and Semantics (vol. 3): Speech Acts. New York: Academic Press, (1975).

[18] D. Lynott & M. T. Keane. The effect of context on conceptual combination. (Manuscript in submission).

[19] G. Tagalakis, M. T. Keane & D. Lynott. Making the Constraint Model of concept combination more familiar. In the *Proceedings of the 14th Irish Conference on Artificial Intelligence and Cognitive Science*, Trinity College Dublin, (2003).

[3] The negative correlation is due to lower goodness scores in PUNC referring to better interpretations, while lower scores in participant's responses refer to poorer interp retations.

Temporal Relations in Visual Semantics of Verbs

Minhua Ma and Paul Mc Kevitt[1]

Abstract. Numerous temporal relations of verbal actions have been analysed in terms of various grammatical means of expressing verbal temporalisation such as tense, aspect, duration and iteration. Here the temporal relations within verb semantics, especially ordered pairs of verb entailment, are studied using Allen's interval-based temporal formalism. Their application to the decomposite visual definitions in our intelligent storytelling system, CONFUCIUS, is presented, including the representation of procedural events, achievement events and lexical causatives. In applying these methods we consider both language modalities and visual modalities since CONFUCIUS is a multimodal system.

Keywords: natural language understanding, knowledge representation, temporal relations, visual semantics, language visualisation, verb semantics

1 INTRODUCTION

There are two main kinds of temporal reasoning formalisms in artificial intelligence systems: point-based linear formalisms to encode relations between time points (moments), and interval-based temporal calculus to encode qualitative relations between time intervals [1]. Point-based linear formalisms can represent moments, durations, and other quantitative information, whilst interval-based temporal logic can express qualitative information, i.e. relations between intervals.

A common problem in the tasks of both visual recognition (image processing and computer vision) and language visualisation (text-to-graphics) is to represent visual semantics of action verbs (events), which happen in both space and *time continuum*. Since states and events are two general types of verbs and events usually occur over some time intervals and involve internal causal structure (i.e. change of states), we use an interval-based formalism rather than a point-based formalism to represent temporal relationships in visual semantics of eventive verbs.

First, we begin with background to this work, the intelligent multimodal storytelling system, CONFUCIUS, and review previous work on temporal relations in story-based systems and natural language processing (section 2). Then we investigate various temporal interrelations between ordered pairs of verb entailment using this interval-based formalism in section 3. We turn next to discuss some attributes of interval relations and revise the conventions to indicate directions for causal relationship and backward presupposition in section 4. Next we apply this method in our visual definitions of verbs in CONFUCIUS and discuss its applications in different circumstances such as procedural events, achievement events and lexical causatives in section 5. Following this, comparisons of our method to other work are considered (section 6), and finally section 7 concludes with a discussion of possible future work of adding quantitative elements to the decomposite visual representation.

[1] School of Computing & Intelligent Systems, Faculty of Engineering, University of Ulster, Derry/Londonderry, Northern Ireland, BT48 7JL, email: {m.ma, p.mckevitt}@ulster.ac.uk

2 BACKGROUND AND PREVIOUS WORK

Our long-term objective of this research is to create an intelligent multimedia storytelling interpretation and presentation system called CONFUCIUS, which automatically generates multimedia presentations from natural language input. It employs temporal media such as 3D animation and speech to present short stories. Establishing correspondence between language and animation is the focus of this research. This requires adequate representation and reasoning about the dynamic aspects of the story world, especially about events, i.e. temporal semantic representation of verbs.

2.1 CONFUCIUS

Any multimodal presentation system like CONFUCIUS needs a multimodal semantic representation to allocate, plan, and generate presentations. Figure 1 illustrates the multimodal semantic representation of CONFUCIUS. Between the multimodal semantics and each specific modality there are two levels of representation: one is a high-level multimodal semantic representation which is *media-independent*, the other is *media-dependent* representation which bridges the gap between general multimodal semantic representation and specific media realization and is capable of connecting meanings across modalities, especially between language and visual modalities. CONFUCIUS uses a decomposite predicate-argument representation [7] to connect language with visual modalities as shown in Figure 1. The interval-based temporal logic we discuss in this paper is applied in the decomposite visual representation which is further discussed in section 5. This method is suited for representing temporal relations and hence helping create 3D dynamic virtual reality in language visualisation.

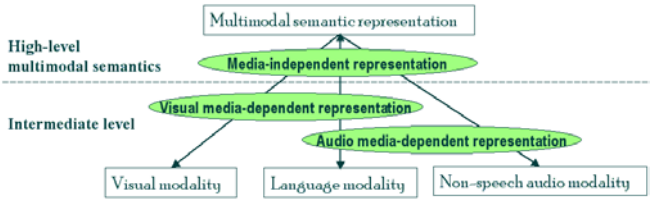


Figure 1. Multimodal semantic representation of CONFUCIUS

Figure 2 shows the knowledge base of CONFUCIUS that encompasses *language knowledge* for the natural language processor to extract semantic structures from text, and *visual knowledge* which includes *object model* and *event model, etc*. The *event model* consists of visual representation of events (verbs) that contains explicit knowledge about the decomposition of high level acts into basic motions, and defines a set of basic animations such as *walk*, *jump*, *crouch* by determining key frames of corresponding rotations and movements of human joints and body parts involved.

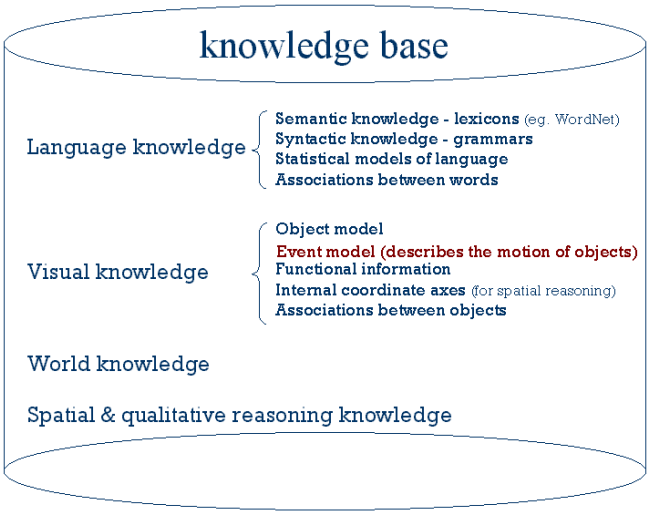


Figure 2. Knowledge base of CONFUCIUS

Here we focus on an efficient temporal representation in *event models* of this knowledge base, exploring how to apply interval relations in modeling the temporal interrelation between the subactivities in one event.

2.2 Previous work on temporal relations

Here we introduce Allen's [1] thirteen basic interval relations (Table 1), which will be used in visual semantic representation of verbs in CONFUCIUS' language visualisation. Allen's interval relations has been employed in story-based interactive systems [8] to express progression of time in virtual characters and handling linear/ parallel events in story scripts and user interactions.

On sentence level temporal analysis within natural language understanding, there are extensive discussions on tense, aspect, duration and iteration. The times involved are *the time of speech*, *the time of situation* and *the time of reference* (i.e. those denoted by time adverbials such as "yesterday", "next Monday"). To represent the relations among them, some use point-based metric formalisms, some use interval-based logic, others integrate interval-based and point-based temporal logic [6] because of the complexity of temporal relations in various situations, for example, the distinction between punctual events and protracted events, achievements and accomplishments [11, 12], stative verbs and eventive verbs, states, events and activities [2]. However, few of these are concerned with the temporal relations at the lexical level, e.g. between or within verbs. In lexical semantics, extensive studies have been conducted on the semantic relationship of verbs [4], but few temporal relations have been considered. The closest work to that presented in this paper was developed about 6 years ago by Badler et al. [3]. They generalized five possible temporal relationships between two actions in technical orders (instruction manuals) domain. In the following sections we investigate temporal relations at the lexical level since this work will facilitate our decomposite visual definitions of verbs in language visualisation.

Basic relations		***Example***	***Endpoints***
precede	x p y	xxxx	$x_e < y_s$
inverse precede	y p^{-1} x	yyyy	
meet	x m y	xxxxx	$x_e = y_s$
inverse meet	y m^{-1} x	yyyyy	
overlap	x o y	xxxxx	$x_s < y_s < x_e \cap x_e < y_e$
inverse overlap	y o^{-1} x	yyyyy	
during	x d y	xxxx	$x_s > y_s \cap x_e < y_e$
inverse during (include)	y d^{-1} x	yyyyyyyyy	
start	x s y	xxxxx	$x_s = y_s \cap x_e < y_e$
inverse start	y s^{-1} x	yyyyyyyyy	
finish	x f y	xxx	$x_e = y_e \cap x_s > y_s$
inverse finish	y f^{-1} x	yyyyyyyy	
equal	x ≡ y y ≡ x	xxxxx yyyyy	$x_s = y_s \cap x_e = y_e$

Table 1. Allen's thirteen interval relations[2]

3 TEMPORAL RELATIONS IN VERB ENTAILMENTS

In this section various temporal relations between ordered pairs of verbs in which one entails the other are studied and their usage in visualisation is discussed.

Verb entailment is a fixed truth relation between verbs where entailment is given by part of the lexical meaning, i.e.

[2] In this table subscript "e" denotes "end point", "s" means "start point".

entailed meaning is in some sense contained in the entailing meaning. Verb entailment indicates an *implication* logic relationship: "if x is true, then y is true" (x⇒y). Take the two pairs *snore-sleep* and *buy-pay* as example, we can infer snore⇒sleep and buy⇒pay since when one is snoring (s)he must be sleeping, and if somebody wants to buy something (s)he must pay for it, whilst we cannot infer in the reverse direction because one may not snore when (s)he is sleeping, and one might pay for nothing (not buying, such as donation). In these two examples, the entailing activity could temporally *include* (i.e. d^{-1}) or *be included in* (i.e. d) the entailed activity. Fellbaum [4] classifies verb entailment relations into four kinds, based on temporal inclusion and other elements between the activities such as causal structure (Figure 3).

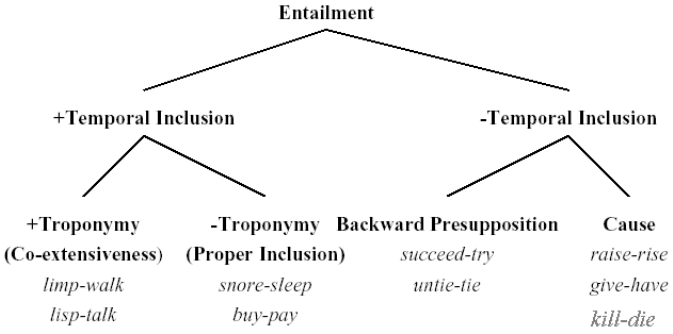


Figure 3. Fellbaum's classification of verb entailment

Troponymy is one important semantic relation in verb entailment [4] which typically holds between manner elaboration verbs and their corresponding base verbs, i.e. two verbs have the troponym relation if one verb elaborates the manner of another (base) verb. For instance, *mumble-talk* indistinctly, *trot-walk* fast, *stroll-walk* leisurely, *stumble-walk* unsteadily, *gulp-eat* quickly, the relation between *mumble* and *talk*, *trot/stroll/stumble* and *walk*, *gulp* and *eat* is troponymy. In CONFUCIUS, we use the method of `base verb + adverb` to present manner elaboration verbs, that is, to present the base verb first and then, to modify the manner (speed, the agent's state, duration of the activity, and iteration) of the activity. To visually present "trot", we create a loop of walking movement, and then modify the `cycleInterval` to a smaller value to present fast walking.

In Table 2 we analyze the possible temporal relations between these verb entailments and give some examples. Notice that the interval relation between a troponym pair of verbs is {≡} (see Table 1), e.g. limp ≡ walk.

The relation set of $\{p,m,o,s,f^{-1},\equiv\}$ may hold in any pair with causal structure (i.e. lexical causatives), between the eventive verb and its result state (either stative verb or adjective), such as *give-have, eat-full, work-getPaid, heat-hot*. Thanks to the productive morphological rules in English deriving verbs from adjectives via affixes such as –en and –ify, these deadjectival verbs, e.g. "whiten", "shorten", "strengthen", "soften", often refer to a change of state or property and have the meaning (*make/become/cause* + corresponding adjective). The temporal relation between the pair of deadjectival verbs and the state of their corresponding adjectives is also $\{p,m,o,s,f^{-1},\equiv\}$. For instance, the possible interval relations set between soften-soft could be `soften` $\{p,m,o,s,f^{-1},\equiv\}$ `soft`. Similarly, the relation set $\{p,m,o,s,f^{-1},\equiv\}$ is also applicable to cognate verbs and adjectives such as *beautify-beautiful* and *clarify-clear*.

Verb entailment relations	*Temporal relations*	*Example*
troponym	{≡}	limp ≡ walk
non-troponym (proper temporal inclusion)	$\{d,d^{-1}\}$	snore d sleep buy d^{-1} pay
backward presupposition	$\{p^{-1},m^{-1}\}$	untie p^{-1} tie ∪ untie m^{-1} tie
cause	$\{p,m,o,s,f^{-1},\equiv\}$	eat p fullUp ∪ eat o fullUp, give m have, build o exist

Table 2. Temporal relations in verb entailments

Eventive verbs, which have internal causal structure and are distinguished from stative verbs on this point according to Gennari and Poeppel [5], are also our main concern in our language visualisation.

4 SOME ATTRIBUTES OF INTERVAL RELATIONS

Reversibility and transitivity are important attributes in temporal reasoning. They provide an algorithm which propagates the temporal relations through a collection of intervals, determining the most constrained disjunction of relations for each pair of intervals which satisfies the given relations and is consistent in time. By adding directions to interval relations we may denote the implication logic relationship between two events.

4.1 Reversibility and transitivity

Reversibility of one interval relation could be defined as: if $\exists R$: act1 R act2, $R\in\{p,p^{-1},m,m^{-1},o,o^{-1},d,d^{-1},s,s^{-1},f,f^{-1}\}$ ⇒ act2 R^{-1} act1, then this temporal relation R is reversible. For instance, untie p^{-1} tie ⇒ tie p untie. All interval relations except ≡ are reversible.

Transitivity of one interval relation could be defined as: if $\exists R$: (act1 R act2) ∩(act2 R act3), $R\in\{p,p^{-1},o,o^{-1},d,d^{-1},s,s^{-1},f,f^{-1},\equiv\}$⇒ act1 R act3, then this temporal relation R is transitive, to wit, the temporal relations between the pairs of intervals can be propagated through the collection of all intervals. For instance, `born p age, age p die ⇒ born p die`. Notice that m and m^{-1} are not in the set of possible transitive relations, because the nature of these two relations is not transitive, i.e. (act1 m

act2) ∩ (act2 m act3) ⇒ ~(act1 m act3). All the other temporal relations {p,p^{-1},d,d^{-1},s,s^{-1},f,f^{-1},≡} must be transitive except o and o^{-1} since act1 o act3 cannot be inferred from (act1 o act2) ∩ (act2 o act3), though it might be true.

The temporal reasoning of the interval relations can be obtained by computing the possible relations between any two time intervals. For instance,

(x d^{-1} y) ∩ (y p z) ⇒ x R z, R∈{p,o,d^{-1},f^{-1},m}

In this case, x could be the activity "buy", y could be the activity "pay" and z could be "consume".

4.2 Revised interval relation conventions

Here we revise Allen's interval logic by adding directions of *implication* logic relationships to it, using R>, <R, or <R>, R∈{p,p^{-1},m,m^{-1},o,o^{-1},d,d^{-1},s,s^{-1},f,f^{-1},≡}. Hence, limp ⇒ walk indicates their troponomy relation, and <≡> indicates synonym relations like speak <≡> say, or same activity from different perspectives such as teach <≡> learn, buy <≡> sell. By this facility we may also use build o> exist to indicate causal relationship (in prediction), and use tie <p untie to indicate backward presupposition (in planning).

5 APPLICATION OF INTERVAL REPRESENTATION

The interval temporal logic discussed above can be applied in a decomposite predicate-argument model of visual definition [7] to represent the temporal relationship between subactivities. The relationship between the definiendum verb and the defining subactivities is temporal inclusion (whether proper inclusion or not), i.e. act1 R act2, R∈{d,s,f,≡}, act1 is part of, or a stage in, temporal realization of act2, and hence it could be one sub-activity in act2's visual definition[3]. ≡ is a special case. If there is only one subactivity in a definition and the relation of this subactivity and its defined verb is ≡ or ≡>, the definition is rather an interpretation than a semantic decomposition, e.g. in the definition slide():- move(), the temporal relation between the subactivity and definiendum is slide ≡ move. Because "slide" is a troponym of "move", i.e. slide ≡> move, we can use "move" to define "slide" but not use "slide" to define "move". The relationship between any subactivity and the verb sense it defines are {d,s,f,≡}:

```
act():-
       subact1(),    ……
       subacti(),    …….
subacti R act, i∈N, R∈{d,s,f,≡}
```

In the proposal of Ma and Mc Kevitt [7] there are two symbols indicating temporal relations between subactivities. The comma separating two sub-activities in sequential order. "act01, act02" means that act02 follows act01. This temporal relation could subsume several relations {p,m,o,f^{-1},d^{-1}} in interval logic, i.e. all temporal relations in x R y while $x_s < y_s$, though p and m are the most frequent relations denoted by comma. Figure 4 shows the visual definition of "call" in Ma and Mc Kevitt [2003] (Figure 4a) and the improved definition using interval logic (Figure 4b). In addition, semicolon is used to indicate the *equal* temporal relation ≡ between two activities which occur simultaneously. "act01; act02" means that act01 and act02 start and finish at the same time. This temporal relation is usable for defining verbs such as rolling of a wheel (Figure 4c, d).

```
call(a):-
  pickup(a, tel.receiver, a.leftEar),
  dial(a, tel.keypad),
  speak(a, tel.receiver),
  putdown(a, tel.receiver, tel.set).
```
a. Visual definition of "call" in Ma and Mc Kevitt [7]

```
call(a):-
 pickup(a,tel.receiver,a.leftEar){p,m,o,f-1,d-1}
 dial(a, tel.keypad) {p}
 speak(a, tel.receiver) {p,m}
 putdown(a, tel.receiver, tel.set).
```
b. Visual definition of "call" using interval logic

```
roll(obj, rollingAngle, newPosition):-
       moveTo(obj, newPosition);
       rotate(obj,0,0,rollingAngle).
```
c. Visual definition of "roll"

```
roll(obj, rollingAngle, newPosition):-
       moveTo(obj, newPosition) ≡
       rotate(obj,0,0,rollingAngle).
```
d. Visual definition of "roll" using interval logic

Figure 4. Visual definition using interval logic

Table 3 compares the original proposal of decomposite visual definitions with the improved version we proposed herein. Note that there is no means to distinguish between the five relations denoted solely by comma in the original proposal. For example, in the definition of "turn" in "turn a vehicle" (Figure 5), the activity of slowdown can also *overlap/include/be finished by* changeGear besides *preceding* or *meeting* changeGear, i.e. slowDown {p,m,o,f^{-1},d^{-1}} changeGear. But there is no way to indicate this by our original representation using ",". It is necessary to distinguish the relation between slowDown and changeGear with the relation between steer and straight, because the latter relation is just a simple *precede* or *meet* relation[4] {p,m} (Figure 5b) whilst the former relation could be anyone in {p,m,o,f^{-1},d^{-1}}. The

[3] act2 is the definiendum verb, and act1 is one of its defining subactivities.

[4] Because the activity "straight" must happen after "steering" finishes.

original representation (Figure 5a) obviously cannot distinguish between them.

Original proposal	*Improved proposal*
act01, act02	act01 R act02,R∈{p,m,o,f^{-1},d^{-1}}
act01; act02	act01 ≡ act02

Table 3. Temporal relations between subactivities

```
turn():-
    … slowDown() ,
      changeGear() ,
    … steer() ,
      straight().
```

a. Original representation of "turn"

```
turn():-
    … slowDown() {p,m,o,f-1,d-1}
      changeGear() {p,m}
    … steer() {p,m}
      straight().
```

b. Improved representation of "turn"

Figure 5. Original and improved temporal representations of "turn"

Another advantage of replacing comma and semi-colon with interval relation symbols is that this method can define multiple temporal relationships in one definition. For instance, one may argue the `eatOut` definition in Figure 6b that in fast food shops people pay first and then get the food they order. Figure 6c includes this circumstance by adding p^{-1} in the relation set between `eat()` and `pay()`, as opposed to defining another event describing `eatOut` in fast food shops.

```
eatOut():-
  bookASeat() ,
  goToRestaurant() ,
  orderDishes() ,
  eat() ,
  pay() ,
  leave().
```

a. Original visual definition

```
eatOut():-
  bookASeat() {p}
  goToRestaurant(){p,m}
  orderDishes() {p}
  eat() {p,m}
  pay() {p,m}
  leave().
```

b. "eatOut" in a restaurant

```
eatOut():-
  bookASeat() {p}
  goToRestaurant(){p,m}
  orderDishes() {p}
  eat() {p,p-1,m}
  pay() {p,m}
  leave().
```

c. "eatOut" in a restaurant/fast food shop

```
eatOut():-
  [bookASeat() {p}]
  goToRestaurant(){p,m}
  orderDishes() {p}
  eat() {p,p-1,m}
  pay() {p,m}
  leave().
```

d. Optional subactivities in definition

Figure 6. Visual definitions of "eatOut"

Either in Schank's scripts [10] or in motion decompostion visual definition [7], there may be some subactivities which are optional in the script/definition. In the eatOut example, `bookASeat()` is optional. We use square bracket to indicate optional subactivities (Figure 6d).

5.1 Punctual events

There are a group of verbs indicating punctual events which never hold over overlapping intervals or two intervals one of which is a subinterval of the other, such as "find", "arrive", "die". Vendler [12] classified them as *achievement* events (distinct from *accomplishment* events), which "occur at a single moment and involve unique and definite time instants". Smith [11] similarly proposes that achievements are "*instantaneous events* that result in *a change of state*." It seems that point-based relations are more appropriate for these verbs. However, some pragmatic approaches [13] deny the semantic distinction between accomplishments and achievements. They think that the length of the event is not a linguistic matter. Pinon [9] introduces the concept of *boundaries* into a temporal ontology for aspectual semantics to analogise achievement events. Boundaries are *ontologically dependent* objects: they require the existence of that to which they are bound.

These considerations are in respect of language modalities. When multimodal representation is concerned, we take visual representation into account, punctual events could also be represented using interval-based relations. As stated in Pinon's boundaries analogy the existence of achievement events depends on the existence of their corresponding accomplishments, we cannot separate these events from their context, e.g. to separate "find" from "search", and "arrive" from "go", in visual representation. In computer games and dynamic visual arts like movies, for example, the event "die" is usually associated with a "falling" movement. When we include context in their visual definitions (Figure 7), these events become intervals rather than moments. Therefore we can declare that all verbs are in time intervals, whether they indicate states, processes, or punctual events. Strictly speaking, the relationships between these punctual events and the subactivities in their visual definitions cover all five possible relations between a point and an interval: *starts*, *before*, *during*, *finishes*, and *after* since these are also the relations between punctual events and their contexts.

```
die():-
  fall().
```

```
find():-
  search(),
  eyesFixedOn().
```

```
arrive():-
  go(),
  stop().
```

Figure 7. Examples of punctual events' visual definitions

5.2 Temporal relations in lexical causatives

Visual definition should also include causative information which helps solve the "frame problem", i.e. to determine the result state following a particular action (the effects of actions). Hence the visual definitions of causative verbs like "kill" must subsume their result states (stative verbs) like "die" (Figure 8).

```
kill(killer, victim, weapon):-
    hit(killer, victim, weapon),
    die(victim).
```

Figure 8. Causative information in visual definition

Moreover, interval relations can represent the distinction between *launching* and *entraining* causation. In the following sentences, (1-4) describe causation of the *inception* of motion (launching causative), whereas (5) describes *continuous* causation of motion (entraining causative). A disjunction set of interval relations between the cause and the effect is adequate to define the difference: {p,m,o,s} for launching causative verbs (from (1) through (4)), and $\{\equiv, f^{-1}\}$ for entraining causatives (see (5)).

Examples	*Temp relation cause-effect*
1) John threw the ball into the field.	{s}
2) John released the bird from the cage.	{p}
3) John gave the book to me.	{m}
4) John opened the door.	{o}
5) John pushed the car down the road.	$\{\equiv, f^{-1}\}$

5.3 Representing actions consisting of repeatable periods

Herein we introduce a facility to represent repeatable periods of subactivities since many actions may be sustained for a while and consist of a group of repeatable subactivities. We use square brackets and a subscript R to indicate this. In the examples of Figure 9 the activities bracketed by $[\ ]_R$ are repeatable.

walk():-
  $[$step()$]_R$.
	hammer(aPerson, aNail):-
	  $[$hit(aPerson,aNail,hammer)$]_R$.

Figure 9. Verbs defined by repeatable subactivities

6 RELATION TO OTHER WORK

Previous temporal representation, analysis and reasoning in syntax (e.g. tense and aspect) and pragmatics is at sentence level, while research on lexical semantics takes few temporal relations into consideration. All temporal relation research within natural language processing is limited within the language modality itself and does not take other modalities such as vision into account. The work we present in this paper brings interval-based temporal logic into visual semantics of verbs at the lexical level and uses this methodology to enhance our decomposite predicate-argument visual definition of action verbs for dynamic language visualisation.

7 CONCLUSION AND FURTHER WORK

Temporal relation is a crucial issue in modelling action verbs, their procedures, contexts, presupposed and result states. In this paper we have discussed temporal relations within verb semantics and proposed an enhanced decomposite visual definition of verbs based on Allen's interval logic. One of the limitations of this temporal representation is lack of quantitative information, which is due to our adoption of the interval-based relations: (1) the durations of activities cannot be specified, though repetition of activities could be indicated by defining one repeatable period and specifying its repeat attribute as shown in the examples in Figure 9; (2) for overlapping events $x\ \{o, o^{-1}\}\ y$, our temporal representation only works when the exact start point of y is unimportant; (3) for events $x\ \{p, p^{-1}\}\ y$, it is hard to relate the distance between the two intervals, i.e. the distance between the end point of x and the start point of y in the case of x p y. Future versions of the decomposite visual representation will introduce quantitative elements to overcome these limitations.

REFERENCES

[1] Allen, J. F. Maintaining knowledge about temporal intervals. *Communications of the ACM* 26(11): 832-843, 1983.

[2] Allen, J. F. and G. Ferguson. Actions and events in interval temporal logic. *Journal of Logic and Computation*, 4: 531-579, 1994.

[3] Badler, N., B. Webber, M Palmer, T. Noma, M. Stone, J. Rosenzweig, S. Chopra, K. Stanley, H. Dang, R. Bindiganavale, D. Chi, J. Bourne, and B. Di Eugenio. *Natural language text generation form Task networks*. Technical report. University of Pennsylvania, 1997.

[4] Fellbaum, C. A semantic network of English verbs. *WordNet: An Electronic Lexical Database*, C. Fellbaum (Ed.), 69-104, Cambridge, MA: MIT Press, 1998.

[5] Gennari, S. and D. Poeppel. Events versus states: empirical correlates of lexical classes. *CogSci2002*, 8-10 August, 2002.

[6] Kautz, H. A. and P. Ladkin. Integrating metric and qualitative temporal reasoning. *Proceedings of the Ninth National Conference on Artificial Intelligence* (AAAI-91), 241-246, 12-19 July, 1991.

[7] Ma, M. and P. Mc Kevitt. Visual semantic representation of events in 3D animation. *Proceedings of The Fifth International Workshop on Computational Semantics (IWCS-5)*, H. Bunt, I. van der Sluis and R. Morante (Eds.), 253-281, Tilburg University, The Netherlands, 15-17 January, 2003.

[8] Pinhanez, C., K. Mase, and A. Bobick. Interval scripts: a design paradigm for story-based interactive systems. *Proceedings of CHI'97*, 287-294, March, 1997.

[9] Pinon, C. Achievements in an event semantics. *Proceedings of Semantics and Linguistic Theory 7*, A. Lawson (Ed.), 276-292, Ithaca, NY: CLC Publications, Cornell University, 1997.

[10] Schank, R. C. and R. Abelson. *Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures*. Hillsdale, NJ: Lawrence Erlbaum, 1977.

[11] Smith, C. S. *The Parameter of Aspect*. Dordrecht: Kluwer, 1991.

[12] Vendler, Z. Verbs and times. *Linguistics in Philosophy*, 97-121, Ithaca: Cornell University Press, 1967.

[13] Verkuyl, H. *A Theory of Aspectuality*. Cambridge: Cambridge University Press, 1993.

Experiments with Reinforcement Learning in Environments with Progressive Difficulty

Michael G. Madden[1] and **Tom Howley**[1]

Abstract. This paper introduces Progressive Reinforcement Learning, which augments standard Q-Learning with a mechanism for transferring experience gained in one problem to new but related problems. In this approach, an agent acquires experience of operating in a simple domain through *experimentation*. It then engages in a period of *introspection*, during which it rationalises the experience gained and formulates symbolic knowledge describing how to behave in that simple domain. When subsequently experimenting in a more complex but related domain, it is guided by this knowledge until it gains direct experience. A test environment with 15 mazes, arranged in order of difficulty, is described. Experiments in this environment demonstrate the benefit of Progressive RL relative to a basic RL approach in which each puzzle is solved from scratch.

1 INTRODUCTION

Reinforcement Learning (RL) is sometimes considered a microcosm of all Artificial Intelligence: an intelligent agent is situated in an environment and must learn to operate successfully within it. The agent explores its environment, receiving rewards and punishments, and seeks to maximize its long-term reward. The appeal of RL is that it is applicable in many domains where it is infeasible to specify explicitly how to perform a task, where proficient humans cannot fully articulate how they do what they are doing. Indeed, Dreyfus and Dreyfus [8] claim that a characteristic of expert level skill in a domain is that one operates at a sub-conscious level, so that it is not generally possible for the expert to specify clear rules that govern how he/she is operating. While this claim is often disputed, the benefit of RL in complex situations is clear: even if one cannot say *how* to do a task well, one can say *whether* it is done well, and provide feedback in the form of a positive or negative reward at the completion of a task. However, a shortcoming of basic approaches to RL is that they discover solutions to individual problems, rather than classes of problems.

Section 2 provides a context for this work by introducing a set of maze problems called the *Theseus and the Minotaur* mazes [1]. A key feature of these mazes is that they exhibit progressive difficulty: a human player would generally find the more complex mazes intractable without having gained experience from solving the less complex ones. Section 3 then describes our Progressive Reinforcement Learning approach, which seeks to augment standard Q-Learning [20] [25] with a mechanism for transferring experience gained in one problem to a new, related, problem. Results of experiments in applying Progressive Reinforcement Learning to the *Theseus and the Minotaur* mazes are presented in Section 4. Then, Section 5 describes related research. Finally, conclusions are drawn and avenues for future research are identified briefly in Section 6.

[1] Department of Information Technology, National University of Ireland, Galway, Ireland. email: michael.madden@nuigalway.ie; hth@eircom.net

2 AN ENVIRONMENT WITH PROGRESSIVE DIFFICULTY

The *Theseus and the Minotaur* maze puzzles were invented by Robert Abbot for a human audience [1]. In these mazes, the player controls Theseus (grey), who must reach the exit without encountering the Minotaur (black). Theseus has five possible moves: Up, Down, Left, Right and Delay. The Minotaur takes two steps for each step taken by Theseus, following a fixed policy: it tries to move first horizontally and then vertically, provided the move takes it closer to Theseus. Abbot first published a single, paper-based maze, and a software developer named Toby Nelson implemented the maze as a Java applet and used a basic form of genetic algorithm to design other maze layouts. The applet now features 15 mazes of various sizes, arranged in order of increasing complexity. (Complexity is determined by the size of the decision space as well as the number of steps to the solution.) Figure 1 shows some of the mazes.

Because of their progressive complexity, these mazes represent an interesting challenge for RL. To adapt them for use with RL, we have developed a Java program in which the original applet executes, substituting for the web browser in which the applet would ordinarily run. This program allows the learner to control Theseus and to get information from the applet about the state of the maze. Unlike human players, the RL agent does not have a high-level view of a maze: it does not know about walls, nor does it know what actions will lead to positive and negative rewards. In typical RL fashion, the agent must discover all of this through trial and error.

As a baseline, tabular Q-learning [25] was applied to the *Theseus* environment. As formulated, each state corresponds to one particular combination of the coordinates of both Theseus and the Minotaur. Theseus has a choice of 5 actions: North, East, South, West and Delay. A lookup table of Q-values is maintained, e.g. for

Maze 1 (6×6) a table of 6480 entries (5 actions × 1296 states) is required. All Q-values are initialised to 0. After an action is carried out, the Q-value for that state-action pair is updated according to the standard Q-learning equation:

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s,a)]$$

where r is the reward received for the action a in state s, α is the step-size parameter, γ is the discount factor and $\max_{a'} Q(s', a')$ is the estimate of the maximum cumulative reinforcement that the agent will receive from the next state s' onwards. A reward of 1 is given for reaching the exit, a reward of −1 is given for reaching the Minotaur and a small negative reward, r_n, is given in non-terminal states. To avoid stalemate situations, when the same state and action path is traversed on two consecutive moves, the play is ended and a reward of −1 is given. Exploration is achieved by following an ε-greedy policy, in which the action with the highest Q-value is chosen with probability $1-\varepsilon$.

Preliminary experiments were carried out to determine good values for the parameters α, γ, ε and r_n. From these, the values found to give the best performance were $\alpha = 0.3$, $\gamma = 0.9$, $\varepsilon = 0.01$ and $r_n = -0.02$. Results of the baseline Q-learning experiments are reported in Section 4.

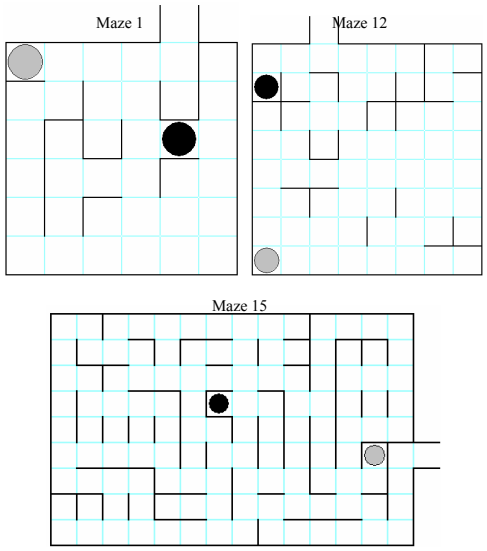


Figure 1. Sample *Theseus and the Minotaur* Mazes [[1]]

3 PROGRESSIVE REINFORCEMENT LEARNING

As stated earlier, basic RL approaches solve individual problems rather than classes of problems; they do not transfer knowledge gained in solving one problem to unseen but related problems. The approach that we introduce here, *Progressive Reinforcement Learning*, provides a mechanism for knowledge transfer. The essential characteristic of Progressive RL is that the agent is informed by alternating cycles of *experimentation* and *introspection*. In the very first experimentation phase, the agent has no prior experience and so solves a relatively simple problem using conventional RL methods. Having mastered the simple problem, it engages in a bout of introspection, in which it analyses the solution(s) found and generates a symbolic representation of the solution, based on high-level features of the problem. In a subsequent experimentation phase with a more complex problem, this symbolic representation is used for those states about which the Reinforcement Learner has no information because they have not been explored. The intuition here is that experience based on simpler problems is only of limited use and may even be misleading, but is better than acting at random when venturing into areas of the current problem state-space that are unknown.

Figure 2 provides a schematic overview of how Progressive RL works. The phases of *experimentation* are conducted using the Reinforcement Learner, and the phases of *introspection* are conducted using the Symbolic Learner. This approach is not tied to a specific form of RL or a specific symbolic learning algorithm. The following sub-sections provide details of how these two major components of Progressive RL are implemented for the experiments reported below in Section 4.

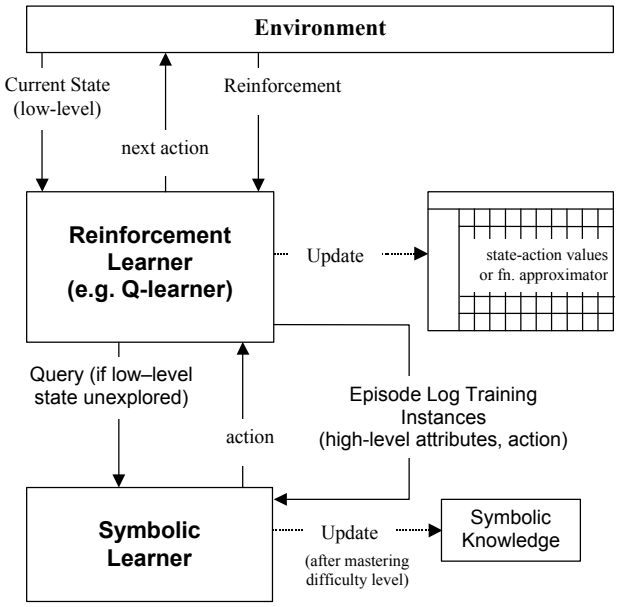


Figure 2. Architecture for Progressive Reinforcement Learning

3.1 The Reinforcement Learner

The current implementation of Progressive RL is based on tabular Q-learning [25]. Table 1 provides details of the algorithm used in the experimentation phase of Progressive RL. The notation used in this description is the same as that used in Section 1. The main additional parameter is *BIAS*, described below. As shown on lines 2 and 14, a vector *Explored(s)* is maintained, indicating whether a state has been visited or not. If the current state is unexplored, a high-level description of it is passed to the Symbolic Learner, which in turn returns an action (lines 6-7). The Q-value for the chosen action is updated using the standard Q-learning approach and the Q-values for all other actions are set to the value *BIAS*. The next time this state is encountered, this *BIAS* value will encourage selection of the same rule-based action again. The need for this is in part due to the use of a default negative reward; in many cases, the chosen action will be updated to a negative value and, if

alternative actions had a value default of 0, that action would be avoided the next time that state is encountered. *BIAS* must not, however, be set to too large a negative number. Otherwise, when the Symbolic Learner chooses a bad action, the Q-values for other actions of that state will never catch up with the Q-value for the bad action and it will always be selected. In Section 4, the effect of *BIAS* is demonstrated experimentally.

Table 1: The Experimentation Phase Implemented in Progressive RL

1	Assume that Introspection Phase has built a set of rules from prior experience
2	$\forall s, \forall a: \ Q(s,a) = 0$, Explored($s$) = false
3	Repeat (for each episode):
4	Initialise s
5	Repeat (for each step of episode):
6	If Not Explored(s):
7	a = SymbolicLearner.ChooseAction(Highlevel(s))
8	$\forall a_i \mid a_i \neq a: \ Q(s, a_i) = BIAS$
9	Else
10	$a = \arg\max_a Q(s, a)$
11	With probability e: a = *random action*
12	Take action a, observe reward r and next state s'
13	$Q(s,a) = Q(s,a) + \alpha[r + g \max_{a'} Q(s', a') - Q(s,a)]$
14	Explored(s) = true
15	$s = s'$
16	Until s is terminal
17	Until stable solution found

Table 2: The Introspection Phase Implemented in Progressive RL

1	Assume that Experimentation Phase has reached steady state
2	Repeat (for a fixed number of episodes):
3	Follow Experimentation Phase procedure (lines 4-16)
4	When each action is taken: Log(Highlevel(s), a)
5	End Repeat
6	Build Symbolic Learner mapping states to actions from logged instances

3.2 The Symbolic Learner

Table 2 outlines the algorithm used in the Introspection Phase. As described earlier, the role of the Symbolic Learner is to propose a course of action in situations that the Reinforcement Learner has not encountered previously, on the basis that experience gained on related (perhaps simpler) problems should be better than acting at random when venturing into areas of the current problem state-space that are unknown. However, the state-space and action policy used by the Reinforcement Learner in a previous problem would not directly correspond to that of the current problem, except where problems are only trivially different. Accordingly, the Symbolic Learner operates in terms of higher-level descriptions of the state space. This achieves two aims: it allows generalisation in associating actions with states in a specific problem, and it allows abstraction in applying knowledge gained in one problem to other related problems.

After an experimentation phase has reached steady state, several episodes are carried out according to the standard experimentation procedure, except that a high-level description of each state and the corresponding action chosen is recorded, as shown in Table 2. These records are then used by the Symbolic Learner to construct rules that map state descriptions to actions. The rules are in effect generated from the learned Q-values of the experimentation phase. This approach is a form of behavioural cloning (see, for example, Šuc [15]) but it does not suffer the problems caused by learning from positive examples only, as the rules are generated from the actions taken when an optimal policy is being followed (steady-state).

Naturally, the two most important issues for the operation of the Symbolic Learner are the learning algorithm and the state descriptions. For the experiments reported in this paper, two learning algorithms are evaluated: Naive Bayes [11] and C4.5 [16]. In both cases, the implementations of these algorithms in the WEKA machine learning package [26] are used. The performance of the two classifiers in this domain is discussed in Section 4.

In future work, it is intended to explore techniques for automatic generation of high-level state descriptions. However, for the present experiments on the *Theseus* mazes, a fixed set of features has been chosen manually:

1. Distance to Minotaur: Manhattan distance
2. Distance to Exit: Manhattan distance
3. Wall to the West: [true,false]
4. Wall to the North: [true,false]
5. Wall to the East: [true,false]
6. Wall to the South: [true,false]
7. Wall to the West of Minotaur: [true,false]
8. Wall to the North of Minotaur: [true,false]
9. Wall to the East of Minotaur: [true,false]
10. Wall to the South of Minotaur: [true,false]
11. Direction to Minotaur: [N, NE, E, SE, S, SW, W, NW]
12. Direction to Exit: [N, NE, E, SE, S, SW, W, NW]

4 EXPERIMENTS AND RESULTS

The objective of these experiments is to compare the performance of Progressive RL with that of standard Q-learning, to see whether its knowledge transfer leads to improvements in later mazes. In these experiments, Progressive RL used cumulative rule building: following the experimentation phase on Maze n, the introspection phase considered experience gained in Mazes 1-n. One experimentation phase corresponded to a fixed number of winning episodes of one maze.

Figure 3 and Figure 4 compare the performance on all mazes of basic Q-learning with two versions of Progressive RL: the first using C4.5 [16] as the Symbolic Learner and the second using Naive Bayes [11]. Figure 3 shows the number of episodes required to solve each maze (win) for the first time, averaged over 20 test runs, where an episode starts in a fixed initial state and ends in a win, lose or stalemate. The same parameter values were used in all experiments: $\alpha = 0.3$, $g = 0.9$, $e = 0.01$, $r_n = -0.02$ and $BIAS = -0.2$.

The results for Maze 1 are identical for all approaches, since Progressive RL has not yet carried out the first phase of introspection. From Maze 2 on, however, both versions of Progressive RL require a significantly smaller number of episodes in getting the first win of each maze. Both versions of Progressive RL show similar levels of speed-up over standard Q-learning on average: 38.5% using C4.5 and 39.1% using Naive Bayes. Progressive RL also performs better than standard Q-learning over the course of entire experimentation phases, as shown in Figure 4, which graphs the average total reward received during each full experimentation phase (4000 wins) on each maze. Similar improvements were found in the number of episodes per win and number of steps taken per win over each experimentation phase.

Additional tests have shown that the experience gained from performing the experimentation phase on a difficult maze (e.g. Maze 15) can be transferred to simpler mazes and outperform standard Q-learning.

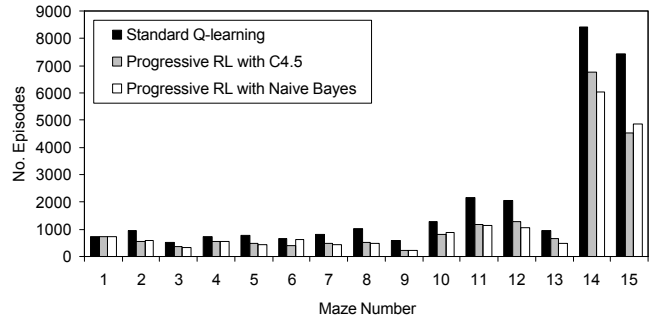


Figure 3. Comparison of Q-learning with Progressive RL: Number of Episodes to First Win

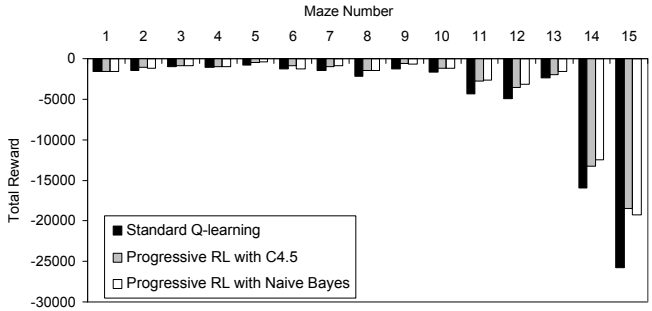


Figure 4. Comparison of Q-learning with Progressive RL: Total Reward per Experimentation Phase

In order to find a suitable value for BIAS, a parameter search was performed: Progressive RL was run on the set of mazes with different values for BIAS and the value –0.2 was found to give the best results. Figure 5 illustrates the effect of BIAS, comparing the performance of standard Q-learning with Progressive RL on a single maze (Maze 15), averaged over 20 test runs. In these results, Naive Bayes is used as the Symbolic Learner, and one version has *BIAS* set to –0.2 and the other version has *BIAS* set to 0. , it may be seen that even without *BIAS*, Progressive RL gains an initial advantage over standard Q-learning in completing the first win. This advantage is lost, however, as the learner without *BIAS* requires more steps to complete the next three wins before it finds an optimal solution. The reason for this is that after the first win in this case, the Q-values of many of the actions chosen by the Symbolic Learner are lower than the other actions of the same state. More learning must therefore be carried out in these states, which has the net effect of delaying the discovery of the optimal solution. In contrast, Progressive RL with *BIAS* set to –0.2 finds the optimal solution after the first win of the maze.

Figure 6 compares the performance of these three learners after a stable solution has been found. This graph, which plots the number steps per win over 100 wins, shows that all the learners settle down to a similar behaviour after a stable solution has been found.

Overall, Progressive RL provides a significant speed-up in the initial discovery of a solution, relative to standard Q-learning. Similar improvement levels were found when Sarsa [20] was used as the Reinforcement Learner (though its base performance was not as good as that of Q-learning).

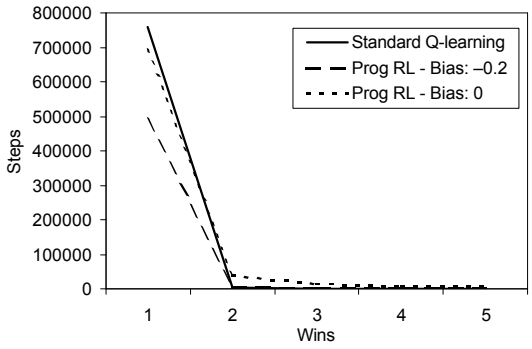


Figure 5. Effect of Bias on Progressive RL: First 5 Wins

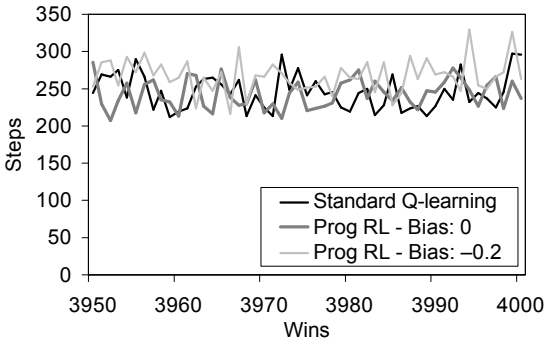


Figure 6. Effect of Bias on Progressive RL: Steady State

5 RELATED RESEARCH

5.1 Symbolic Knowledge in Reinforcement Learning

While Tesauro [22] demonstrated how to achieve state-space generalisation by using a neural network, other researchers have considered symbolic representations of Q-functions. The survey by Kaelbling *et al.* [10] discusses several approaches to generalisation. More recently, Džeroski, De Raedt and Driessens propose Relational Reinforcement Learning [9], in which the Q-function is represented as a logical regression tree [3]. The nodes in logical regression trees are first order logic tests, rather than simple propositional logic tests as used in classical decision trees (e.g. [16]). The main benefits of this approach are that it allows state-space abstraction and can represent structural aspects of the environment.

5.2 Cognitive Skill Acquisition

Breisemeister *et al.* [4] [5] propose a model for cognitive skill acquisition that comprises four basic components: a problem solver, a concept formation component (incorporating a decision tree learning algorithm), knowledge structure (decision tree itself) and a state transformation loop. This system is used in two phases: preparatory and application. In the preparatory phase, the problem solver, an A* search algorithm, is applied to a given number of initial states (a training set). The problem solver generates a number of state-control action pairs that are passed to the learning component, which in turn induces a decision tree that classifies the set of states. During the application phase, the system tries to solve new problems by using the stored decision tree first. If the actual state cannot be classified or a cycle appears, the whole task is passed on to the problem solver. The problem solver then produces a solution, which, as in the preparatory phase, is also processed by the learner.

This work is of interest as it presents a model in which knowledge gained by the problem solver is used in future similar tasks by accessing the current decision tree structure. It is not clear, however, how the model generalises the state space in, for example, the problem of an agent going to a grocery store to buy items from a shopping list [5]. In their approach, each situation or state is a combination of the current grid square (x,y) and the list of goods remaining. The decision tree learner uses these x-y coordinates as attributes, suggesting that knowledge gained is only applicable to identical store layouts.

5.3 Hybrid Reinforcement Learners

CLARION [17] [18] [19] is a two-level model that uses Q-learning at the lower level to gain procedural knowledge and uses of a set of symbolic propositional rules (declarative knowledge) at the upper level. Essentially, the approach is to operate standard RL and symbolic RL (discussed earlier in Section 5.1) in parallel, using a weighted sum to arbitrate between the actions proposed by the two techniques. The approach was applied to a maze problem [17] and to a simulated underwater mine navigation task [18]. It was concluded that CLARION outperformed basic Q-learning in the navigation task [18], and that rule induction facilitated transfer between problems where boundaries or obstacles were changed.

The CLARION architecture, like our Progressive RL architecture, has Q-learning and rule learning components. The principle difference is that CLARION consults both the rules and the Q-values in order to decide every action, combining their recommendations with a weighted sum, whereas Progressive RL uses symbolic knowledge only for unexplored states. Because of this difference, CLARION updates both the rules and Q-values at each step, whereas Progressive RL only constructs symbolic knowledge (in an introspection phase) when an experimentation phase is complete.

Dixon *et al.* [7] propose a more general hybrid model than that of CLARION, in which an *exploration control module* is used to couple a reinforcement learner with one or several sources of prior knowledge. The control module arbitrates between their recommendations, using an arbitration policy that may be varied. They demonstrate the approach in two domains where complex tasks are manually decomposed into simpler sub-tasks, and Q-learners trained on the sub-tasks are used as the prior knowledge for the overall tasks. Their results show that (as might be expected) this decomposition results in significantly faster learning than standard Q-learning.

5.4 Other Techniques for Handling Complexity

One of the motivations behind the work described in this paper is to develop techniques that are able to tackle complex problems more efficiently, by first gaining experience at less complex problems and then transferring that experience to problems that are more complex. Other researchers have investigated alternative approaches to handling complexity in RL. The brief but useful introduction to RL by Russell and Norvig [14] identifies two areas of active research that aim to improve the effectiveness of RL algorithms at tackling complex problems:

1. Reward Shaping, in which rewards are provided for making progress (e.g. Ng *et al.* [13])
2. Hierarchical RL, in which large tasks are broken into subtasks which are tackled at a level where they are tractable; this includes research by Dietterich [6], Sutton *et al.* [21], and Andre and Russell [2].

The SKILLS algorithm of Thrun and Schwartz [[24]] is related to Hierarchical RL. In that algorithm, a skill is a partial action policy that is defined for a subset of all skills (the domain of the skill). Because the skills are defined for a region of the state-space rather than the whole state-space, they are applicable to entire sets of related tasks, rather than individual tasks. However, it is assumed that all related tasks have identical states and actions. Accordingly, the SKILLS approach would not be directly applicable to the *Theseus* environment.

Clearly, these approaches are quite different from the approach of Progressive RL. This means, of course, that it might be possible to gain additional improvements in handling complexity by adopting a mixed strategy, for example combining Hierarchical RL with Progressive RL.

6 CONCLUSIONS & FUTURE WORK

This paper has introduced Progressive Reinforcement Learning, which augments standard Q-Learning with a mechanism for transferring experience gained in one problem to a new, related, problem. Accordingly, this work falls into the area of lifelong learning, as originally identified by Thrun [23].

Progressive RL may be considered to implement a simplified version of the cognitive model of progression from novice to expert, proposed by Dreyfus and Dreyfus [8]. In our approach, an agent acquires experience of operating in a simple domain through *experimentation*. It then engages in a period of *introspection*, during which it rationalises the experience gained and formulates rules describing how to behave in that simple domain. When subsequently experimenting in a more complex but related domain, it is guided by the rules until it gains direct experience. Our experiments demonstrate the benefit of Progressive RL in a sequence of 15 maze puzzles, relative to a basic RL approach in which each puzzle is solved from scratch.

This research is in its early stages. In the current implementation, the RL and Symbolic Learner components are quite basic, and there are many other options to be considered. Also, limitation of the current approach is that the high-level features used by the symbolic learner are manually selected, so we

hope to develop schemes that are more general for producing such features. Another aspect of this approach that may merit future research is that it would support manually specified rules (e.g. "you cannot walk through walls") as well as the automatically-discovered ones. Finally, we intend to develop a suite of different RL test environments, each of which will have progressive levels of difficulty.

ACKNOWLEDGEMENTS

This research has been supported by NUI Galway's Millennium Research Fund. The authors are grateful to Mr. Robert Abbott and Mr. Toby Nelson for their permission to use the *Theseus and the Minotaur* mazes.

REFERENCES

[1] Abbot, R: SuperMazes: Mind Twisters for Puzzle Buffs, Game Nuts, and Other Smart People. Prima Publishing, 1997 (http://www.logicmazes.com).

[2] Andre, D and Russell, SJ: State Abstraction for Programmable Reinforcement Learning Agents. Proc. 18th National Conference on Artificial Intelligence, 2002.

[3] Blockeel, H and De Raedt, L: Top-Down Induction of First Order Logical Decision Trees. Artificial Intelligence, Vol. 101(1-2), pp. 285-297, 1998.

[4] Breisemeister, L, Scheffer, T and Wysotzki, F: Combination of Problem Solving and Learning from Experience, Technical Report 20/95, TU Berlin, 1995.

[5] Breisemeister, L, Scheffer, T and Wysotzki, F: A Concept Formation Based Algorithmic Model for Skill Acquisition, Proc. First European Workshop on Cognitive Modelling, 1996.

[6] Dietterich, T: Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition. Journal of Artificial Intelligence Research, Vol 13, 2000.

[7] Dixon, KR, Malak, RJ and Khosla, PK: Incorporating Prior Knowledge and Previously Learned Information into Reinforcement Learning Agents. Technical Report, Institute for Complex Engineered Systems, CMU, 2000.

[8] Dreyfus, HL and Dreyfus, SE: Mind Over Machine: The Power of Human Intuition and Experience in the Era of the Computer. Blackwell, 1986.

[9] Džeroski, S, De Raedt, L and Driessens, K: Relational Reinforcement Learning. Machine Learning, Vol. 43, pp. 7-52, 2001.

[10] Kaelbling, LP, Littman, ML and Moore, AW: Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research, Vol. 4, pp. 237-285, 1996.

[11] Langley, P, Iba, W and Thompson, K: An Analysis of Bayesian Classifiers. Proc. 10th National Conference on Artificial Intelligence, pp 223-228, 1992.

[12] Michie, D, Bain, M and Hayes-Michie, J: Cognitive Models from Subcognitive Skills. Knowledge-Based Systems in Industrial Control, pp 71-99, 1990.

[13] Ng, AY, Harada, D and Russell, SJ: Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. Proc. 16th International Conference on Machine Learning, 1999.

[14] Russell, SJ and Norvig, P: Artificial Intelligence: A Modern Approach. Prentice Hall, 2003 (Second Edition).

[15] Šuc, D.: Skill Machine Reconstruction of Human Control Strategies, Ph.D. Thesis, University of Ljubljana, Slovenia, 2001.

[16] Quinlan, JR: C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

[17] Sun, R, Peterson, T and Merrill, E: Bottom-up Skill Learning in Reactive Sequential Decision Tasks. Proc. 18th Cognitive Science Society Conference, 1996.

[18] Sun, R and Peterson, T: Autonomous Learning of Sequential Tasks: Experiments and Analyses. IEEE Transactions on Neural Networks, Vol. 9, pp. 1217-1234, 1998.

[19] Sun, R and Merrill, E: From Implicit Skills to Explicit Knowledge: A Bottom-Up Model of Skill Learning. Cognitive Science, Vol. 25, No. 2, 2001.

[20] Sutton, RS and Barto, AS: Reinforcement Learning, an Introduction. MIT Press, 1998.

[21] Sutton, RS, McAllester, DA, Singh, SP and Mansour, Y: Policy Gradient Methods for Reinforcement Learning with Function Approximation. Proc. Advances in Neural Information Systems, 2000.

[22] Tesauro, G: Practical Issues in Temporal Difference Learning. Machine Learning, Vol. 8, pp. 257-277, 1992.

[23] Thrun, S: Explanation-Based Neural Network Learning: A Lifelong Learning Approach. Kluwer Academic Publishers, 1996.

[24] Thrun, S and Schwartz, A: Finding Structure in Reinforcement Learning. Proc. Advances in Neural Information Systems, 385-392, 1995.

[25] Watkins, CJCH: Learning from Delayed Rewards. Ph.D. Thesis, Cambridge University, 1989.

[26] Witten, IH and Frank, E: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann Publishers, 2000.

Probable Interpretations in a Task-Based Dialogue System

Bryan McEleney[1] and Gregory O'Hare[1]

Abstract An empirical approach to interpreting utterances in task-based dialogue is described, where evidence from the utterance and from the dialogue context is combined to estimate a probability distribution over interpretations. The algorithm for the utterance evidence uses nearest neighbour classification on a training set of utterances derived from a corpus, or more usefully, the user at hand. The contextual evidence is based on an abstract belief-desire-intention agent model of the user, in which probabilities for intention formation and plan preferences are estimated. These estimates are also produced from dialogue corpora or from experience of the user at hand. Experimental results for the utterance interpreter show that adaptation to a particular user's training utterances improves recognition accuracy over training on utterances from the general population.

1 Introduction

Designing computers to interact using spoken dialogue is not such an easy matter. The traditional means by which an interpretation is made for a given natural language utterance is to perform a semantic interpretation, whereby compositional rules attached to each grammar rule produce a logical-form in predicate logic, a database query language, or whatever formal representation is useful to the task at hand. Consider this approach applied to understanding a user's request to edit a contact in the contact book of his PDA (Personal Digital Assistant). The following are transcriptions of the first five utterances spoken by a randomly selected user in this situation:

1: "Change a contact in the list"
2: "Edit a contact"
3: "Change a name in the list"
4: "Change a contact"
5: "Change the details of a contact"

There is a lot of variety here, so to provide a semantic interpretation for all of these constructions is a little tricky. Syntactic coverage is not too difficult, indeed a grammar can be constructed that should cover most of the relatively simple constructions seen in PDA interactions, but covering the semantics requires domain specific knowledge and some clever rule coding. The right connections must be made between lexical choices and the concepts they denote and difficult and syntactically ambiguous constructions such as the attachment of "in the list" (utterances 1 and 3) cause problems. Clearly, constructing a rule-based semantic interpreter that covers all of the commands for a PDA application, for each new application that is produced and for each language it is produced for, is not straightforward.

Another important reason why semantic interpretation fails in the real world is that its input isn't always perfect. Users frequently produce utterances that are not covered by the grammar because they are lazy in articulating a correct sentence and because of disfluencies caused by insertion of confusing fragments ("I want to send an email to….let's see….Joe Bloggs"), or false starts ("I want to send….eh….write an email"). Aside from the failings of the user, speech recognition is not yet perfect. If even one word is incorrectly recognised in an utterance, the parsing stage is very likely to fail and no reasonable output is produced by the interpreter – not even a guess.

These are all reasons to worry about how to approach semantic interpretation, but consider one more common phenomenon, that of ambiguity due to underspecification in the utterance. People take short cuts in their language production, expecting the hearer to decode their intention from the context, for instance by the use of anaphora, elliptical expressions or ambiguous illocutionary acts (eg. "Can you send an email?"). Often these can be resolved by looking up the most salient referent or matching expressions in the dialogue history. Other times though, the reference needs to be decoded by considering the most likely intention of the speaker based on some planning knowledge. Consider a footballer's typical utterance "Pass!". Working out an interpretation from this utterance is not so much a matter of processing lexical and structural information, since there is very little, and there may not be suitable expressions in the dialogue history to provide suitable referents. Instead, by considering the player as a rational agent, who most likely intends to score a goal, requires the ball in order to do so, and cannot acquire the ball on his own, the inference that he has requested another agent to "Pass the ball to me" can be drawn.

1.1 A probabilistic model for interpreting utterances

Under conditions of uncertainty from speaker error, underspecification and poor interpretation capabilities of the system, a more reasonable interpretation model is to estimate a probability distribution over interpretation hypotheses, given the evidence of the utterance. These interpretations are called *dialogue acts*[2] (DAs) in the sense that a dialogue act is the combination of illocutionary force and propositional content of an utterance. Formally, the distribution estimate is of:

$$P(I \mid U) \qquad (1)$$

[1] Department of Computer Science, University College Dublin, Dublin 4, Ireland

[2] We distinguish this definition from that of [1] where the dialogue act refers to the illocutionary force alone

Similarly, evidence from an agent model estimates the probability of various DAs as supplementary information. For example, from the hearer's perspective the footballer's intention may be any of a number of tactics that feature in the game, such as moving into defence, obtaining a pass, etc, which then account for the DA. The contextual evidence estimates:

$$P(I \mid C) \tag{2}$$

A possible model for interpretation is one that combines these two sources of evidence to produce an estimate for:

$$\hat{I} = \underset{I}{\operatorname{argmax}}\ P(I \mid U \wedge C) \tag{3}$$

To estimate (3) as a function of (1) and (2), Bayesian updating is used. Applying Bayes' rule and assuming conditional independence of U and C on I produces:

$$\hat{I} = \underset{I}{\operatorname{argmax}}\ \frac{P(I \mid U).P(I \mid C)}{P(I)} \tag{4}$$

The formula is actually an approximation since U and C are not strictly conditionally independent, but it is nevertheless practical in that separate estimates can be combined from utterance and contextual evidence, and has been shown in analogous problems, such as in combining language and acoustic models in speech recognition, to perform well.

It might seem impractical to individually compute probability distributions for the agent model and the utterance evidence, since many utterances will be so ambiguous that an unmanageably large distribution of utterance-based DA hypotheses is produced. In such a situation, a suitable approach might be to generate the agent model distribution first and then match each hypothesis against the utterance. For instance in the football example, "Pass" taken out of context has a range of word senses, and being a transitive verb a large number of potential subjects and objects, optional prepositional phrases and furthermore a range of illocutionary force labels. On the other hand, working from the agent model hypotheses, the logical forms could be matched one by one against the utterance, so that a consistency measure can be obtained. In the particular example, the logical forms could be checked for predicates that match one of the senses of "Pass". Such a matching procedure depends on producing a logical form representation of the utterance however, which we argue is difficult to obtain. An alternative tack is to assume that the domain of the dialogue is constrained, so that there can only be a reasonably small number of DA classes. Such an assumption is valid in the kind of task-oriented dialogue that one associates with PDA interactions, but might not be in a natural language database query interface or for a model of free conversation. In fact for the PDA application twenty-three basic DAs were identified for an email and contacts application. Each class corresponds with a tuple consisting of the illocutionary type (**request**, **inform**, etc), and the main predicate of the utterance. Some noun phrases which can identify a broad number of objects (people, dates etc) are ignored when performing the classification, since they would cause an unmanageable explosion in the number of classes under consideration. Instead, they are replaced by tokens such as <person> and <date> before interpretation, and then a postprocessing step is performed in which, once the correct class is identified, variables in an associated logical form are bound to the objects denoted by the noun phrases.

In the following section, we describe how the estimate for distribution (1) may be obtained using a machine learning algorithm, and then describe a plan recognition algorithm for estimating (2). Each algorithm depends on empirical data, in particular, the utterance interpreter can learn from a particular user's habit in their turn of phrase, and the plan recogniser learns their usual intentions and planning habits to generate better expectations.

2 Learning to classify utterances

The assumption that semantic interpretation is a matter of classification indicates an approach the problem. Instead of writing interpretation rules, a method that covers a broad and varying range of labeled example sentences is preferred that can be trained automatically to classify new utterances. Using such an algorithm has a powerful consequence in that as well as training on corpus data from the general population before deployment, the system can be trained on the particular user at hand, adapting to his particular turn of phrase by recording his dialogue utterances and adding them to the training set.

The chosen learning algorithm is that of nearest neighbour classification [2]. This is a very simple approach, whereby the target utterance is compared with a corpus of training examples, and the class assigned is that of the most similar utterances in the training set. The question of similarity is also simply addressed by assuming semantic similarity based on similarity of structure and word content. This similarity is computed using the Levenschtein distance (edit distance) between the word strings of the example and the target, defined as the minimum number of word additions, deletions or replacements necessary to transform one string into another. The use of edit distance is appealing since it responds gracefully to word recognition error from the speech recogniser, and it can cope with false starts and filled pauses since the extraneous words will add roughly the same distance increment to each training example. It does however fail to generalise from examples. For instance, given training examples "Write an email" and "Compose a letter", no generalization can be made so that

"Write a letter" is well classified. More than just the nearest neighbour is used, so that noise is smoothed out, but primarily to smooth the resulting probability distribution. Without smoothing, classes that are highly ranked by the agent model distribution may be discounted altogether if a score of zero is assigned by the utterance-based classifier (by equation 4). This can happen for example with a poorly formed input string such as "Email", which may score highly as the class for checking for new email based on the agent model, but may easily score zero since there are many candidate classes based on the utterance evidence alone, such as composing an email, or sending an email. The probability mass is partitioned by allocating most of it to the closest matches, a small part to matches that are one edit further away and a smaller amount still to matches that are two edits further away.

3 The Agent Model

The agent model is used to perform plan recognition, in the sense of abductively explaining a set of actions [3,4]. Once an explanation is obtained, one can predict the next action of the agent using planning rules. For instance, my putting a key in the lock to my office can be explained by my intention to enter the room, which in turn can be explained by my intention to get some work done today. Given such an explanation, a reasonable inference about my next action is that I will probably read a book or do some writing. While plan recognition is useful for predicting physical actions in a multi-agent situation such as in a board game or in when someone politely opens the door for you when you are laden with shopping bags, it can be extended to predict spoken acts (speech acts). Such an extension requires that speech acts are axiomatised as action schemata in the planning formalism. The formalism adopted is that of hierarchical planning [5] which is an extension of STRIPS planning [6]. Hierarchical planning proceeds by repeatedly decomposing actions into sets of subactions. The knowledge used in constructing a hierarchical plan is a set of action schemas which specify parameters, applicability constraints, preconditions, addition and deletion effects, and a set of decompositions for each action.

The schemata for spoken actions are based on those set out by Appelt [7] in his formulation of a system that plans sentences according to input goals, and correspond with our dialogue act classes in that each action type corresponds with illocutionary force, and the action's parameters with propositional content. There are two basic actions, the first of which is **inform**, used where one agent intends that another adopts a belief:

```
Schema

        Name:           inform
        Agents:         { S, H }
        Parameters:     P
        Applicability:          {}
        Precondition:
        {
                bel(S,P),
                intend(S,adopt-bel(H,P))
        }
        Effects:
        {
                bel(H,
                intend(S, adopt-bel(H,P))
                bel(H,P)
        }
```

This schema captures conditions of agent benevolence. First, a sincerity condition ensures that S always speaks the truth, so while it might be reasonable to tell lies, our agent always believes what it informs. Second, it is assumed that the hearer is gullible, always adopting P as a belief no matter what other evidence is available. Two other variations of inform appear in the schema set, namely **informref**, whose purpose is in dealing with wh-questions, and **informif**, which corresponds with yn-questions of propositional content P, by answering with either P or ¬P.

The other main speech act schema is that of request, whose main effect is that the hearer takes an action. It is defined as follows:

```
Schema

        Name:                   request
        Agents:                 { S, H }
        Parameters:             A
        Applicability:          {}
        Precondition:
        {
                intend(S,A)
                intend(S,
                        adopt-intention(H,A))
                bel(S,capable(H,A))
        }
        Effects:
        {
                bel(H,intend(S,
                        adopt-intention(H,A)))
                intend(H,A)
        }
```

Again there are benevolence assumptions made here, particularly that S is rational in always requesting actions that it wants to achieve, and that H is fully cooperative, in the sense that it will commit to whatever is asked of it.

Finally, questions are included. These are reducible to informs and requests, in that their schema is matched by chaining the subactions, but the schemata and the proof are omitted:

```
wh-question( S,H,λxP(x)) ->
        request(S,H,informref(H,S,λxP(x))

yn-question(S,H,P) ->
        request(S,H,informif(H,S,P))
```

In the agent model, the speaker is taken to be capable of these spoken actions and possibly a set of physical actions, and there is a model of the speaker's plan

library available to the hearer which allows inference of intentions and prediction of further action in the dialogue.

3.1 Cooperative attitudes

Some plan recognition methods make an assumption of keyhole recognition, in that the agent being observed is not aware of that observation and consequently does not construct plans with an expectation of complementary action from another agent. Conversely, intended recognition is considered here, since the speaker attempts to communicate with the system and is aware that the system will recognise and act upon his intentions according to its capabilities and in a cooperative manner. The fact that the agent attempts communication at all provides important initial evidence. First, the plan recogniser can infer that the agent has some particular underlying intention. Second, it can infer that there is no plan available that the agent is capable of executing through its own action- instead the agent's intention involves acts that it wants the recogniser to perform, thus the plan must be a multi-agent plan that must be coordinated between the two agents. Third, the agent recognises that the recogniser's purpose is to cooperate with the agent's intention, and chooses acts that identify relevant intentions. For instance, a PDA user is hardly likely to say "I'm going to walk the dog, and then I want to send an email". Conforming with Grice's maxim of quantity, the recogniser can expect that any posited intention that is too ambiguous can be rejected in favour of another, since it is unreasonable for the user to underspecify his intention. One particular phenomenon in dialogue is that the agent will often specify his underlying intention before dealing with a task in detail, since this information has a strong disambiguating effect.

3.2 A Belief-Desire-Intention (BDI) model

In the previous discussion about the plan schemas, reference was made to the modal operators **bel** and **intend,** which are mean that the agent believes a proposition, and that the agent intends an action [8]. A *mental-state* is defined with sets of propositions representing the belief and intention sets. Each of these sets is subject to constraints of consistency in that incorporation of new beliefs and intentions might cause retractions of older ones. Belief retractions may be rationalised by positing an action occurrence whose effect negates the agent's former belief. The recogniser may infer further beliefs using the standard axiom set S5. The mental state is separate from static properties of the agent, in particular the agent's knowledge about constructing plans – the plan schemata, and the agent's capabilities, which represent the actions the agent can take.

The agent is modelled as a rational agent in that it constructs a tree of intentions, using decomposition rules from the plan schemata, that lead the agent from the possible world states that satisfy its current beliefs to ones that satisfies some measure of state utility. This utility function is known as the agent's desires, which are usually modelled as a (possibly inconsistent) set of desirable propositions. It is usual to add the modal operator **desire** to complete the BDI model. However this model of desire is impractical for our purposes. For a start, nothing is said about which consistent subset of the desires is chosen by the agent from which to generate intentions – there may be several uncertain alternatives. Second, agents rarely discuss desires in dialogue since they do not provide the same strength of evidence about the task solution structure as discussion of intention does. Nor are desires considered relevant. For instance, if I ask you for the spanner, you may infer my intention to make a repair of some sort and cooperate with that intention, but there is a limit to the number of levels of explanation that are relevant or indeed possible to infer. For instance, you might easily infer that I want to repair my bike but experience an explosion of tenuous explanations about why I want to travel by bike. Another problem with the desire model is that intentions can form from a third party **request**-ing an action as much as from the agent's personal desires. Unless the recognising agent models the third agent as well, such intentions cannot be explained. There must then be some horizon of explanation that separates the tree of intentions into a disconnected forest, each with a top-level explanation at the horizon level. Our alternative model for intention formation is thus one of chance top-level intention adoption. In task-oriented dialogue, a trained Markov model can accurately model this process. Top-level intentions are adopted in sequence over time and are conditionally dependent on previous adoptions:

$$P(I \mid I_1, I_2, I_3, ...) \approx P(I \mid I_1, I_2, ..., I_{n-1}) \quad (5)$$

With an intention set that is restricted similarly to our restriction on the dialogue act set (by abstracting away terms from propositions for example), such a model is trainable using n-gram frequency counts on intention occurrences given the previous n-1 intention occurences. The training set for the model can be offline corpus data, or more accurately, data collected online from a specific user. So, if John usually reads and sends emails first thing in the morning, and then makes some phone calls, his top-level intention generating activity can be appropriately covered by the n-gram. In addition to the n-gram model, a belief-intention consistency rule filters candidate intentions, stating that the agent model must not contain a belief that the intention's effects are currently true.

3.3 Interpreting the dialogue

Dialogue interpretations can conveniently be represented as graphs, with two types of node, and several types of edge. State nodes represent propositions, particularly about intentions and beliefs. Action nodes represent the spoken and dialogue actions that have already occurred in the task. Edges represent relationships between nodes. The graph represents actions and mental state propositions of all of the

agents involved, so that relationships between agents can be represented. There is a directed edge (T-edge) between each action to represent their temporal order. Proposition nodes are related to action nodes by way of their being on the add list of an action schema's effects (A-edge), or as preconditions to an action (P-edge), since to have executed an action, the preconditions must have been true. A special node connects intention proposition nodes to actions, to represent the motivational link between an agent having an intention and then acting upon it (M-edge). Proposition nodes may generate other proposition nodes by means of inference (I-edge). For a given dialogue history, a dialogue graph can be constructed by adding the physical and spoken actions of the dialogue to the graph in order. As each is added, their preconditions are added too, through P-edges. Deletion effects serve to remove state nodes. Addition effects add state nodes. Once the whole dialogue is added, the collection of state nodes reflects what is mutually known by the agents about one another's mental state and the physical state.

3.4 Generating expectations

The plan recogniser is required to produce an expectation of the next dialogue act. The expectation generation algorithm works by generating a set of possible extensions to the interpretation graph by one of two operations: introducing top-level intentions and action decomposition based on plan schemata. Each of these operations is probabilistic in that there are several choices in top-level intention introduction and several choices of plan decomposition. The probabilities are given by the n-gram for intention introduction, and by frequency counts for the action decompositions, learned from offline corpus data or from repeated usage by the same user. In the `generate` procedure below, a set of leaf graphs is produced by branching on decomposition choices from the root graph until atomic actions are reached. This is done by leftmost expansion so that the earliest atomic action is found. At each point, decompositions are checked for precondition satisfaction by checking the appropriate proposition nodes. Before generating an atomic action, it is ensured that all preexisting actions in the graph have been accounted for in the plan structure. If decomposition fails to reach any atomic actions, the intention at hand must be complete, so a top-level intention is introduced and the algorithm tries decomposition again. Each new graph produced by either procedure inherits the probability of its parent multiplied by the probability of the choice that produced it. Finally, the required DA distribution estimate is extracted by plucking the atomic action from each of the leaf graphs and associating it with the graph's assigned probability.

```
Procedure generate(in: Graph, out: Estimate)

1  Graphset <- { InitGraph }
2  Graphset <- decomp(GraphSet}
3  if Graphset = {}
4     GraphSet <-
            introduce-intention(GraphSet)
      goto 2
5  Estimate = make-estimate(GraphSet)
```

In the algorithm there are some simplifying assumptions. First, it is assumed that action decompositions are totally ordered. To represent partial ordering, a set of totally ordered decompositions is needed. However, it is assumed that users skew towards certain orders even when the plan does not logically require that order, and so a total order reflects that skew. Second, we assume that the user is *totally focused*, in that they will never act upon a second intention before the first one is satisfied, and similarly will not "climb out of" a decomposition subtree before all its actions are complete [9]. Such an assumption is not always valid – consider a user pausing while writing an email to take a phone call. On the other hand, such events are relatively rare, and difficult to predict anyway, so the complexity of the algorithm is improved by only expanding one active top-level intention at a time, and performing leftmost decompositions only. If a totally focussed expansion fails to account for the dialogue history, the system must fall back to an unfocussed explanation.

4 Implementation and Results

The dialogue system has been implemented in PROLOG using the SPHINX speech recogniser, and so far, experimental results are available for the utterance classifier. The experiment involved users speaking utterances in a mock-up of an email and contact book application. A program to generate random tasks was used to guide the user. Each user utterance therefore had a pre-determined class, and the user had to think of an utterance corresponding with the class at hand. There was a total of 20 classes, but there was a slight skew in their distribution. The classification perplexity (the equivalent number of evenly distributed classes) was calculated to be 15.1. The most prevalent class occurred with a probability of 0.212, which is the baseline accuracy corresponding with the accuracy achieved just by blindly guessing that class every time.

Four subjects recorded 250 utterances each, a total of about four hours of dialogue. Accuracy results were obtained for each user by setting aside 80 utterances for testing. This gives a total test set size of 320 utterances. Two variables were adjusted in the experiment corresponding with the amount of training data obtained from other users and the amount of training data obtained from the user in question. For each user the system was trained on one, two, and then three other users' data (called *offline* training), and within these groupings, the system was trained on 0%, 50% and 100% of the remaining 170 of the user's own examples (*online* training).

The results shown in figure 1 correspond with the theoretical notion of the shape of the learning curve. Consider that the likelihood of an accuracy improvement by adding a training example is at most inversely proportional

to the number of training examples, n, already in the training set, since it must be closer than n other examples to better classify the target. Integrating this over n gives us the learning curve function log n + C. Any extrapolation of the results shown will thus be roughly logarithmic, but also must be convergent as the training set becomes saturated.

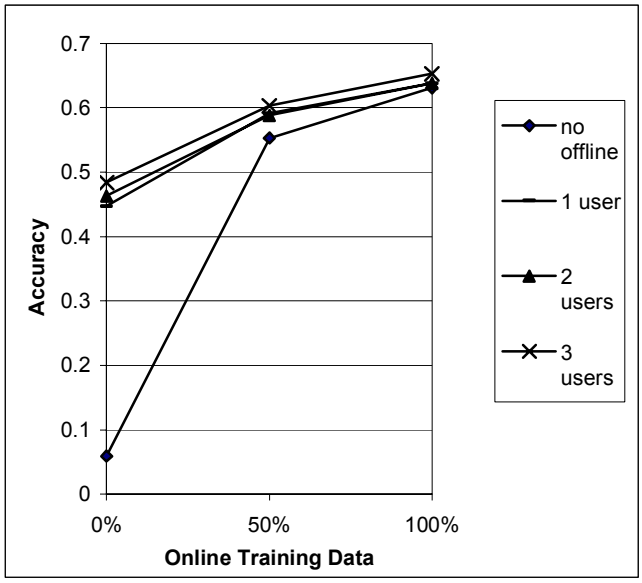


Figure 1. Classification Accuracy

The results show that offline training is rapidly convergent. With offline data from three users, an accuracy of 0.484 is achieved. However, online training on the user's own data raises this accuracy to 0.653. Since each test instance is a binary decision of whether the classification is correct or incorrect, the mean accuracy follows a binomial distribution. From this, the 95% confidence interval for the system trained on all of the offline and online data is [0.601, 0.705], demonstrating that online training offers a statistically significant performance gain. This also indicates that individual users do have a particular habit in their turn of phrase, and suggests that an implementation of a machine learning utterance recogniser for PDA applications can benefit greatly from online training. While it was not performed due to limited processor time, cross-validation could have allowed testing on all 1000 utterances rather than 320, leading to a more accurate results.

5 Conclusions and future work

It was shown that generating interpretations for utterances in task-oriented dialogue, can be reasonably be approached from a probabilistic perspective, relying on empirical data to learn both the user's planning habits and the way in which they phrase their utterances. Experimental results showed that such adaptation improves recognition accuracy in utterance interpretation.

Work is continuing on the plan recognition model with the aim of producing an integrated interpretation system. Combined evidence should then improve classification accuracy. A pleasant by-product of having the plan recogniser in place is that planning the system's dialogue turns is a straightforward matter of selecting a rational utterance or action by using the same planning model, and so a dialogue control module can easily be introduced. With the addition of an utterance generation component, a complete system will be available. It will be possible to evaluate the system on standard task-based dialogue corpora, such as the ATIS (Airline Travel Information Service) corpus so that comparisons can be drawn with other approaches [10]. With adaptation to diverse tasks in mind, apart from writing the plan library and semantic classes and implementing the physical actions, the system is domain-independent.

References

[1] M. Core and J. Allen. Coding Dialogues with the DAMSL Annotation Scheme.*Working papers of the American Association for Artificial Intelligence Fall Symposium on Communicative actions in Humans and Machines* (1997).

[2] T. Mitchell, *Machine Learning*, McGraw-Hill, 1997

[3] N. Lesh, C. Rich and C. Sidner. Using Plan Recognition in Human-Computer Collaboration. *Proc. International Conference on User Modelling* June 1999.

[4] H. Kautz, A Circumscriptive Theory of Plan Recognition. in Cohen, P.R, Morgan, J. and Pollack, M. E., editors, *Intentions in Communication.* MIT Press, Cambridge, Massachusetts, 1990.

[5] E. Sacerdoti Planning in a hierarchy of abstraction spaces. *Artificial Intelligence* 5(2):115 – 135, 1974

[6] R. Fikes and N. Nilsson STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving *Artificial Intelligence* 2(3-4) 189 – 208, 1971

[7] Douglas Appelt, *Planning English Sentences*. Cambridge University Press, 1985

[8] P. Cohen and H. Levesque Intention is Choice with Commitment *Artificial Intelligence*, 42:213--261, 1990.

[9] B. Grosz and C. Sidner Attention, Intention and the structure of Discourse *Computational Linguistics* 12(3) 175-204, 1986

[10] H. Ng and J. Zelle. Corpus-Based Approaches to Semantic Interpretation in Natural Language Processing. *AI magazine* 18(4) : 45 – 64, 1997

Mixed-Initiative Intelligent Systems for Classification and Diagnosis

David McSherry[1]

Abstract. While the importance of mixed-initiative interaction in intelligent systems is increasingly recognised, ensuring that the potential benefits are achieved remains a challenging task for system designers. Issues to be addressed in intelligent systems for classification and diagnosis include how to deal with the problem of incomplete data, take account of information that the user chooses to volunteer, and provide meaningful explanations of the reasoning process. We present new techniques for addressing these issues in mixed-initiative intelligent systems for classification and diagnosis based on decision trees and their implementation in a prototype intelligent system called CONFIRM.

1 INTRODUCTION

The importance of intelligent systems having the ability to support mixed-initiative interaction is increasingly recognised by artificial intelligence researchers [1,2,5,9,12]. Potential benefits include involving the user more closely in the problem-solving process [14] and adapting to the needs of users with different levels of experience [8]. Problem-solving efficiency may also benefit by making use of the knowledge and expertise that users can often bring to bear [12]. In fault diagnosis, for example, the user may know from experience of similar problems which symptoms are most relevant.

Dialogue features typically associated with mixed-initiative interaction in intelligent systems for classification and diagnosis [12,14] are listed below.

Volunteering Data. The user can volunteer data at any stage without waiting to be asked.

User in Control. The user can decide when the system has the initiative, and recapture the initiative at any stage.

Tolerating Incomplete Data. The user can decline to answer any question.

Updating Data. The user can change her answer to a previous question at any stage.

Intelligent Question Selection. When given the initiative, the system can select questions that are, in some sense, most useful.

Explanation of Reasoning. Before answering any question, for example, the user can ask why it is relevant.

User-Specified Goals. The user can select a target outcome class (or diagnosis) to guide the selection of relevant questions.

[1] School of Computing and Information Engineering, University of Ulster, Coleraine BT52 1SA, Northern Ireland

Examples of case-based reasoning (CBR) tools that support some or all of these features include Adaptive Place Advisor [9], CBR Strategist [12], ExpertGuide [16], and NaCoDAE [1]. In this paper, we focus on mixed-initiative intelligent systems for classification and diagnosis based on decision trees. However, our discussion is not restricted to CBR approaches. Although decision trees are often used to guide the retrieval process in CBR, we assume that the system does not have access to the data, if any, from which the decision tree was constructed. For example, the decision tree may have been constructed manually with the assistance of a domain expert. Nevertheless, the techniques we present are applicable to CBR systems in which retrieval is based on a static decision tree, for example to keep response times within acceptable limits [4,18].

In Section 2, we present techniques for building mixed-initiative intelligent systems based on decision trees. In Sections 3 and 4, we present an implementation of the techniques in a prototype intelligent system called CONFIRM. Our conclusions are presented in Section 5.

2 INTELLIGENT DECISION-TREE SYSTEMS

Decision trees are increasingly used in intelligent systems for fault diagnosis and e-commerce applications [4,7,13] Using a decision tree to solve a given problem is simply a matter of following the path determined by the user's answers to questions in the decision tree until a leaf node is reached and reporting the conclusion at the leaf node. However, issues to be addressed in mixed-initiative intelligent systems based on decision trees include how to deal with the problem of incomplete data, take account of information that the user chooses to volunteer, and provide meaningful explanations of the reasoning process.

The example decision tree that we use to illustrate the discussion is based on the contact lenses dataset [6]. This well-known dataset is based on a simplified version of the optician's real-world problem of selecting a suitable type of contact lenses (none, soft, or hard) for an adult spectacle wearer. Our example decision tree, shown in Figure 1, was induced from the contact lenses dataset with information gain [15] as the splitting criterion.

Figure 2 shows the basic components of a mixed-initiative intelligent system for classification based on a decision tree. Allowing the user to volunteer data at any stage means that before asking any question, the system must check if the user has already volunteered the answer. A repository for problem-specific data

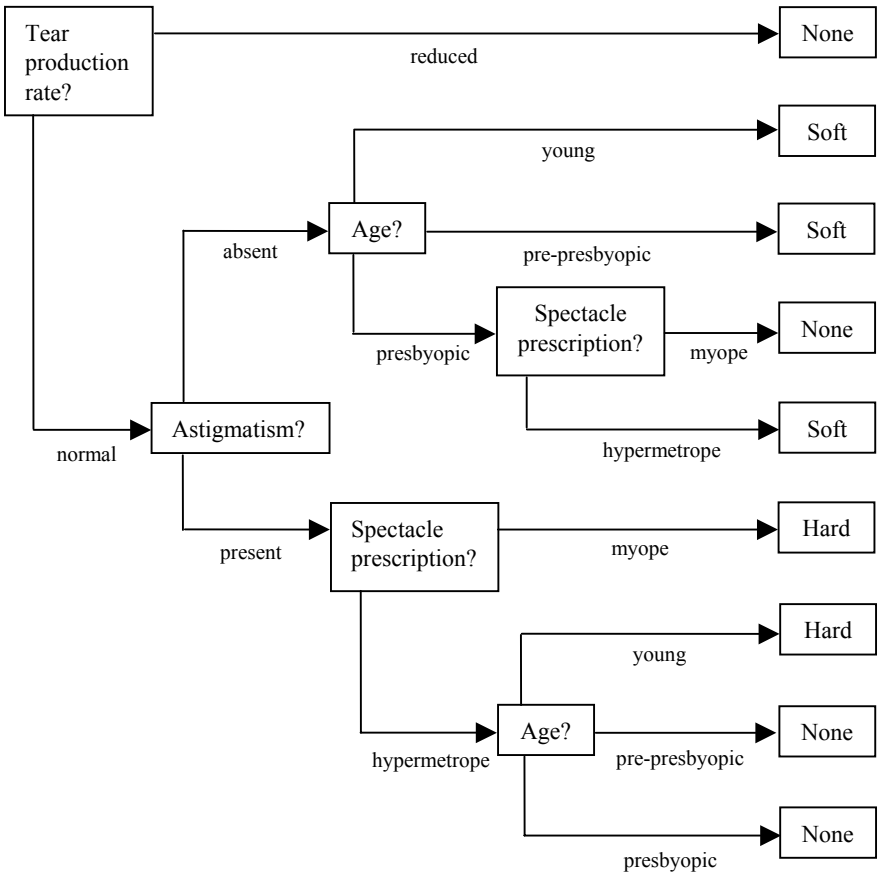


Figure 1. Example decision tree based on the contact lenses dataset [6].

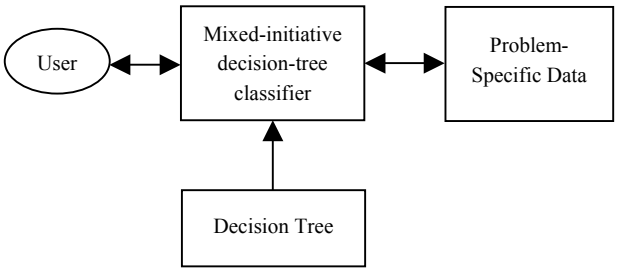


Figure 2. Components of a mixed-initiative intelligent system for classification based on a decision tree.

reported by the user (cf. the working memory in an expert system) is therefore essential. We also assume that the system can explore all paths in the sub-tree starting at a given node of the decision tree, for example to identify all conclusions that can be reached from the given node.

2.1 Effects of volunteered data

A basic algorithm for classification based on a decision tree, partially modified to support mixed-initiative interaction, is shown in Figure 3. The modified algorithm ensures that the user is never asked a question if the answer is already known. However, further refinement of the basic algorithm is needed to enable the system to take full advantage of the effects that volunteered data may have on the decision tree. For example, consider how the example decision tree is affected if it is known in advance that the age of the patient is young. First, the upper and lower sub-trees beginning with the question 'Age?' can be replaced by leaf nodes labelled 'Soft' and 'Hard' respectively. However, it is now clear

```
algorithm classify
begin
    start at the root node of the decision tree
    repeat
        if the answer to the question at the current node is known
        then select the child node that corresponds to the answer
        else begin
                ask the user the question at the current node
                select the child node that corresponds to her answer
                store the answer in the data repository
             end
    until a leaf node is reached
    report the conclusion at the leaf node
end
```

Figure 3. Basic algorithm for classification based on a decision tree, partially modified to support mixed-initiative interaction.

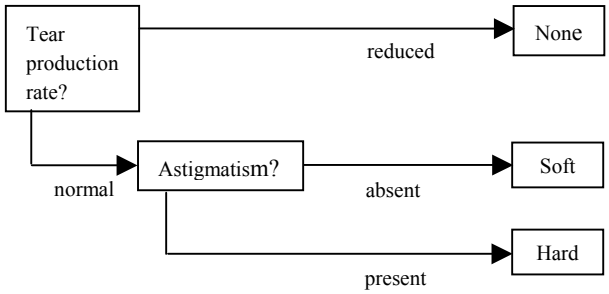


Figure 4. Simplified version of the contact lenses decision tree when the age of the patient is known to be young.

that the spectacle prescription cannot affect the conclusion, so we can eliminate this question from the decision tree. The simplified decision tree is shown in Figure 4.

While the potential benefits in terms of problem-solving efficiency are clear, the decision-tree classifier in Figure 3 is unaware of the simplifying effects of the volunteered data and will needlessly ask the user about the spectacle prescription if the tear production rate is normal and astigmatism is present. The technique we have developed to address this issue relies on *look-ahead* search but requires much less computational effort than simplifying the entire decision tree. Our look-ahead technique requires the intelligent system to perform the following additional steps at each non-leaf node:

Step 1. Identify the outcome classes at all leaf nodes that can be reached from the current node by following all paths that are consistent with any data volunteered by the user.

Step 2. If the same outcome class appears at all the leaf nodes that are reachable from the current node, then report this outcome class as the conclusion; otherwise, ask the user the question at the current node.

To illustrate the technique, consider what happens when it is known in advance that the age of the patient is young. As all three outcome classes are reachable from the root node of the contact lenses decision tree, the user must be asked about the tear production rate. If the user reports that the tear production rate is normal, then two different outcome classes, 'Soft' and 'Hard', can still be reached from the 'Astigmatism?' node. However, if the user now reports that astigmatism is present, then the only outcome class that is reachable from the 'Spectacle prescription?' node is 'Hard'. The system can therefore report hard contact lenses as the conclusion without asking about the spectacle prescription.

2.2 Effects of updating data

An important consequence of allowing the user to update previously reported data at any stage is that the path to the node reached by the intelligent system before the initiative was taken by the user may no longer be valid when the system regains the initiative. The classification process must therefore be restarted from the root node of the decision tree on each occasion that the system loses and then regains the initiative.

2.3 Effects of incomplete data

A well-known limitation of decision trees is that if the user is unable to answer any question in the decision tree, for example because it involves a test that carries high risk or cost, then no conclusion may be possible [3,10,18]. One solution to this problem if the system has access to data from which the decision tree was constructed is to use a dynamic (or lazy) approach to induction in which the next best question can be selected if the user is unable (or declines) to answer the most useful question [4,11]. However, this is not possible in an intelligent system that does not have access to the data, if any, from which the decision tree was constructed.

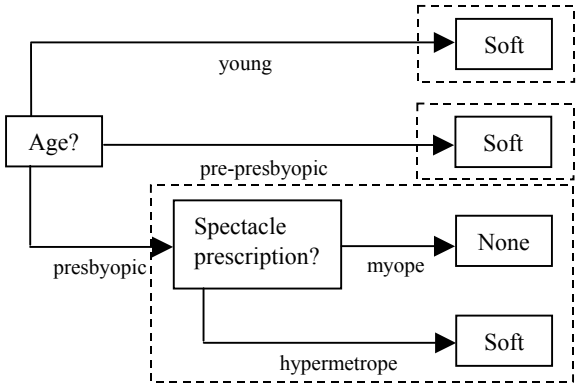


Figure 5. If all three sub-trees give the same result, then the conclusion is soft contact lenses even if the age of the patient is unknown.

To address this issue, we have developed a new approach to tolerating incomplete data which resembles a committee-based approach to classification. In effect, the overall decision is based on the votes of a committee of classifiers applied to the sub-trees corresponding to all possible values of the unknown attribute. If all the classifiers return the same outcome class from their sub-trees, then the attribute whose value is unknown cannot affect the conclusion. On the other hand, if two or more of the classifiers return different outcome classes, then there is insufficient data to reach a conclusion. If unknown values are encountered in the sub-trees, they are dealt with by recursive application of the committee-based approach.

To illustrate the approach, suppose the intelligent system has reached the upper 'Age?' node in the contact lenses decision tree, and the age of the patient is unknown. As shown in Figure 5, two of the sub-trees corresponding to the possible values of age are degenerate decision trees from which the corresponding classifiers immediately return the outcome class 'Soft'. The outcome class returned by the classifier applied to the third sub-tree depends on the user's answer to the question 'Spectacle prescription?'. The user is therefore asked this question. If she reports that the spectacle prescription is hypermetrope, then the outcome class returned by all three classifiers is 'Soft', so the conclusion is soft contact lenses regardless of the age of the patient. On the other hand, if the answer is myope, then a conclusion cannot be reached - unless of course the user can provide the missing data.

3 THE CONFIRM PROTOTYPE

We now present an implementation of the techniques presented in Section 2 in CONFIRM, a prototype intelligent system for classification based on decision trees. The example consultations that we use to illustrate the system's ability to support mixed-initiative interaction are based on the decision tree in Figure 1. The CONFIRM dialogue is in fact menu driven as well as mixed initiative. Because of the limitations of space, however, only the menu option (or combination of menu options) selected by the user at each stage is shown in the example consultations.

3.1 Sharing of initiative

Initially, the system has the initiative in CONFIRM. The user can remain in a passive role as long as she wishes by simply answering the questions she is asked by the system. However, the user can take the initiative at any stage to VOLUNTEER data or UPDATE previously reported data. As described in Section 4, the user can also SUGGEST a "target" outcome class to be pursued.

Confirm:	What is the tear production rate?
User:	VOLUNTEER age
Confirm:	What is the age of the patient?
User:	young
Confirm:	What is the tear production rate?
User:	normal
Confirm:	Is astigmatism present or absent?
User:	present
Confirm:	The conclusion is hard contact lenses

Figure 6. Example consultation showing CONFIRM's ability to take account of information volunteered by the user.

3.2 Volunteering data

The example consultation in Figure 6 shows CONFIRM's ability to take account of information that the user chooses to volunteer. Before answering CONFIRM's question about the tear production rate, the user takes the initiative to report that the age of the patient is young. On regaining the initiative, CONFIRM asks the user only about the tear production rate and astigmatism. At this point, the look-ahead technique described in Section 2.1 enables it to recognise that the spectacle prescription cannot affect the conclusion.

3.3 Updating data

Figure 7 shows CONFIRM's ability to take account of changes in the reported data. When the user changes her previous answer to the question about astigmatism, the only path that is consistent with the modified data leads CONFIRM to the upper 'Age?' node in the contact lenses decision tree. However, as the spectacle prescription is known to be hypermetrope, look-ahead search reveals that the only outcome class that is reachable from this node is 'Soft'. CONFIRM therefore realises that it no longer needs to know the age of the patient.

Confirm:	What is the tear production rate?
User:	normal
Confirm:	Is astigmatism present or absent?
User:	present
Confirm:	What is the spectacle prescription?
User:	hypermetrope
Confirm:	What is the age of the patient?
User:	UPDATE astigmatism
Confirm:	Is astigmatism present or absent?
User:	absent
Confirm:	The conclusion is soft contact lenses

Figure 7. Example consultation showing CONFIRM's ability to take account of changes in the reported data.

3.4 Tolerating incomplete data

Figure 8 shows an example consultation in which the user is unable (or declines) to answer the first question in the contact lenses decision tree. Using the committee-based approach to the problem of incomplete data described in Section 2.3, CONFIRM is able to reach a conclusion in the absence of the requested information. As the outcome class returned from both sub-trees of the node at which the unknown value occurs is 'None', the conclusion is no contact lenses regardless of the tear production rate.

Confirm:	What is the tear production rate?
User:	UNKNOWN
Confirm:	Is astigmatism present or absent?
User:	absent
Confirm:	What is the age of the patient?
User:	presbyopic
Confirm:	What is the spectacle prescription?
User:	myope
Confirm:	The conclusion is no contact lenses

Figure 8. Example consultation showing CONFIRM's ability to tolerate incomplete data.

Of course, even a single unknown value may mean that there is insufficient data to reach a conclusion. In this case, the user has the options of obtaining the results of omitted tests, examining their potential effects by performing a *what-if* analysis, or abandoning the consultation.

4 EXPLANATION IN CONFIRM

In previous work, we have argued that a goal-driven approach to attribute selection in decision-tree induction provides a natural basis for explaining the relevance of questions when the decision tree is used for problem solving [11,12]. In CBR Strategist, the relevance of an attribute can be explained in terms of the purpose for which it was selected, such as confirming a target outcome class or eliminating a competing outcome class [12]. Unlike CBR

Strategist, CONFIRM does not have access to the data, if any, from which the decision tree was constructed and cannot be expected to explain why an attribute was selected. It is also unable to identify a target outcome class on the basis of estimated class probabilities as in CBR Strategist. As we now show, however, CONFIRM can select a "target" outcome class in a way that requires access only to the decision tree. It can also explain the relevance of a given attribute in terms of its potential role in helping to confirm the target outcome class.

4.1 Selecting a target outcome class

By default, the selection of a target outcome class in CONFIRM is based on the *leaf counts* of the outcome classes that are reachable from the current node of the decision tree. That is, the outcome class selected at the current node is the one that occurs most frequently at the leaf nodes that are reachable from the current node. In contrast to an expert system, the target outcome class is continually updated as new information is obtained. Among the leaf nodes in the contact lenses decision tree, all of which are reachable from the root node, 'None' appears at 4 leaf nodes, 'Soft' at 3 leaf nodes, and 'Hard' at 2 leaf nodes. Initially in CONFIRM, the target outcome class selected by default is therefore no contact lenses. At the astigmatism node, the leaf counts for 'None' and 'Soft' are equal, so the target outcome class remains unchanged. However, if the tear production rate is normal and astigmatism is absent, the target outcome class changes to soft contact lenses.

Confirm:	The target outcome class is no contact lenses. What is the tear production rate?
User:	WHY
Confirm:	Because if tear production rate = reduced this will confirm no contact lenses. What is the tear production rate?
User:	normal
Confirm:	Is astigmatism present or absent?
User:	WHY
Confirm:	Because if astigmatism = present and spectacle prescription = hypermetrope and age = pre-presbyopic this will confirm no contact lenses. Is astigmatism present or absent?
User:	absent
Confirm:	The target outcome class is now soft contact lenses. What is the age of the patient?
User:	WHY
Confirm:	Because if age = young this will confirm soft contact lenses. What is the age of the patient?
User:	young
Confirm:	The conclusion is soft contact lenses

Figure 9. Example consultation showing CONFIRM's ability to explain the relevance of the questions it asks.

Alternatively, the user can select a target outcome class. A target outcome class selected by the user remains in force unless eliminated by the reported data, in which case CONFIRM informs the user that the target outcome class cannot be confirmed and selects a replacement by the default method. It should be noted that allowing the user to select a target outcome class does not affect the problem-solving process, as the order in which questions are presented is determined by the decision tree. Nevertheless, we believe that CONFIRM's ability to explain the relevance of an attribute in terms of its potential role in helping to confirm a target outcome class selected by the user may be beneficial in terms of acceptability.

4.2 Explaining the relevance of a question

Before answering any question in CONFIRM, the user can ask why it is relevant. In a rule-based expert system, explaining the relevance of a question is simply a matter of showing the rule that the system is currently attempting to fire [17]. In contrast, the explanation provided by CONFIRM is dynamically generated by look-ahead search from the current node to a leaf node at which the target outcome class is confirmed. The explanation is a rule-like statement of the additional evidence (the attribute values on the discovered path) required to confirm the target outcome class. At each node, CONFIRM gives priority to a branch that leads immediately to a leaf node at which the target outcome class is confirmed, or leads to a node at which the target outcome class remains unchanged. The example consultation in Figure 9 shows CONFIRM's ability to explain the relevance of the questions it asks in the context of the target outcome class. In contrast to an expert system, the user is shown only the *additional* evidence required to confirm the target outcome class. In this example, the user has left the choice of target outcome class to CONFIRM and is informed when it changes from no contact lenses to soft contact lenses.

4.3 Explaining how a conclusion was reached

Explaining how a conclusion was reached may seem an easier task than explaining the relevance of questions in a decision tree. The standard approach is to show the attribute values on the path from the root node to the leaf node at which the solution was found. However, this ignores the possibility that some of the evidence reported by the user may not affect the conclusion. Even worse, it is possible that not all of the evidence reported by the user supports the conclusion. For example, the following explanation for a conclusion of no contact lenses was generated by the standard approach from the contact lenses decision tree:

> **Since** tear production rate = normal **and** astigmatism = absent **and** age = presbyopic **and** spectacle prescription = myope the conclusion is no contact lenses

However, it can be seen from the contact lenses decision tree that a normal tear production rate cannot be regarded as evidence in favour of no contact lenses, since a reduced tear production rate is enough evidence on its own to reach the same conclusion. So the first condition of the example explanation is not only redundant but also potentially misleading.

The example consultation in Figure 10 shows how CONFIRM can eliminate a redundant condition from its explanation of how a conclusion was reached. Although the user reported that the age of the patient is presbyopic, this is not mentioned in CONFIRM's explanation of the conclusion. It can be seen from Figure 1 that if the age had been young or pre-presbyopic, the conclusion would

still have been soft contact lenses. The technique used by CONFIRM to eliminate redundant conditions like age in this example is similar to the technique for dealing with the problem of incomplete data described in Section 2.3. For each attribute value on the path from the root node to the leaf node at which the solution was found, CONFIRM explores the sub-trees corresponding to all the other values of the attribute. If all the sub-trees give the same conclusion, then the attribute can be eliminated from the explanation.

Confirm:	The target outcome class is no contact lenses. What is the tear production rate?
User:	SUGGEST hard contact lenses
Confirm:	The target outcome class is hard contact lenses. What is the tear production rate?
User:	normal
Confirm:	Is astigmatism present or absent?
User:	WHY
Confirm:	Because if astigmatism = present and spectacle prescription = myope this will confirm hard contact lenses. Is astigmatism present or absent?
User:	absent
Confirm:	Unable to confirm hard contact lenses. The target outcome class is now soft contact lenses. What is the age of the patient?
User:	presbyopic
Confirm:	What is the spectacle prescription?
User:	hypermetrope
Confirm:	The conclusion is soft contact lenses
User:	HOW
Confirm:	Since tear production rate = normal and astigmatism = absent and spectacle prescription = hypermetrope the conclusion is soft contact lenses

Figure 10. Example consultation showing CONFIRM's ability to explain how a conclusion was reached.

The elimination of redundant conditions in this way may help to make explanations of how a conclusion was reached more natural and easier for users to understand. Currently, CONFIRM does not attempt to eliminate more than a single condition from the explanations it produces, so its explanations are not guaranteed to be maximally general. Whether a more rigorous approach to the elimination of redundancy in explanations would be worthwhile in practice is one of the issues to be addressed by further research.

5 CONCLUSIONS

We have presented techniques for building mixed-initiative intelligent systems for classification and diagnosis based on decision trees. Issues addressed in the research presented include how to proceed in the absence of complete data, take account of information that the user chooses to volunteer, and provide meaningful explanations at the user's request. Our implementation of the techniques presented in CONFIRM provides proof of concept that mixed-initiative interaction can be effectively supported in decision-tree classification even when the intelligent system does not have access to the data, if any, from which the decision tree was constructed.

REFERENCES

[1] D.W. Aha, L.A. Breslow and H. Muñoz-Avila, 'Conversational case-based reasoning', *Applied Intelligence*, **14**, 9-32 (2001).

[2] J.E. Allen, 'Mixed-initiative interaction', *IEEE Intelligent Systems*, **6**, 14-16 (1999).

[3] E. Auriol, M. Manago, K.-D. Althoff, S. Wess and S. Dittrich, 'Integrating induction and case-based reasoning: methodological approach and first evaluations', *Proceedings of the 2nd European Workshop on Case-Based Reasoning*, 18-32 (1994).

[4] R. Bergmann and P. Cunningham, 'Acquiring customers' requirements in electronic commerce', *Artificial Intelligence Review*, **18,** 163-193 (2002).

[5] D. Bridge, 'Towards conversational recommender systems: a dialogue grammar approach', *Proceedings of the ECCBR-02 Workshop on Mixed-Initiative Case-Based Reasoning*, 9-22 (2002).

[6] J. Cendrowska, 'PRISM: an algorithm for inducing modular rules', *International Journal of Man-Machine Studies*, **27**, 349-370 (1987).

[7] M. Doyle and P. Cunningham, 'A dynamic approach to reducing dialog in on-line decision guides', *Proceedings of the 5th European Workshop on Case-Based Reasoning*, 49-60 (2000).

[8] M.H. Göker, 'Adapting to the level of experience of the user in mixed-initiative web self-service applications', *Proceedings of the ICCBR-03 Workshop on Mixed-Initiative Case-Based Reasoning*, 162-171 (2003).

[9] M.H. Göker and C.A. Thompson, 'Personalized conversational case-cased recommendation', *Proceedings of the 5th European Workshop on Case-Based Reasoning*, 99-111 (2000).

[10] M. Lenz, E. Auriol and M. Manago, 'Diagnosis and decision support', in M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard and S. Wess (eds.), *Case-Based Reasoning Technology*, Springer-Verlag, Berlin-Heidelberg, 51-90 (1998).

[11] D. McSherry, 'Strategic induction of decision trees', *Knowledge-Based Systems*, **12**, 269-275 (1999).

[12] D. McSherry, 'Interactive case-based reasoning in sequential diagnosis', *Applied Intelligence*, **14**, 65-76 (2001).

[13] D. McSherry, 'Minimizing dialog length in interactive case-based reasoning', *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, 993-998 (2001).

[14] D. McSherry, 'Mixed-initiative dialogue in case-based reasoning', *Proceedings of the ECCBR-02 Workshop on Mixed-Initiative Case-Based Reasoning*, 1-8 (2002).

[15] J.R. Quinlan, 'Induction of decision trees', *Machine Learning*, **1,** 81-106 (1986).

[16] H. Shimazu, A. Shibata and K. Nihei, 'ExpertGuide: a conversational case-based reasoning tool for developing mentors in knowledge spaces', *Applied Intelligence* **14**, 33-48 (2001).

[17] R.W. Southwick, 'Explaining reasoning: an overview of explanation in knowledge-based systems', *Knowledge Engineering Review*, **6**, 1-19 (1991).

[18] I. Watson, *Applying Case-Based Reasoning: Techniques for Enterprise Systems*, Morgan Kaufmann, San Francisco (1997).

Tuning Themes: Finding an Appropriate Level of Role Representation for Translation

Brian Murphy[1]

Abstract. Many schemes have been proposed for describing the semantics of the relationship between entities and situations. The number of thematic or semantic roles posited ranges from few (2 for Dowty [7]) to many (35 for Dixon [4]). This preliminary study examines the same text in Irish, English, Chinese, German and Spanish to see what approximate level of specificity is sufficient to find cross-language agreement when assigning roles to translated sentences.

1 Introduction

Role relationships are believed by some to determine what grammatical form arguments take in a language. But the degree of granularity required in these roles depends on the application. Highly generalised role schemes (e.g. Dowty's scheme [7], (1a)) are inadequate for describing less prototypical arguments, while verb-specific schemes (Robert van Valin's terminology [22]) are complex and tied to a set of mini-domains or frames (e.g. the speaking domain, (1b)).[2]

(1) a. Tara$_{agt}$ told him$_{pat}$ to go$_{pat}$

b. Tara$_{speaker}$ told him$_{addressee}$ to go$_{message}$

In this paper I look at thematic roles (an intermediate level of description) to see if they are adequate for the representation of predicate-argument relationships across languages, and so, whether they could be applied in automatic translation and other multi-lingual applications.

A subsequent question is what particular set of roles and interpretations a scheme should use. Are independent levels of description necessary for location and causality, or can they be conflated? And are dedicated roles required for psychologically salient domains such as communication or possession?

This is a preliminary study to identify the 'edges of the ballpark' that thematic roles should fall within. I have concentrated on participant roles, as many schemes do not include circumstantial roles.

In this paper, section 2 introduces the idea of thematic roles and why and how they are used, followed by a section reviewing various specific schemes that have been put forward. In sections 4 and 5 I describe the methodology used to compare three of those schemes, before making my conclusions.

[1] Trinity College Dublin, email: Brian.Murphy@cs.tcd.ie

[2] The roles and interpretations used in these and following examples are intended for illustration and do not follow any one scheme consistently. The abbreviations used are: *agt* agent, *act* actor, *pat* patient, *man* manner, *exp* experiencer, *src* source, *gol* goal, *thm* theme, *pro* property, *cau* cause, *loc* location, *tem* temporal.

2 What are thematic roles?

Thematic roles are semantic constructs posited by linguists to describe the nature of relationships between associated entities and situations. The sentence *The motorist blew his horn at the young lad for cycling like an maniac*, might be represented in this underspecified logical form:

(2) sound(Motorist, Horn, YoungLad)
cycle(YoungLad, LikeAManiac)

Here the order of predicates and arguments corresponds, by convention, to their order in English, but tells us nothing about their semantics. Thematic roles labels such as *agent*, *patient*, *goal*, *manner* and *cause* can fill this gap:

(3) sound(Motorist$_{agt}$, Horn$_{pat}$, YoungLad$_{gol}$)
cycle(YoungLad$_{agt}$, LikeAManiac$_{man}$)$_{cau}$

Thematic roles are generally used to describe three types of inter-term relations: those of causality, location and, less frequently, focus. Causality encompasses what is affected by the situation in question, what causes it, and whether that cause is intentional or inadvertent. Location typically covers where terms are located relative to the situation, and, if they are moving, where from and to. Focus describes what aspects of the situation are highlighted or emphasised (e.g. to differentiate between the roles of *corporation* and *units* in the sentences below).

(4) a. The corporation contains three business units

b. Three business units comprise the company

Thematic roles are not selectional restrictions[3], nor do they encode subject area or topic.

Participant thematic roles are usually considered the 'core' roles, describing the relationship between a situation and the entities that take part in it. As such they are part of our knowledge of verb and preposition meaning, and correspond to predicate/argument relationships. Many schemes have been proposed. Below is a sample of commonly used roles and their typical interpretation (adapted from [19]). Other roles suggested include Actor (a non-volitional Agent), Recipient, Percept (that perceived), Possessor and Property (a state attributed to an entity: e.g. I_{thm} *feel atrocious*$_{pro}$).

[3] Selectional restrictions are semantic requirements that verbs are thought to have on their arguments. For example the pottery sense of the verb *fire* requires something made of ceramics as its object, while the military sense requires a weapon.

Agent the (typically volitional) initiator, or doer of an action
Patient the affected party, or undergoer of the action
Theme the entity whose state, movement or location is described
Experiencer the entity aware of, but not affected by the situation
Beneficiary the entity for which the action is performed
Instrument the entity with which the action is performed
Goal the location towards which an entity moves
Source the location away from which an entity moves

Circumstantial thematic roles are less common, and generally describe relationships between situations (i.e. predicate to predicate) and so are part of our knowledge of the function of clause types and clausal operators. Some of the more common circumstantial roles are:

Cause the situation that precipitated this one
Condition the situation that would precipitate this one
Purpose the aim of the situation
Manner the way in which a situation takes place
Measure the extent (e.g. in cost, time or distance) associated with the situation
Location the location at which the situation takes place
Temporal the time at which the situation takes place

2.1 Why do we need them?

The grammatical roles found in sentences (subject, direct object, indirect object and adjuncts), though driven by semantic structure, are far from a direct mapping of it [7, 12]. While we may feel that the subject is usually the 'doer' and the object the 'undergoer', this is often not the case. For example, depending on the choice of active or passive form, or of the verb *buy* or *sell*, the simple proposition sold(Charlie, Albert, UsedConstituencyOffice) can be realised in English with many candidate utterances, of which these are just a few:

(5) a. Charlie$_{src}$ sold Albert$_{gol}$ a used constituency office$_{thm}$
b. Charlie$_{src}$ sold a used constituency office$_{thm}$ to Albert$_{gol}$
c. Albert$_{gol}$ was sold a used constituency office$_{thm}$ by Charlie$_{src}$
d. A used constituency office$_{thm}$ was sold to Albert$_{gol}$ by Charlie$_{src}$
e. Albert$_{gol}$ bought a used constituency office$_{thm}$ from Charlie$_{src}$

When we make comparisons across languages, there can be an even more dramatic divergence in structure. The English sentence *Dustin$_{sub}$ likes cream cake$_{dobj}$*[4] translates as (literal translations are added for illustration):

(6) a. Is maith$_{sub}$ le Dustin$_{adj}$ cáca uachtar$_{dobj}$ [Irish]
be good with Dustin cake cream
b. Dustin$_{sub}$ xǐhuan nǎiyóu dàngāo$_{dobj}$ [Chinese]
Dustin like cream cake
c. Sahnekuchen$_{sub}$ gefählt dem Dustin$_{iobj}$ [German]
cream-cake please Dustin
d. Le$_{dobj}$ gusta tarta$_{sub}$ de nata a Dustin$_{iobj}$ [Spanish]
him please cake of cream to Dustin

Since the relationship between sentence structure and underlying semantic form is so variable, many turn to thematic roles for semantic representation. In varying forms, they are central to grammar formalisms such as Chomsky's (termed θ-roles, e.g. see [17] and [18]) and Jackendoff's Lexical Conceptual Structure [12, 14], and are also used in Halliday's Functional Grammar [11] and Pustejovsky's Generative Lexicon [16]. Knowledge representation schemes make frequent use of them (e.g. [3], [21]) as do NLP applications including machine translation [5, p.110].

The general description of locational and causal relationships might also provide rich input to reasoning systems, for example when determining resultant positional information or revealing causal chains.

2.2 Requirements for Roles

There are three key requirements for thematic roles, by which their suitability can be measured (adapted from [7] and [19]). First a successful set of thematic roles must be comprehensive, in that it can be applied to all arguments and clauses in all usages and contexts. The only exceptions to this are semantically vacuous elements such as syntactic dummies (e.g. *It* in *It's raining, it's pouring*) and speech fillers (e.g. *eh um*).

The role set should also be differentiating. In a given utterance, every argument or clause should have an associated role that is unique to its function. This does allow arguments to share a role, but only when they share a common function, as *Fergus* and *Fergal* do in (18).

Lastly, all assignment of roles should be unambiguous, so that one and only one can be applied to a given argument or clause. It should not be the case that one argument qualifies for multiple roles, or seems to fall 'between' roles. A caveat to this involves schemes that suggest more than one level of thematic description ([11], [14]). However roles should remain unambiguous at a given level, as in example (13).

3 Models of Roles

While many authors agree on the utility of thematic roles, precisely what set to apply is still open. A primary question is what level of specificity, the roles should have [22].

Pollard and Sag's HPSG [20] uses verb specific roles such as *frightener* and *frightened*, but these add little semantic description, merely being a restatement of the verb's (in this case *subject-V-object*) argument structure. Other 'splitters' such as Fillmore (in his current research programme, Frame Semantics [9]), Dixon [4] and Halliday [11] have roles that are more general, but still specific to classes of verbs or situations. Dixon's 30 or so English verb types include classes such as SPEAKING, with the roles *speaker*, *addressee*, *medium*

[4] Abbreviations used for argument structure: *sub* subject, *dobj* direct object, *iobj* indirect object, *adj* adjunct.

and *message* and OWN, with the roles *owner* and *possession*. Other than their increased complexity, a drawback of these approaches is that they go beyond describing predicate/argument relationships, as some semantic aspects properly belonging to the verb are duplicated in the role labels.

'Lumpers' include Anderson [1], whose localist approach posits three roles (*source*, *location* and *goal*), and Dowty [7], who proposes just two prototype roles (*proto-agent* and *proto-patient*). These compact systems appear to lack the expressive power needed for many common utterances. Dowty's approach plots arguments in terms of obliqueness and cannot account for verbs of similar meaning but displaying inverted grammatical role structure, such as *include/comprise* and *fear/frighten*. And it is hard to see how minimal schemes can account for large numbers of arguments, e.g.:

(7) I have insured [my house] [against fire] [to a value of $130,000] [for $105 per month]

In this paper we will be examining the middle ground of semantic relations. Thematic roles aim for the economy that specific roles schemes lack, and the level of description that is not offered by generalised roles.

3.1 Roles as Cause and Effect

Charles Fillmore first introduced a causal set of thematic relations in 1968 as Case Grammar [8]: "... semantically relevant syntactic relationships involving nouns and the structures that contain them." He identified a preliminary list of six differentiating 'cases', primarily describing causation:

Agentive (typically animate) instigator
Instrumental inanimate force/object causally involved
Dative affected animate being
Factitive resulting entity
Locative location or spatial orientation
Objective affected entity (most neutral case)

Fillmore did not consider this scheme comprehensive, saying that "... additional cases will surely be necessary" – something that he appears not to have pursued.

Sowa ([21, pp.506-512]) has adapted the classical model of roles for use in knowledge representation, basing his scheme on the work of Harold Somers and Judith Dick. Again causally based, he posits four participant types, that for a given verb type correspond to more conventional roles:

Initiator initial cause
Resource material cause
Goal final cause
Essence formal cause

However this scheme fails to differentiate even in many relatively familiar situations. In the following transfer sentence both *she* and *Offaly* would presumably be tagged as *initiator*.

(8) She sent a postcard home from Offaly

Table 1. Somers-Dick-Sowa Roles

	Initiator	**Resource**	**Goal**	**Essence**
Action	Agent, Effector	Instrument	Result, Recipient	Patient, Theme
Process	Agent, Origin	Matter	Result, Recipient	Patient, Theme
Transfer	Agent, Origin	Instrument, Medium	Experiencer, Recipient	Theme
Spatial	Origin	Path	Destination	Location
Temporal	Start	Duration	Completion	PointInTime
Ambient	Origin	Instrument, Matter	Result	Theme

3.2 Roles as Location and Movement

Jackendoff [12], building on Gruber's [10] seminal work, suggested a set of primarily localist roles as an alternative to Case Grammar. Noticing patterns in the extension of verbs of location and movement to more abstract situations, he posited a single set of "thematic relations" for both.

(9) a. Ciara$_{thm}$ stayed [at work]$_{loc}$/[angry]$_{loc}$
b. Saoirse$_{thm}$ went [from Larne$_{src}$ to Bantry$_{gol}$]/[from hopeful$_{src}$ to exasperated$_{gol}$]

He later [13] extended this "Thematic Relations Hypothesis" to cover temporal and possessive expressions:

(10) The meeting$_{thm}$ will be at 6pm$_{loc}$
(11) The tricycle$_{thm}$ is mine$_{loc}$!

However, ultimately he admits that his mapping from location to causality is incomplete, and he is forced to introduce an *agent* to his scheme to explain the difference between intended and inadvertent movement of themes.

(12) a. Cian$_{thm}$ rolled down the hill$_{loc}$ (in his sleep)
b. Cian$_{agt\&thm}$ rolled down the hill$_{loc}$ (just for fun)

3.3 Hybrid Roles

There are obvious problems with both the purely causal or localist approaches. How, for example, does a localist scheme tag an *instrumental* role, and with verbs of perception (e.g. *hear*, *look*) is the *experiencer* the *goal* or the *source*? And how does a purely causal scheme distinguish between positional *source* and *goal*? Because of these difficulties Jackendoff proposes a two-tier scheme [14] with his localist roles on a 'thematic' tier, and causal roles in an 'action' tier:

(13) Pete$_{src\&agt}$ threw the ball$_{thm\&pat}$

He does not fully expand on this proposal, but Saeed suggests completing it as such [19]:

Thematic Tier theme, goal, source, location
Action Tier actor, agent, experiencer, patient, beneficiary, instrument

Dorr [5] took Jackendoff's [12, 14] and Levin's [15] work as a departure point when designing the semantic representation

lexicon for her interlingual[5] machine translation system UNITRAN, seeking to "...strike a balance between the causal and motion/location dimensions ..." [5, p.111]. Her one-tier thematic scheme mixes roles from both approaches (table 2). She has made available a database of verbs (11,000 entries), annotated with argument syntax and thematic structure [6].

Table 2. Dorr's LCS Roles

AG	agent
TH	theme
EXP	experiencer
INFO	information
SRC	source
GOAL	goal
PERC	perceived item
PRED	identificational predicate
LOC	locational predicate
POSS	possessional predicate
BEN	benefactive modifier
INSTR	instrument modifier
PROP	event or state
PURP	purpose modifier or reason
MANNER	manner
TIME	time modifier

The Universal Networking Language [3] is a machine-tractable interlingua for storing and sharing knowledge across computer systems. Thematic roles are at the core of its 'relations', describing the relationships between its 'universal words'. The table below (3) lists the UNL relations that correspond to participants. The circumstantial roles and conjunction relations (*and*, *or* and 'sequential and' *seq*) have been omitted. Besides its richness, an interesting feature of this scheme are the three comitative roles, to distinguish between the principal (i.e. *subject*) and accompanying entities among shared roles (see example (18) below).

Table 3. UNL Participant Roles

agt	agent: a thing which initiates an action
cag	co-agent: a thing not in focus which initiates an implicit event which is done in parallel
aoj	thing with attribute: a thing which is in a state or has an attribute
cao	co-thing with attribute: a thing not in focus is in a state in parallel
gol	goal/final state: the final state of object or the thing finally associated with object of an event
ins	instrument: the instrument to carry out an event
obj	affected thing: a thing in focus which is directly affected by an event or state
cob	affected co-thing: a thing which is directly affected by an implicit event done in parallel or an implicit state in parallel
ptn	partner: an indispensable non-focused initiator of an action
src	source/initial state: the initial state of an object or thing initially associated with the object of an event

[5] I.e. using a meaning representation that is neutral with respect to all languages.

4 Procedure

The aim of this study was to find a small representative text that is available in a wide range of world languages (including Irish), to which thematic role schemes could be applied. The UN Declaration of Human Rights [2] is a 1500 word text, available in translation to more than 300 languages. Despite its legalistic style, it does cover a very wide range of topic areas, including education, politics, religion, law, the family, asylum, ownership, employment, leisure, culture and health.

The five languages chosen are those that I am familiar with: English, Irish, Chinese, Spanish and German. These cover several branches of the Indo-European family of languages (Celtic, Romance and Germanic) together with a Sino-Tibetan language, and they have a varied argument structure. Though sharing English's *subject–verb–object* word order, German differs in allowing object fronting, and Spanish allows subject deletion. Irish has a *verb–subject–object* word order, while Chinese argument realisation is very flexible.

All 30 articles of the declaration were manually aligned, and then paraphrased in a simplified predicate form that adequately represented the meaning of each article and subarticle as commonly expressed in all five languages.

Then three of the schemes examined were applied to the predicate form: Uchida and Zhu's UNL, Dorr's LCS and the expanded version of Jackendoff's Thematic/Action tier model. Some major predicate types (e.g. verbs of movement, cognition, creation etc.) were then evaluated comparatively across the three representations in terms of coverage, differentiation and lack of ambiguity.

4.1 Predicate Form Used

As can be seen in the example below, some simplifying assumptions were made in producing the paraphrased predicate form. Firstly, I assumed that meaning is compositional at the clause level, so that external relations between predicates will not influence participant relations within those predicates. Hence tense, logical (e.g. *and*, *or*) and modal conjunctions (e.g. *if*, *until*) have not been annotated.

Complex sentences were deconstructed into component predicates, and nominalisations were given predicate translations (e.g. *offence* becomes *offend(x,y)*) where possible. Passive sentences were expressed actively with an undefined subject, and verbal and nominal modifiers (e.g. *innocent*, *according to law*) were expressed as one-place predicates:[6]

(14) Everyone charged with a penal offence has the right to be presumed innocent until proved guilty according to law in a public trial

(15) charge(U,P,penally(offend(P)))
lawfully(publicly(prove(U,P,guilty(P))))
entitle(P,presume(U,innocent(P)))

Two criteria were used to resolve conflicts when deciding on a common predicate structure across languages. Majority rule was one. For example the original form of the English *right to be presumed innocent* in the other four languages was:

[6] The logic and model theory of this representation language have not yet been developed. Entity abbreviations used are P a person, U undefined entity.

(16) a. is tuigthe é a bheith neamhchiontach [Irish]
be understood that be innocent

b. yǒu quán bèishìwéi wúzùi [Chinese]
have right regarded innocent

c. hat das Recht, als unschuldig zu gelten [German]
have the right as innocent to count

d. tiene derecho a que se presuma su inocencia [Spanish]
have right to that one presume their innocence

In this case, the predicate *presumed* won out, as it was used in both Spanish and English, and was semantically close to both *regarded* (Chinese) and *understand* (Irish).

Secondly, the most deconstructed predicate form available was used. Hence, in the example below, the German version which yielded force(U,P,leave(P,Country)) was preferred over force(U,P,exile(P)).

(17) a. No one shall be subjected to ... exile

b. Ní déanfar ... aon duine ... a chur ar deoraíocht [Irish]
not make ... single person ... that put in exile

c. rènhé rén bùdé jiāyǐ ... fàngzhú [Chinese]
any person must-not be-made ... exile

d. Niemand darf ... des Landes verwiesen werden [German]
no-one may ... the country expel be-made

e. Nadie podrá ser ... desterrado [Spanish]
no-one will-be-able to-be ... exiled

Perhaps a more problematic simplification was to ignore the quantification of nominals. As thematic roles are concerned with causality and location, I have assumed that the internal structure of a participant argument does not affect its role assignment – Verkuyl makes a similar assumption about the independence of verb aspectuality from thematic role [23, pp.256-259]. There are some cases where group/individual alternations result in different argument structures (so called comitatives), but this seems only to be when the individuals share a common role. Otherwise their combination in a group argument should not be possible.

(18) a. [Fergal and Fergus]$_{thm}$ met

b. Fergus$_{thm}$ met Fergal$_{thm}$

The examples found in the declaration (e.g. *be same*, *gather*, *marry*, *suit*) seem to bear this out. However the question of dependencies between quantification and role assignment merits further investigation.

5 Comparative Evaluation

The four classes of situations examined are chosen to cover aspects of the spatial, causal and psychological domains. For each class all the relevant predicates were identified, and then tagged following each of the three schemes under consideration, in order to identify problems and inconsistencies of coverage, differentiation and lack of ambiguity.

5.1 Verbs of Movement and Location

Unsurprisingly these verbs (*stay*, *locate*, *leave*, *enter* and *move*) are straightforward in the Jackendoff scheme. Subjects are *theme* (19a) and, when the movement is intentional, additionally *agent* (19b). The object is *goal*, *source* or *location*.

(19) a. Everyone has the right to recognition everywhere ...
entitled(P,recognise(U,P)) located(P,Everywhere)

b. Everyone has the right ... to return to his country
entitle(P,enter(P,possess(P,Country)))

In UNL movement verbs require an *agent* subject and the static verbs a *thingWithAttribute* subject, whether intentional or not. The location is assigned *place*, *placeTo* or *placeFrom*.

Dorr's scheme has the same problem with volition, and it seems that an *agent* subject forces a *theme* object. Both leave(*agent*, *theme*) and leave(*theme*, *source*) are possible, but leave(*agent, source*) apparently not (e.g. *He$_{agt}$ left home$_{src}$*).

5.2 Verbs of Directed Action

Verbs that do not necessarily bring about a change in the object (e.g. *protect*) have *theme&agent* subjects, and *goal* objects in the Jackendoff analysis. Affected objects (of telic verbs like *enslave*) have the additional role *patient*.

(20) a. Everyone has the right to ... protection ...
entitle(P,protect(U,P))

b. No one shall be held in slavery ...
must(not(enslave(U,P))

c. No one shall be subjected to torture ...
must(not(torture(U,P)))

Telic verbs are unproblematic for UNL, being *agent-V-object*, but non-telic verbs are assigned the same pattern.

Dorr's account of these verbs seems to vary somewhat, apparently depending on the ontological type of common arguments. All these verbs have an *agent* subject, but the object is variously *theme* (e.g. *enslave, remedy*), *goal* (e.g. *protect, defend*) and *experiencer* (e.g. *torture*).

5.3 Verbs of Possession and Transfer

Jackendoff sees possession as an analogy of location: possessions are *themes* and owners *locations*; donors *sources* and recipients *goals*. It is interesting to note that the syntax of possession in Irish corresponds exactly to this model:

(21) Tá [ag gach uile dhuine]$_{loc}$ [an ceart]$_{thm}$ ...
be [at every every person]$_{loc}$ [the right]$_{thm}$ ...
'Everyone has the right ...'

UNL has a dedicated *possessor* role, and the object carries *object*, though, as above, it is difficult to see in what respect it is 'affected'. Transfers (e.g. *deprive*, *grant*, *provide*) originate in the *from* role, arriving at the *to* role.

Dorr's *possessor* role makes ownership straightforward, but transfer is irregular, variously *agent-V-theme-goal* (e.g. *grant, pay*) and *agent-V-theme-possessor* (e.g. *deprive, provide*).

5.4 Verbs of Cognition and Communication

The verbs of thinking found in the texts (e.g. *believe*, *think*, *enjoy*, *recognise*) can be tagged with an *theme&experiencer* subject and *locative* object (one believes *in* something) for Jackendoff. However for intentional thought *agent* would have to replace *experiencer* – the tier system does not permit multiple roles on a single level. Judgement verbs (*presume*, *prove*, *determine*) may have the same structure, but the motivation for tagging the object as *location* is rather weak. Verbs of communication (*speak*, *teach*) have the pattern *source&agent* (subject) and *theme* (direct object) and *goal&experiencer* (indirect object).

(22) a. Everyone has the right ... to enjoy the arts ...
entitle(P,enjoy(P,Arts))

b. Everyone ... has the right to be presumed innocent until proved guilty ...
prove(U,guilty(P)) entitle(P,presume(U, innocent(P)))

c. ... to manifest his religion or belief in teaching ...
teach(P,U,possess(P,Belief))

UNL would have to select *agent* for subjects of all these verbs (thus not allowing for inadvertent thoughts). As regards the object, the documentation does not make it clear whether *object* or *thingWithAttribute* is appropriate.

Finally Dorr's model is well equipped for cognitive verbs. Thinking verbs use the *experiencer-V-percept* pattern, judgements verbs *agent-V-information* and communication verbs *agent-V-goal-information*.

6 Conclusions

The first thing to note is that there was less divergence in text structure across the languages than was anticipated. Perhaps this is because it is a legalistic text, and so the translation is particularly literal.

The hypothesis that thematic (rather than verb-specific) roles are adequate is supported by the data, as there were no instances found where they offered insufficient differentiation.

As regards which thematic roles to use, the comparison between the three systems was perhaps somewhat unfair for several reasons. Firstly, while none of the schemes is accompanied by clear criterial documentation for the assignment of roles, the categories used in the Jackendoff scheme are better established in the literature, and so easier to apply. Dorr's scheme was also handicapped by being in isolation from the rest of her LCS (the roles are not designed to be a wholly independent system) and because the use of the verb database removed any flexibility of interpretation for marginal uses (i.e. roles were being assigned on a type rather than token basis). It also has to be noted that the publicly available UNL documentation is particularly sparse. Finally the volume of data examined, the restricted style of the text, and the fact that no other subjects were consulted to verify my linguistic judgements, make any conclusions preliminary and rather tenuous.

Nevertheless, it is surprising that the Jackendoff scheme performed so well, despite having the smallest number of roles. For verbs of location, action and possession, Jackendoff's two-tier system gives the most convincing analyses in terms of differentiation and lack of ambiguity. Dorr's scheme is the most successful for cognitive verbs. Coverage does not present problems for either of these schemes.

This is compelling vindication of the decoupling of the action and thematic tiers. Dorr's scheme in particular seems to suffer as a result of the conflation of causality and location. Hence a two tier system, adopting some of Dorr's participant roles (such as *information* and *property*), together with UNL's comprehensive circumstantial roles suggests itself as a promising way forward.

While any multi-tier model could complicate computation, it could be precompiled to a single tier of all permutations, such as *theme&agent*, omitting semantically anomalous combinations such as *location&agent* and *source&experiencer*.

REFERENCES

[1] Stephen R Anderson, 'On the role of deep structure in semantic interpretation', *Foundations of Language*, **7**, 387–396, (1971).

[2] United Nations General Assembly. Universal declaration of human rights. http://www.unhchr.ch/udhr/navigate/alpha.htm, 1948.

[3] UNL Centre. The universal networking language specifications 3.0. http://www.unl.ias.unu.edu/unlsys/unl/UNL Specifications.htm, 2001.

[4] R M W Dixon, *A New Approach to English Grammar, on Semantic Principles*, Clarendon Press, Oxford, 1991.

[5] Bonnie Jean Dorr, *Machine Translation: A View from the Lexicon*, MIT Press, Cambridge, 1993.

[6] Bonnie Jean Dorr. LCS database documentation. http://www.umiacs.umd.edu/∼ bonnie/LCS_Database_Documentation.html, 2001.

[7] David Dowty, 'Thematic proto-roles and argument selection', *Language*, **67**(3), (1991).

[8] Charles J Fillmore, *The Case for Case*, 1–92, Universals in Linguistic Theory, Holt, Rinehart and Winston, New York, 1968.

[9] Charles J Fillmore, *Frame Semantics*, 111–138, Linguistics in the Morning Calm, Hanshin, Seoul, 1982.

[10] J S Gruber, *Studies in Lexical Relations*, Indiana University Linguistics Club, Bloomington, 1965. Reprint of PhD Thesis.

[11] M A K Halliday, *An Introduction to Functional Grammar*, Edward Arnold, London, 1985.

[12] Ray Jackendoff, *Semantic Interpretation in Generative Grammar*, MIT Press, Cambridge, 1972.

[13] Ray Jackendoff, *Semantics and Cognition*, MIT Press, Cambridge, 1983.

[14] Ray Jackendoff, *Semantic Structures*, MIT Press, Cambridge, 1990.

[15] Beth Levin, *English Verb Classes and Alternations*, University of Chicago Press, Chicago, 1993.

[16] James Pustejovsky, *The Generative Lexicon*, MIT Press, Cambridge, 1995.

[17] Andrew Radford, *Syntax: A Minimalist Introduction*, Cambridge University Press, Cambridge, 1997.

[18] Malka Rappaport and Beth Levin, *What to do with θ-Roles?*, 7–36, Thematic Relations, Academic Press, New York, 1988.

[19] John I Saeed, *Semantics*, Blackwell, Oxford, 1997.

[20] Ivan A Sag and Thomas Wasow, *Syntactic Theory: A Formal Introduction*, CSLI, Stanford, 1999.

[21] John F Sowa, *Knowledge Representation: Logical, Philosophical, and Computational Foundations*, Brooks/Cole, London, 2000.

[22] Robert D van Valin, *Generalized Semantic Roles and the Syntax-Semantics Interface*, 373–389, Empirical Issues in Formal Syntax and Semantics, Thesus, The Hague, 1999.

[23] Henk J Verkuyl, *A Theory of Aspectuality*, Cambridge Studies in Linguistics, Cambridge University Press, 1993.

A Comparison of Collaborative Recommendation Algorithms Over Diverse Data[1]

John O'Donovan, John Dunnion[2]

Abstract. Information filtering techniques are becoming more widely used as available information spaces grow exponentially larger. New techniques for filtering information are being developed to tackle the information overload problem. This paper presents an assessment of the performance of popular recommendation stratagem over a range of diverse data. Our aim is to show that the relative performance of these algorithms varies as they are applied to different data, and to classify datasets according to a range of heuristics, and determine a suitability for recommendation techniques to a dataset classification. We run performance tests on Item-Based Collaborative Filtering and a Pure Collaborative Filtering algorithm. These tests consider neighbourhood size, and density of the training set. Our empirical results show provisionally that there is a significant difference in the relative performance of these algorithms over our four experimental data platforms, and that this can be used to determine suitability towards different data types. We classify our datasets according to a range of metrics, including sparsity, user-item ratio and set size. This classification will enable us to employ our results as guidelines for algorithm performance on new datasets, based solely on the classification metrics.

1 Introduction

Recommender Systems are designed to help tailor a users information space by suggesting items the system believes will be of interest to the user. An increasing number of online stores provide recommender systems on their sites, e.g. E-Bay, Amazon.com etc. There are two main bases upon which these systems operate: Collaborative[8][10] and Content-Based Filters[6]. A Content-Based approach to the recommendation task compares similarities in descriptions of content of items to user profiles to arrive at its goal. In the Collaborative approach, which is the focus of this paper, users provide ratings for items in a particular domain, and the system exploits similarities and differences between users based on these to compute its recommendations: If users A and B rate k items similarly, they share similar tastes and should rate other items similarly. There are many implementations and variants of collaborative filtering techniques in todays recommender systems, all of which are restricted by the same fundamental drawbacks. Collaborative Filtering (CF) techniques do not require items to be machine-analysable, and can arrive at serendipitous recommendations, that is, they can recommend relevant items that are completely different from those in a users profile. They also require little knowledge-engineering overhead.[7] CF techniques are all subject to two serious restrictions. *Sparsity Problem*: In any given case, it is unlikely that two users have co-rated many of the items in the system. Accurate similarity measurements depend on rich user profiles with high overlap which can be costly to attain. *Latency Problem*: This affects new, uncommon, or items liked by few users. These items will not be recommended by a system until they are included (if ever) in a sufficient number of user profiles[7].

We investigate these performance difficulties in four different experimental datasets, using tests for density of the data, and neighbourhood size. Using these metrics, we aim to show that there is a difference in the relative performance of our recommendation algorithms based on the type of data they are used on, and hence realise suitability of particular technique(s) to individual data types. For example, we would like to show that as the number of neighbours looked at for similarity computation increases past some threshold on a dataset of jokes, and one of movies, predictive accuracy of our algorithms will fall faster on the jokes set, as there are smaller clusters of people with similar tastes in jokes, than movies, (Peoples tastes in jokes tends to be more erratic than in movies).

If for example, we can show that predictive accuracy for the Item-Based (IBCF) algorithm peaks at a higher point than pure CF on jokes data, and a lower point on movie data, we can hence assert that there may be a suitability towards that data type for the IBCF algorithm. Of course the density of the training data will have a serious effect on the accuracy of the predictions, so we introduce density tests in our evaluation of neighbourhood sizes, and normalise across all datasets by multiplying all parameters by the density of each dataset. Our overarching goal in this paper is to uncover these performance differences and essentially uncover links between performance of recommendation techniques and data types.

2 Pure Collaborative Filtering

This algorithm is the popular user-based CF, first introduced in 1989 by John Hey of LikeMinds, and furthered 1998[8] by the GroupLens[10] research group. Our User-Based CF is performed by calculating the Pearson correlation coefficient[2] of

[1] This work is part of the INTINN project, from which the support of Enterprise Ireland is gratefully acknowledged.

[2] Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland, email: john.odonovan@ucd.ie, john.dunnion@ucd.ie

two vector representations of users, using this to define peer-groups, and suggesting items a users peers liked that has not yet been seen by the user. There are a range of methods for calculating similarity coefficients, such as Cosine-Similarity, Jaccard's coefficient etc. We use Pearson's as it is the most widely used in CF and allows us a more direct comparison with other existing systems. When the peergroup of users is chosen based on the similarity to an active user (the user receiving recommendations), a weighted combination their ratings is used to generate predictions, as seen in [6]. The following is a step-by-step outline of the CF algorithm.

1. Calculate the similarity between the active user and every other user.
2. Form the peer group by selection of the top-n most similar users.
3. Generate a prediction from a weighted combination of the peer groups ratings.
4. Return a ranked list of these items as the recommendation set.

The following formula calculates the Pearson Correlation in the first step:

$$corr_{x,y} = \frac{\sum u \in U(R_{u,x} - \overline{R}_x)(R_{u,y} - \overline{R}_y)}{\sqrt{\sum u \in U(R_{u,x} - \overline{R}_x)^2 \cdot \sum u \in U(R_{u,y} - \overline{R}_y)^2}} \quad (1)$$

Where $corr_{x,y}$ is the Pearson correlation coefficient between user x and user y. $R_{k,x}$ is the rating of user x on item k, and R_x is the average item rating by user x.

Prediction calculation (step 3 above) is done using 2, which calculates the weighted average of deviations from the neighbours mean:

$$p_{x,i} = \overline{r}_x + \frac{\sum u \in U(r_{u,i} - \overline{r}_u) \times corr_{x,u}}{\sum u \in U(corr_{x,u})} \quad (2)$$

Where $p_{x,i}$ is the prediction for user x on item i, and $corr_{x,u}$ is the similarity between users x, and u.

3 Item-Based Collaborative Filtering

One of the first detailed papers on IBCF was published in 1997 by D. Fisk [4], based on the MORSE movie recommender system. More recent work [9] was carried out on this algorithm by Konstan and Riedl in 2001. The main difference between Item-Based CF (IBCF) and the User-Based algorithm is that IBCF makes its predictions based on a model of Item similarity rather than user similarity. This algorithm looks at a set of items the active user has rated and computes their similarity to a target item i, in order to evaluate it for recommendation. The IBCF algorithm selects the k-most similar items $\{i_1, i_2, ..., i_k\}$ and their corresponding similarity values $\{s_1, s_2, ..., s_k\}$. The prediction is found by taking the active users ratings on these similar items, and weighting these by multiplying them by their similarity values. The algorithm boils down into two main phases: Phase 1 is the Model-Building and Phase 2 is the Prediction Calculation.

3.1 Model Building

The Model Building phase involves computation of similarity between items. This particular implementation uses a correlation-based similarity metric, calculated using Pearson correlation.

To make this algorithm as efficient as possible, we use the cases where items i and j have been co-rated by users. In figure 1, below, using Pearson's equation from section 3, the set of co-raters (U) consists of only those users who have rated both item i and item j. In this case, users 0 and 3 This calculation results in an $n \times n$ matrix of item similarities, where n is the number of items in the dataset. Since we are generally only interested in the top-n most similar items to each other item, we can sort each item profile in non-increasing order, and store only the top-n from this list as our model. One advantage of this method is that the entire model-building phase is an off-line computation, resulting in rapid recommendations for online users. Another is that this technique is scalable to any number of users, as their recommendations are generated from a static item-item model.

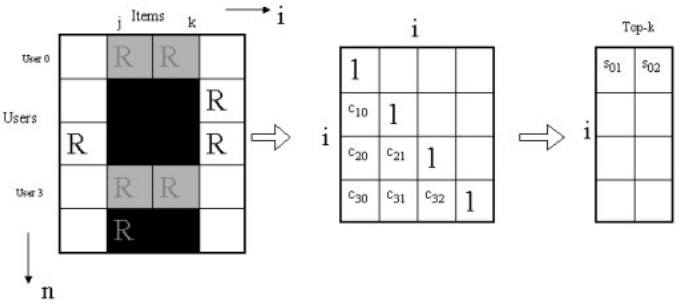


Figure 1. Offline Computation of the Item-Based Model

3.2 Prediction Calculation

The prediction phase of the IBCF algorithm is done as follows:

1. User rates a set U of items.
2. A Candidate set C of items is formed by taking the union of the k most similar items for each item $j \in U$. (Simply by taking the top-n from the ordered lists computed in the model-building phase)
3. Any duplicate items are removed. ie any $j \in (C \cap U)$.
4. Find the similarity between each item $c \in C$ and the set U. This is done by consulting the Item-Item correlation matrix from phase 1, and summing the correlation values between each $c \in C$ and every item in U.
5. C is then sorted in non-increasing order w.r.t these similarities.
6. The top-n items in C are the Recommendation Set.

4 Experimental Evaluation

4.1 Experimental Data

For training and testing our algorithms, four experimental datasets are used: Jester[5] (an experimental dataset of jokes ratings, consisting of 21,800 users ratings of 100 jokes), EachMovie (73,000 users ratings of 1628 movies), PTV[3] (622 user ratings on TV programmes), and MovieLens[8] (100,000 user ratings in the movie domain.) It is hoped to also include a customer-product purchase database from an online sales company in the near future. For the purposes of our testing we selected subsets of 900 profiles from each of the above datasets comprised of the largest profiles: those users who had rated 20 items or more (with the exception of PTV which only contains 622 profiles). For modularity in our system, these datasets were all parsed into the same format and stored in an SQL database.

4.1.1 Dataset Classification

We classify our datasets according to the metrics outlined below for the purpose of proposing suitability of particular techniques to other datasets which fall into the same classification. The metrics employed are dataset size, dataset density, and the ratio of users to items. Dataset size and data type are explained in the previous section, leaving sparsity and user-item ratio. Firstly, we define the sparsity of our data as seen in [11]:

$$1 - \frac{number\,of\,nonzero\,entries}{number\,of\,total\,entries}$$

Table 1 details the sparsity and user-item ratios for our four datasets.

Dataset	Users-Items	Sparsity	Type
PTV	1:6	94.25%	TV Progs
MovieLens	9:13	63.86%	Movie Ratings
Jester	9:1	54%	Jokes Ratings
EachMovie	9:17	33.97%	Movie Ratings

Table 1. Classification of Experimental Data

The User-Item ratio is a valuable classification metric since our similarity algorithms for this paper are based on these two features. Is has been shown in [9] that IBCF performs better with a greater number of users in an item-profile from the training set, whereas the total number of items is not as relevant. This is quite intuitive since there is obviously more information on which to base our similarity computations. The same holds true for the number of items in a users profile in the UBCF algorithm. Another area of interest is the distribution on ratings within a dataset. For future work on dataset classification, Chauvnet's criterion [4] will be employed to attempt to remove possible misleading ratings.

4.2 Experimental Procedure

Common metrics for evaluating the accuracy of a prediction algorithm are in two main areas: Statistical Accuracy and Decision-Support metrics. Due to the number of algorithms and datasets, we focus on Statistical Accuracy only for our experimental evaluation. Our statistical accuracy is obtained by comparing a set of predicted values with a set of user-provided values (Test Data). MAE (Mean Absolute Error), is the average absolute difference between predicted ratings and actual ratings. We have simplified this somewhat further and measured predictive accuracy of our algorithms. For each dataset, if a users rating is beyond a certain threshold, the item is "liked" by that user. This level of granularity was chosen because people will use the rating scales differently. We tailored this threshold value for each individual scale, based on distribution of ratings. We predict the "liked" items for the unseen test data and record accuracy on each dataset. User profiles are split into training and test data. The training data is fed to each filtering component individually and each generates its own predictions for the unseen test data.

4.2.1 Density testing

In order to assess an algorithms performance with respect to density of a dataset, we vary the density of the set by running our accuracy tests over training ratios from 10% to 90% in increments of 10%. We should see increases in the accuracy of our recommendations as we increase the training set. There is a peak around 90% for the training set. This may be due to the fact that the test set becomes too small to be sufficient test sample after this threshold. For the rest of our testing we select this test-train ratio, as it should provide the most accurate results.

4.2.2 K-nn testing

It has been recorded[9] that there is a peak in the curve for the user-based and item-based algorithms at a neighbourhood size of around 30-40. We vary this neighbourhood-size between 10 and 100 in intervals of 10, in order to ascertain an optimal value for number of neighbours. We find that there is a difference in the relative performance of our algorithms over the datasets, most notably with the Jester dataset, where quality of recommendations drops dramatically after a threshold of approximately 40 with User-Based filtering, and 64 with Item-Based filtering. This may be attributed to the fact that groups of users who share similar tastes in jokes are in fact much smaller than those in a movie / TV-program domain.

4.2.3 The Zero-R algorithm

For comparison purposes we also test a primitive algorithm known as Zero-r. This simply predicts the majority class in the training data. This algorithm itself is not much use as a functional predictor, but serves well as a benchmark for the rest of our prediction algorithms. In one experimental case on the Jester dataset, IBCF and CF both perform worse than this basic algorithm. It has been suggested in [1] that this may be due to serious *overfitting*: Modelling of the training data so closely that the algorithm loses sight of the real picture.

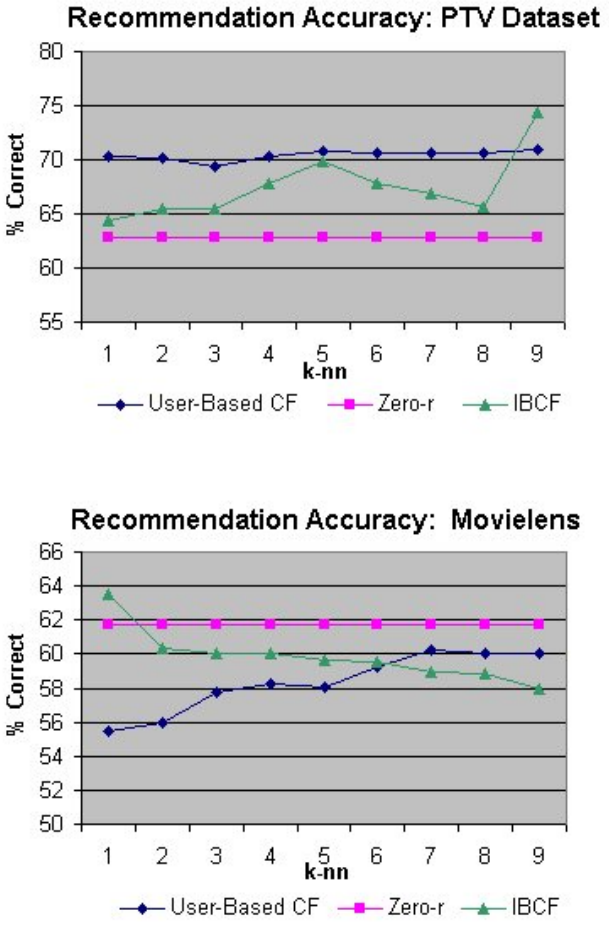


Figure 2. Testing neighbourhood sizes

4.3 Experimental Results

To recapitulate the aims outlined in section one, we aim to investigate relative performance differences of our recommendation algorithms over the different datasets we test them on. Our empirical results show that these performance differences do exist. In our density evaluations, we varied train-test ratios in order to assess the effect of density of training data on our datasets. It was found across all sets that a train-test ratio of around 80% was the optimal value for the density, so this value was made static at 80% for all of the k-nn testing.

4.3.1 PTV

Our results do indicate that there is a relative difference in the performance of our algorithms with different data. Starting with the PTV dataset: At neighbourhood size k of 10, UBCF performs better than IBCF by 5%. The performance of the UBCF algorithm does not vary much with k, reaching a peak of 70.6% around $k = 50$. The performance of the IBCF algorithm improves to almost equal to UBCF at $k = 50$. This suggests that the neighbourhood size has a greater effect on the IBCF algorithm for the PTV dataset. All of our prediction algorithms outperform the Zero-r predictor, predicts the majority class for the PTV dataset at 62.5%. The best performance seen on this dataset throughout our tests is by the UBCF algorithm at a neighbourhood size of 50. The result in this dataset is somewhat different to [9], where IBCF outperforms UBCF at all k values. With respect to sparsity of the dataset, PTV is 94% sparse. We can see from 3 that item based has a better relative performance in a more sparse environment. Looking again at 3 for comparison, on the most densely populated set, movielens (66%), the performance of IBCF is notably worse than its user-based counterpart.

4.3.2 Movielens

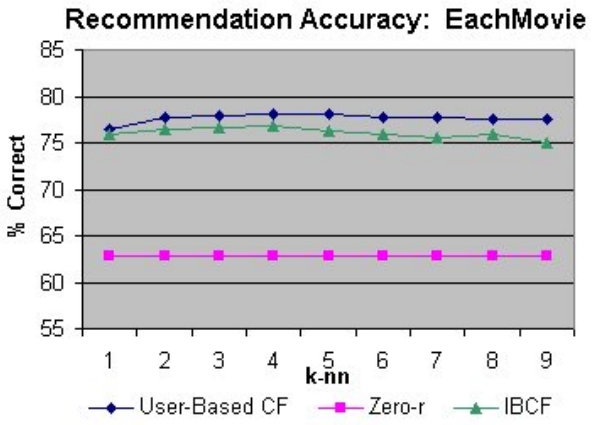
On the Movielens dataset, the results are very different. In this case, the UBCF algorithm is more affected by changes in neighbourhood size. At low k values, UBCF performs up to 9% worse than IBCF. However, unlike in the PTV set, this accuracy improves with k, and by $k = 60$ the UBCF algorithm outperforms IBCF. Interestingly, our baseline algorithm, Zero-r, performs better than the other prediction algorithms in this case. This is an odd result, which normally could be explained by examining the size of the majority class. However, the Zero-r algorithm has produced between 62% and 64% accuracy across all four of our test sets, which is the size of the majority class, meaning the Movielens set is not too dissimilar from the others in terms of this metric.

4.3.3 EachMovie

In the EachMovie dataset, there are further relative differences in performance between our algorithms with respect to data. Unlike all of the other test sets, the accuracy of IBCF and UBCF follow roughly the same trends, and are only marginally different at approximately 1%. The optimal recommender configuration for this set was found to be at a density of 30, and neighbourhood size of 40, using the UBCF algorithm. Our benchmark algorithm was outperformed across all k values by around 10%. These are the expected ranges for the other algorithms. EachMovie is the most dense of the datasets, the overall adverse effect of the collaborative filtering sparsity problem can be seen in 2 and 5, where our trained algorithms perform between 6 and 20% better than our baseline algorithm. This is not the case with the other datasets which are 20, 30 and 60% more sparse than EachMovie. We can note that our algorithms are capitalising on the more dense training set to generate better recommendations.

4.3.4 Jester

Jester shows more interesting results for our algorithms. The best performance here is given by the IBCF algorithm at low k values. This may be due to the fact that the clusters of people with similar tastes in jokes are smaller than those with similar tastes in movies, or TV programs. IBCF performs best up to a neighbourhood size of 50 before falling sharply with increasing k. At $k = 10$, UBCF performs as well , but falls sharply with increasing k. The Zero-r algorithm at 62.75% performs worse than the others, until neighbourhood reaches



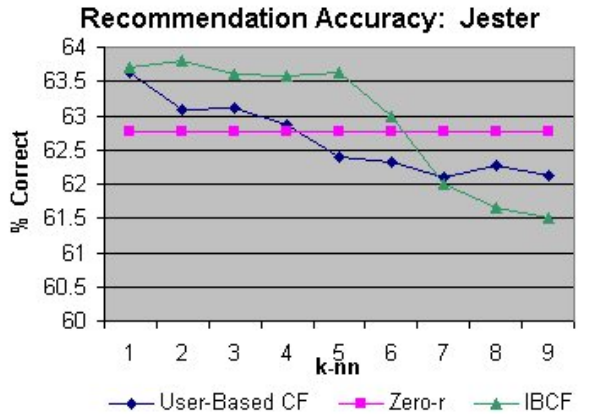
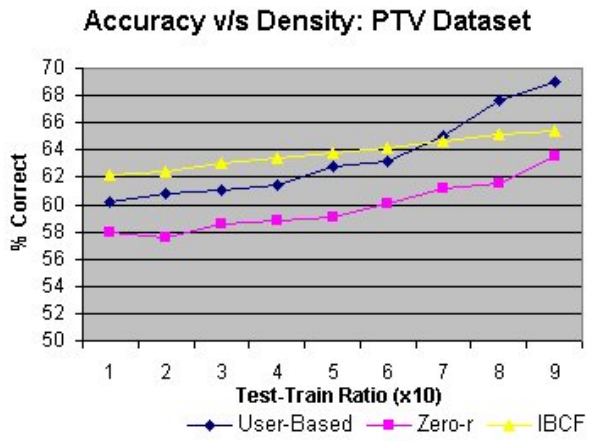


Figure 3. Testing neighbourhood sizes



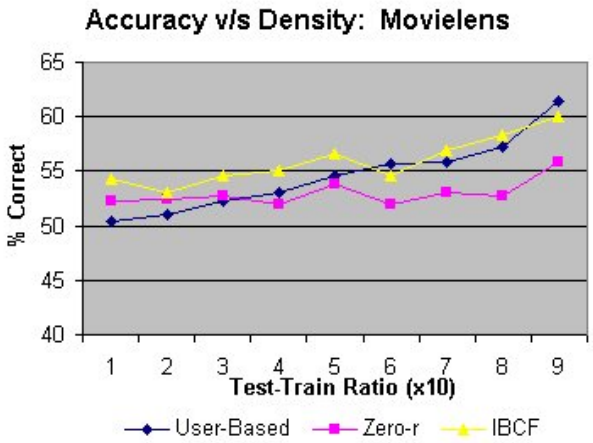


Figure 4. Density Testing

a max threshold (40 for UBCF and 64 for IBCF). We can see the impact of the user-item ratio on the jester results in 2 where the IBCF algorithm performs better than the UBCF algorithm. Because the User-Item ratio for this dataset is 9:1 there are more users in the item-profiles on which we compute our similarities. This result reinforces the proposal in [9] that IBCF works best with more users and less items.

4.4 Recommendation Accuracy with Varying Density Levels

Density of the training data can be experimentally controlled by varying the test-train ratio of the experimental data. In 4 and 5 we can see the results of this variation. It is clear that across all of our datasets there an increase in predictive accuracy as the percentage of training data increases. This indicates that the algorithms are in fact learning well. We can see that the Item-Based algorithm seems to perform relatively better at lower test-train ratios, most notably in the Jester dataset, and is less affected overall by the change in the training set size. This tells us something about the quality of the information captured by the Item-Based algorithm. The good performance of this algorithm at low test-train ratios may indicate that this filtering strategy may be employed as a method to solve the sparsity problem with User-Based collaborative filtering. Also, since better predictions can be made with less training data, this technique will save on memory (small model sizes) and will generate faster recommendations.

4.5 Summary of Results

The empirical results discussed here do show that performance varies across data types. It is hoped that we can capture this knowledge dynamically in a Machine Learning environment in order to adapt recommendation strategies to suit particular data types. These results have also shown that the Item-Based algorithm has many advantages over the User-Based algorithm. This information will be valuable in the development of any hybrid stratagems.

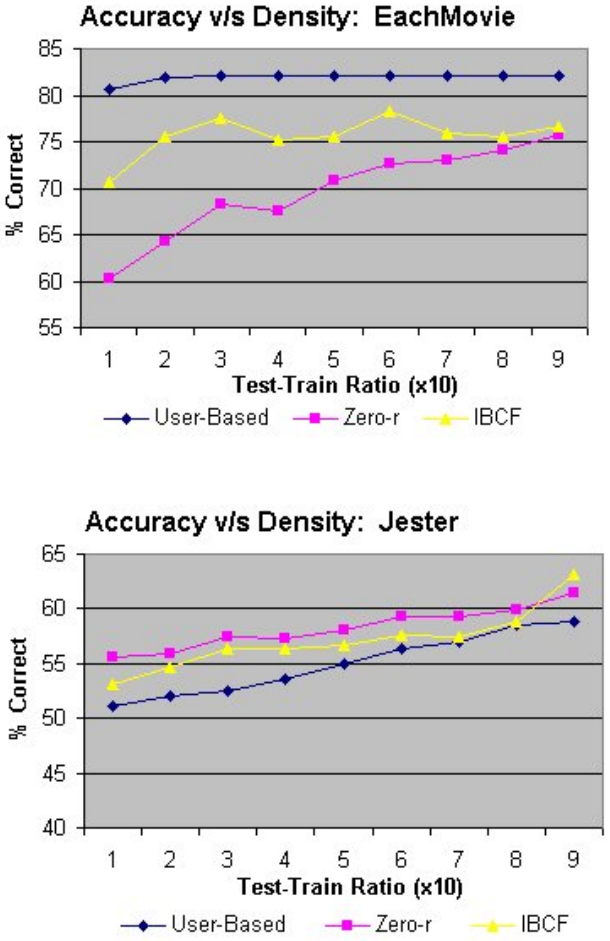


Figure 5. Density Testing

5 Conclusion and Future Work

Collaborative recommendation techniques are powerful tools which enable the delivery of relevant information without users being forced to trawl through irrelevant material to find what they require. These techniques are relied on more and more as technology takes a stronger foothold in everyday living. These techniques also have great value to business as they can present a user/customer with items/products they are interested in, thereby generating more sales for the business with less hassle for the customer. Recommendation techniques are not without their flaws however, especially collaborative techniques. Problems such as sparsity of data, latency and early-rater discussed earlier lead to varying performance of recommendation techniques. In this paper we have evaluated three recommendation strategies on varying data. Our results indicate that there is a relative performance difference of these techniques over the datasets they were tested on.

Our future work will include extending the scope of test data to other domains, starting with a product-purchase database. It is also hoped to include new recommendation techniques in this analysis as they arrive. A machine-learning application is currently being developed to automatically learn the best recommendation technique (or hybrid technique) for a given dataset. This aspect of the AdRec system will be dealt with in a later paper.

REFERENCES

[1] Syed S. Ali and Susan McRoy. Links: Java resource for artificial intelligence. *intelligence*, 11(2):15–16, 2000.

[2] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Gregory F. Cooper and Serafín Moral, editors, *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 43–52, San Francisco, July 24–26 1998. Morgan Kaufmann.

[3] Paul Cotter and Barry Smyth. PTV: Intelligent personalised TV guides. In *Proceedings of the 7th Conference on Artificial Intelligence (AAAI-00) and of the 12th Conference on Innovative Applications of Artificial Intelligence (IAAI-00)*, pages 957–964, Menlo Park, CA, July 30– 3 2000. AAAI Press.

[4] D. Fisk. An application of social filtering to movie recommendation. *Lecture Notes in Computer Science*, 1198:116–131, 1997.

[5] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[6] P. Melville, R. Mooney, and R. Nagarajan. Content-boosted collaborative filtering, 2001.

[7] Derry O'Sullivan, David C. Wilson, and Barry Smyth. Improving case-based recommendation: A collaborative filtering approach. In *Proceedings of the Sixth European Conference on Case Based Reasoning.*, pages LNAI 2416, p. 278 ff., 2002.

[8] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work*, Sharing Information and Creating Meaning, pages 175–186, 1994.

[9] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Reidl. Item-based collaborative filtering recommendation algorithms. In *World Wide Web*, pages 285–295, 2001.

[10] Badrul M. Sarwar, Joseph A. Konstan, Jonathan L. Herlocker, Bradley N. Miller, and John Riedl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Computer Supported Cooperative Work*, pages 345–354, 1998.

[11] B. Smyth, D. Wilson, and D. O'Sullivan. Improving the quality of the personalised electronic programme guide. In *In Proceedings of the TV'02 the 2nd Workshop on Personalisation in Future TV, May 2002.*, pages 42–55, 2002.

An Evaluation of the Performance of Collaborative Filtering

Michael P. O'Mahony[1] and **Neil J. Hurley**[1] and **Guénolé C.M. Silvestre**[1]

Abstract. Personalisation features are key to the success of many web applications and collaborative recommender systems have been widely implemented. These systems assist users in finding relevant information or products from the vast quantities that are frequently available. In previous work, we have demonstrated that such systems are vulnerable to attack and that recommendations can be manipulated. We introduced the concept of *robustness* as a performance measure, which is defined as the ability of a system to provide consistent predictions in the presence of noise in the data. In this paper, we expand on our previous work by examining the effects of several neighbourhood formation schemes and similarity measures on system performance.

1 Introduction

Given the success of recommender systems in web applications, much research has been conducted into understanding and improving the technology involved. One of the most popular and successful techniques that has been used to implement these systems is known as *collaborative filtering* [1, 7, 8]. This approach helps users to make choices based on the opinions of other users in a system. The basic heuristic employed is that users who agreed or disagreed on items in the past are likely to agree or disagree on future items. To make recommendations for a particular user, collaborative filtering algorithms find the most similarly-minded users (neighbours) in a system, and weight and combine the preferences of those users.

The performance of recommender systems is, of course, a key consideration. Performance has been extensively evaluated from the point-of-view of *efficiency*, *coverage* and *accuracy* [1, 2, 9]. Recently in [5], we have proposed another important performance measure – namely the *robustness* of the algorithm. A robustness analysis considers how system recommendations vary with changes in the system dataset. In general, some inaccuracies in system data should be expected since, for example, it is natural that some degree of noise will occur when users are required to explicitly express a preference on items. We have demonstrated in [5] that memory-based collaborative filtering is reasonably robust to such unbiased noise, with little change in recommendations observed when random noise was inserted into the system database.

However, it is also conceivable that malicious users may attempt to attack recommender systems in order to promote or demote particular products. (We refer to these attacks as *product push* or *product nuke* attacks respectively). Attacks can be implemented by inserting bogus data through the normal user interface. In some cases, attacks may be costly to perform if, for example, the input of data required the actual purchase of an item. However, if a system permits data entry without a purchase, then the effort required to mount an attack is not significant. Hence, it is important to also understand how biased noise, entered with a specific motive in mind, can affect the performance of the system. Our recent work has shown that collaborative filtering is vulnerable to a biased noise attack and that, with the addition of relatively small amounts of data, significant changes in system output can be achieved. While research has been conducted into privacy issues arising in recommender systems (for example, using the output of recommender systems to elicit users' personal details) [4, 6], to the knowledge of the authors, we are the first to consider attacking recommender systems with a view to manipulating the recommendations that are made.

In this paper, we extend our robustness analysis by examining the performance of memory-based collaborative filtering systems in detail. In particular, we consider several commonly-used neighbourhood formation schemes and similarity measures, and compare the robustness, accuracy and coverage provided by the various techniques.

2 Collaborative Filtering

We base our analysis on the frequently used collaborative filtering algorithm taken from the GroupLens project [7]. We model the collaborative filtering database, containing the past transactions of all users in the system, by a user-item matrix, in which users and items are represented by unique rows and columns respectively. Thus the $(i,j)^{th}$ entry represents the rating or vote, $v_{i,j}$, of user i for item j. A prediction, $p_{a,j}$, for user a (the active user) for an item j is made using

$$p_{a,j} = \bar{v}_a + \kappa \sum_{i=1}^{n} w(a,i)(v_{i,j} - \bar{v}_i) \tag{1}$$

where n is the neighbourhood size, $\bar{v}_a$ is the mean rating for user a, $w(a,i)$ is the similarity weight between users a and i and κ is a normalising factor. This equation essentially calculates a weighted average of users' ratings for the item in question with any variations in the means of users' rating distributions removed.

2.1 Neighbourhood Formation

Neighbourhood formation concerns the identification of similar users in a system that are used as predictors for the active user. Previous work indicates that neighbours with a high degree of similarity to the active user can be much more valuable as predictors than those with lower similarity, and therefore greater accuracy and performance can be achieved by limiting neighbourhood size [2]. In this paper, we

[1] Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland, email: michael.p.omahony@ucd.ie

consider two neighbourhood formation schemes and determine what effects, if any, each has on system robustness.

2.1.1 k - Nearest-Neighbour

The k - Nearest-Neighbour (k-NN) scheme [2, 8] forms a neighbourhood of size k by simply selecting the k most similar users. Generally, neighbourhood size is application-dependent, with an optimal size from the point-of-view of recommendation accuracy determined by experiment. The k-NN scheme has been shown to be quite accurate and provides good coverage [2].

2.1.2 Similarity Thresholding

In this scheme, a similarity threshold is set, and those users whose similarities with the active user exceed the threshold value are selected as neighbours [2, 9]. Setting a high threshold restricts the neighbourhood to very similar neighbours, but for many users, few (if any) close neighbours are available and coverage issues can therefore arise. If the threshold is set too low, the contributions of the best predictors tend to get lost in the noise from the other neighbours.

2.2 Similarity Measures

The weights $w(a, i)$ in (1) are calculated according to some similarity measure, and such measures are a critical component of memory-based algorithms. Here, we consider three measures that have been widely used in collaborative filtering systems.

2.2.1 Cosine Similarity

Cosine (or vector) similarity [1] is a widely used technique in which each user is represented by a vector in the $|I|$-dimensional item space, where I is the set of items available in a particular domain. The similarity between any two users a and i can be calculated as the cosine of the angle between the two corresponding vectors as follows

$$w(a,i) = \sum_j \frac{v_{a,j}}{\sqrt{\sum_{k\in I_a} v_{a,k}^2}} \frac{v_{i,j}}{\sqrt{\sum_{k\in I_i} v_{i,k}^2}} . \qquad (2)$$

where I_a is the set of items rated by user a. This weight gives a value between 0 and 1, and is 0 for any given pair of users who have not rated any common items, i.e. $w(a, i) = 0$ if $I_a \cap I_i = \phi$. Note that the weight tends to be higher if the items that users have in common are *liked* (i.e. receive high ratings) by both users. The normalisation terms in the denominator are important to ensure that users who have rated many items are not *a priori* more similar to other users.

2.2.2 Pearson Correlation

The Pearson correlation coefficient weighting, proposed in the GroupLens system [7], is defined as

$$w(a,i) = \sum_{j\in I_a\cap I_i} \frac{v_{a,j}}{\sqrt{\sum_{k\in I_a} v_{a,k}^2}} \frac{v_{i,j}}{\sqrt{\sum_{k\in I_i} v_{i,k}^2}} \qquad (3)$$

which results in a weight of between -1 and $+1$, and where the summations are over only those items that have been rated by both users. A correlation of $+1$ indicates perfect agreement between users, while a correlation of -1 indicates total disagreement. Unlike cosine similarity, Pearson correlation is not influenced by whether or not common items are liked by users, and it is possible to achieve perfect correlation between users provided that both users have *agreed* on the ratings given to items.

2.2.3 Spearman Rank Correlation

Pearson correlation is derived from a linear regression model, in which certain assumptions are made for the data being correlated. For example, a linear relationship is assumed to exist between the variables and variables are assumed to follow a normal distribution. Pearson correlation can become a much less accurate indicator of similarity if underlying model assumptions are violated. Thus, Spearman rank correlation, which is similar to Pearson, provides an alternative. In contrast to Pearson, Spearman [2] does not rely on any set of assumptions and is calculated over the ranks of the rating values, rather than the values themselves. It is defined as

$$w(a,i) = \frac{\sum_j (r_{a,j} - \bar{r}_a)(r_{i,j} - \bar{r}_i)}{\sqrt{\sum_{j\in I_a\cap I_i} (r_{a,j} - \bar{r}_a)^2 \sum_j (r_{i,j} - \bar{r}_i)^2}} \qquad (4)$$

and, as in the case of Pearson, results in a weight of between -1 and $+1$.

Similarity weights calculated on the basis of a small number of common items may often provide an unreliable measure of the "true" similarity between users. For example, Pearson and Spearman rank correlations give a result of either 1, -1 or 0 when the number of common items between users is 2. Significance weighting has been proposed in [2] as a solution to this problem, in which weights are modified based on the number of common items between users, n, as follows: $w'_{a,i} = w_{a,i} \times \frac{n}{N}$ if $n < N$, where N is an arbitrary constant. The weight is unchanged when $n \geq N$. In our experiments, we use this technique and set $N = 50$ as per [2].

3 Robustness

Security is a major concern for all Internet systems and applications, and yet, thus far, it seems little consideration has been given to the security of recommender systems. Even if one assumes that a system's database and recommendation algorithms are secure against attack, recommender systems remain vulnerable given the very manner in which they operate. Since it is practically impossible to assess the integrity of those who use a system (especially true for on-line systems), there is nothing to prevent users from inputting false data into the system.

Robustness is defined as the ability of a system to provide consistent or stable predictions given some degree of noise present in the data. In this paper, we consider noise that has been inserted into the data in the course of an attack. We evaluate robustness by defining *mean absolute prediction error* (MAPE) as the absolute difference between pre- and post-attack predictions. All the attacks considered in this paper are item-based (as opposed to user-based) – for example, in product push attacks, an attacker attempts to promote all predictions that are requested for targeted items, regardless of who requested them.

Let A_i be the set of users over which we evaluate an attack on item i. We calculate MAPE for item i as

$$\text{MAPE}(i) = \frac{1}{|A|} \sum_{a\in A} |p'_{a,i} - p_{a,i}| \qquad (5)$$

where $p'_{a,i}$ is the post-attack prediction. We then calculate the overall robustness of a system by taking the average MAPE over all items attacked. For product push and nuke attacks, we also report the *direction* of the prediction shifts achieved as per the above. This is necessary since, in product nuke attacks, for example, the objective is to force predictions toward the minimum rating, and thus negative shifts are required.

4 Attack Strategies

We examine the robustness of the collaborative filtering algorithm by mounting a product nuke attack on the system. The attack is implemented by creating bogus or false user profiles and inserting this data into the system through the normal user interface – no other access to the system database is assumed for an attacker.

From the attacker's perspective, there are two criteria that need to be satisfied if an attack is to be successful. In the first instance, if false profiles are to have any influence on predictions, they need to be present in the neighbourhood of targeted users. Therefore, these profiles need to be constructed to have a strong similarity to as many genuine users in the database as possible. One approach is to build false profiles using popular items since, by definition, many user profiles contain these items and ratings are likely to be consistent, and high.

Secondly, for Pearson and Spearman rank correlation measures, false profiles need to correlate in the same direction (i.e. either positively or negatively) with targeted users if all predictions made for an item are to be nuked. (If this is not the case, the opposite effect will be achieved, and the item will instead be pushed). Thus, along with the item to be nuked, the false profiles are constructed from two "groups" of items. The first group consists of items that are generally rated higher than average in the database (i.e. *liked* items), and the second group consists of items that are generally rated lower than average (*disliked* items). By assigning a high rating to the liked items and a lower rating to the disliked items, an attacker can be confident that these false profiles will correlate in the same direction with the majority of genuine users in the database.

Referring to (1), the contribution of any potential neighbour to a particular prediction depends on the magnitude or the difference term $(v_{i,j} - \bar{v}_i)$. For attack profiles, the obvious strategy is to choose ratings for the item groups that will maximise this term. Thus, the maximum rating, R_{max}, is assigned to each of the liked items and ratings of $R_{max} - 1$ to the disliked items. The item under attack is set to the minimum rating, R_{min}.

With regard to Cosine similarity, matters are simplified for attacks on systems that operate on *positive* rating scales. For such systems, Cosine can only result in a positive value, and thus, false profiles can be constructed by simply picking some popular items from the database. The items can be rated as before by setting the item under attack to R_{min} and, to allow for some variation, the remaining items are assigned ratings of R_{max} and $R_{max} - 1$.

While the above strategy requires a certain knowledge of the data contained in databases, it is not unreasonable to assume that such knowledge is possible to estimate (e.g. in movie domains) or to mine (e.g. on Amazon.com, using the feedback provided by users). Finally, we note that product push attacks can be implemented by using a similar strategy to that outlined above.

5 Evaluation

In our evaluation, we used the EachMovie collaborative filtering dataset provided by Compaq Equipment Corporation. The original dataset has some $72,916$ users who entered a total of $2,811,983$ numeric ratings for $1,628$ different movies. From this, we selected a random sample of $1,399$ users, containing $91,982$ transactions on a 6-point rating scale of 1 to 6 inclusive (modified from the original 6-point scale of 0 to 1). In addition, we used the MovieLens [3] dataset, which we found to give similar results to EachMovie – therefore we discuss the EachMovie data only.

In all experiments, there was very close agreement between Pearson and Spearman rank correlations. Similar findings have also been reported in [2]. This result was not too surprising since ratings are used to rank items, and considering that the ratings themselves are *de facto* ranks, little difference between the two was to be expected. Thus, in the analysis below, we present results for Pearson correlation and cosine similarity only.

The experimental procedure adopted in all cases was to remove a single user-item pair from the dataset, and a prediction for this pair was then made using all remaining data. To begin with, we examine the coverage and accuracy provided by the system, prior to attack.

Coverage. Figure 1 shows the coverage provided by Cosine similarity and Pearson correlation using similarity thresholding and k-NN neighbourhood formation schemes. Four similarity threshold values, T, were considered, ranging from 0.35 to 0.75. For k-NN, the neighbourhood size was fixed at 60. The k-NN scheme provided the best coverage, achieving 88% and 87% coverage for Cosine and Pearson respectively. Thresholding is known to result in reduced coverage, and this evident in our results. Coverage decreased significantly as the threshold was increased, and at $T = 0.65$, the coverage provided by both similarity measures was almost zero. Similar trends have been noted in [2, 9]. In addition, we found that Cosine gave better coverage than Pearson at threshold values of 0.35 and 0.5. For example, at $T = 0.35$, Cosine and Pearson provided coverage of 0.67 and 0.49 respectively.

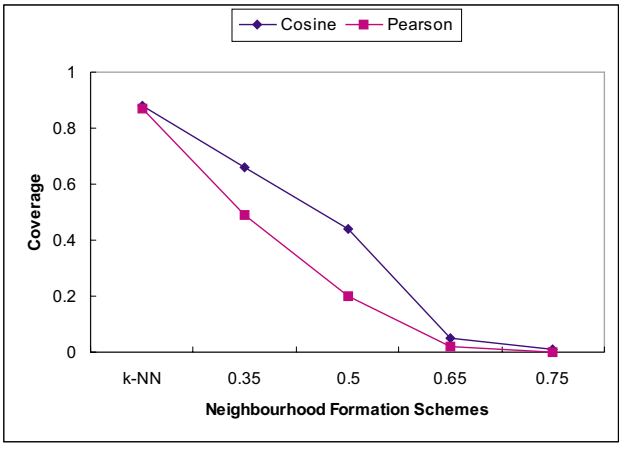


Figure 1. System coverage achieved by Cosine similarity and Pearson correlation using similarity thresholding and k-NN neighbourhood formation schemes

Accuracy. System accuracy calculated according to mean ab-

solute error (MAE) [9] is shown for the various similarity measures and neighbourhood formation schemes in Figure 2. Pearson gave marginally better (or no worse) accuracy than Cosine, except for thresholding at $T = 0.75$. However, this result was not significant since the coverage provided at this threshold value was very close to zero. The k-NN neighbourhood formation scheme performed best; closely matched by thresholding at $T = 0.35$. Thereafter, accuracy began to fall off as T was increased. These findings are in agreement with those reported in [2, 9].

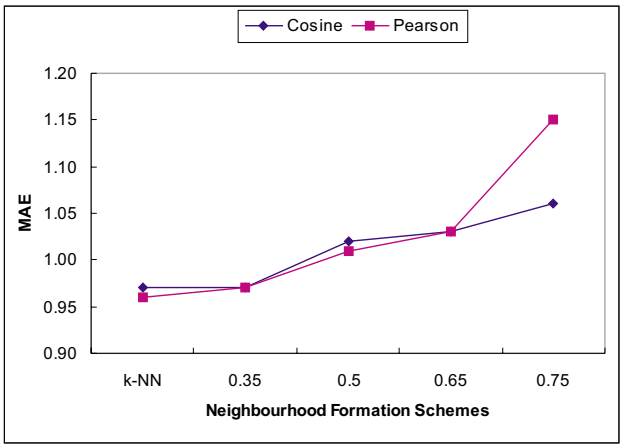


Figure 2. Accuracy according to MAE for Cosine similarity and Pearson correlation using similarity thresholding and k-NN neighbourhood formation schemes

Robustness. In Figure 3, we present robustness according to (5) for an attack strength of 70 false profiles inserted into the database. While there was little difference in the results between the two similarity measures, it is clear that the neighbourhood formation schemes had a significant impact on robustness. The k-NN scheme performed the worst, resulting in an MAPE of approximately 2.35 for both similarity measures. Similarity thresholding gave an MAPE of approximately 1 at $T = 0.35$. At higher threshold values, the scheme was essentially robust.

To explain the above, recall that the attack strategy involved creating false profiles that have a good probability of being similar to many users in the system. For k-NN, neighbourhood sizes were fixed at 60, and since users did not typically have 60 *very* similar neighbours, the result was that false profiles were included in many of the neighbourhoods. For thresholding at $T = 0.35$ and above, false profiles were located in the neighbourhoods of increasingly fewer users, and thus, a corresponding decrease in MAPE was observed. At the higher threshold values, false profiles were almost completely filtered from the neighbourhoods.

Note that, for Pearson, the strategy used to control the direction of prediction shifts was successful. On average, 98% of all shifts achieved were in the negative direction, as required for product nuke attacks.

Finally, in Figure 4, we show the effect of attack strength on robustness for k-NN. As more false profiles were inserted into the database, MAPE increased until, at a certain point, no further decrease in robustness was observed. This point coincides with the neighbourhood size. In our attacks, we used the same false profile

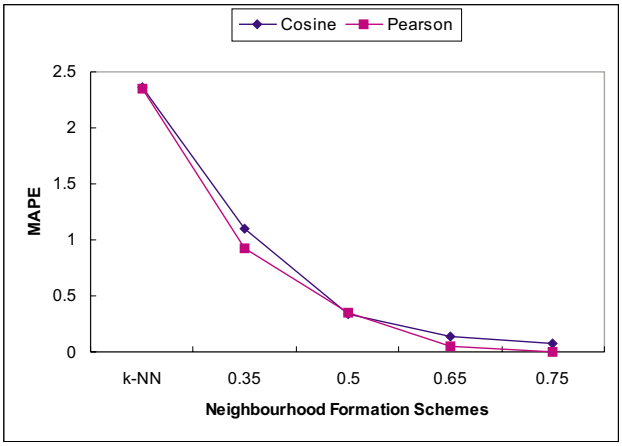


Figure 3. Robustness according to MAPE for an attack strength of 70 false profiles inserted into the database

repeatedly, and therefore no additional effect can be achieved by inserting more than this number of false profiles into the system. Note that the attack achieved sizable prediction shifts even at low attack strengths. Because false profiles have been designed to cause the maximal prediction shifts, significant influence can be exerted by them on predictions, even when relatively few are present in the system.

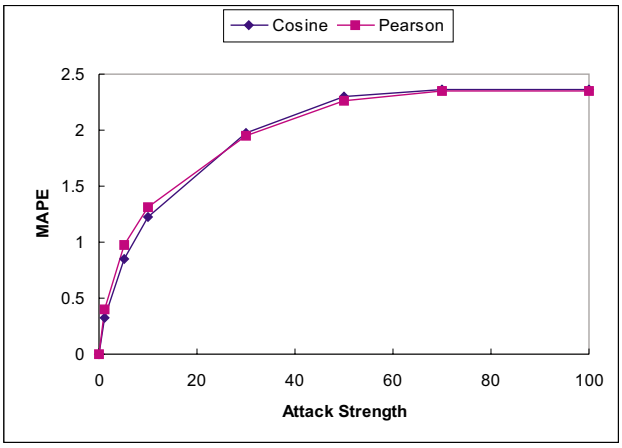


Figure 4. The effect of attack strength on system robustness using the k-NN neighbourhood formation scheme

6 Conclusion

In this paper, we have examined the effect of various neighbourhood formation schemes and similarity measures on the performance of memory-based collaborative filtering systems. Our results have clearly indicated that systems which use the k-NN neighbourhood formation scheme are vulnerable to attack. There was relatively little in the results to distinguish between the performance of the similarity measures tested. However, for systems that operate on positive rating scales, product push and nuke attacks are more straightforward to implement (and consequently more likely to succeed) when Cosine similarity is used.

In future work, we will examine techniques to improve the robustness of k-NN, given that this scheme was the most accurate and provided excellent coverage compared to the other schemes tested.

REFERENCES

[1] John S. Breese, David Heckerman, and Carl Kadie, 'Empirical analysis of predictive algorithms for collaborative filtering', in *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI '98), San Francisco, CA, USA*, pp. 43–52, (July 1998).

[2] Jonathan Herlocker, Joseph Konstan, Al Borchers, and John Riedl, 'An algorithmic framework for performing collaborative filtering', in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, Berkeley, CA, USA*, pp. 230–237, (August 1999).

[3] http://movielens.umn.edu/.

[4] Douglas W. Oard, Anton Leuski, and Stuart Stubblebine, 'Protecting the privacy of observable behavior in distributed recommender systems', in *Proceedings of SIGIR'03 Workshop on Implicit Measures of User Interests and Preferences, Toronto, Canada*, (August 2003).

[5] Michael P. O'Mahony, Neil J. Hurley, and Guenole C. M. Silvestre, 'Towards robust collaborative filtering', in *Proceedings of the 13th Irish International Conference, AICS 2002, Limerick, Ireland*, pp. 87–94, (September 2002).

[6] Naren Ramakrishnan, Benjamin J. Keller, Batul J. Mirza, Ananth Y. Grama, and George Karypis, 'Privacy risks in recommender systems', in *IEEE Internet Computing (Vol. 5, No. 6)*, pp. 54–62, (November/December 2001).

[7] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J.Riedl, 'Grouplens: An open architecture for collaborative filtering of netnews', in *Proceedings of the ACM Conference on Computer Supported Cooperative Work, Chapel Hill, NC, USA*, pp. 175–186, (October 1994).

[8] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Riedl, 'Analysis of recommendation algorithms for e-commerce', in *Proceedings of the 2nd ACM Conference on Electronic Commerce (EC-00), Minneapolis, MN, USA*, pp. 158–167, (October 2000).

[9] Upendra Shardanand and Patti Maes, 'Social information filtering: Algorithms for automating word of mouth', in *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems, Denver, CO, USA*, pp. 210–217, (May 1995).

A new Simple recurrent network with Real time recurrent learning process

Tarik Rashid and **BingQuan Huang** and **Tahar Kechadi**[1]

Abstract. The simple recurrent network (SRN) is one of the most attractive types of recurrent neural networks (RNN), which deals with temporal sequences. The SRN has been used to handle many tasks, for example prediction and classification. Nevertheless, the SRN trained with back propagation through time (BPTT) has some limitations which restrict the real time application. To avoid these limitations, real time recurrent learning (RTRL) is used to train SRNs. The training speed of the SRN with RTRL is not fast. In this paper we propose a modified network trained with RTRL to try to solve these problems. Based on SRN, the modified architecture adds two extra parts: a Multi-Context Layer (MCL), and feed forward connections from the MCL to the output layer. This paper includes the full mathematical model derived according to this model architecture and learning algorithm. Some simple applications are implemented with the modified network.

1 Introduction

The Elman Network [6], [5], is a common type of recurrent network [2], [3], and the network is referred to as an SRN [1], or called by Karjala and Cheng as an internally recurrent network (IRN) [14]. The network architecture is made up of four layers: an input layer, a hidden layer, an output layer, and a context layer that acts as an internal input to the network. The network layers are connected in two directions: feed forward (one to many) connections from the input and context layers to the hidden layer, from the hidden layer to the output layer, and feed backward or recurrent (one to one) connections from the hidden layer to the context layer.

The Elman network would work as follows: at time t_0, the input layer receives an external input pattern as either a scalar or a vector depending on the nature of the task. Initially the context layer neurons might have values in the range between 0 and 1, the neurons in the hidden layer receive activations from both input and context neurons, and then the hidden neurons simultaneously feed forward and backward to the output neurons and context neurons respectively. The output in the output layer is compared with the target, and the error back propagates through the network layers to modify the connection weights. The recurrent connection weights from the hidden layer to the context layer as stated by Elman; are not exposed to modification [5]. The same process is repeated at subsequent time steps. The Elman network trained with BPTT [6], [5], the BPTT algorithm creates an extra hidden layer for each previous time step; this is called unfolding the temporal sequences into a feed forward network [10], [9], for which it is possible to use the ordinary error back propagation [9] technique. An Elman Network trained with BPTT has been successfully used for many tasks such as prediction and classification [1], [5]. However it fails to handle more real-life tasks, because the entire time sequences must be used; therefore, the memory and computation grow proportionally with sequence length, and for the long sequences, this technique is not practical. To improve the above limitations researchers used recurrent networks with RTRL [12], [11], [4] or fast real time recurrent learning algorithm (FRTRL) which was used to perform text-to-phone conversation [8]. The RTRL in [12] updates the weights at every time step, it can deal with sequences arbitrary of length, it has no need for memory storage proportional to sequence length. Because of the above properties associated with the RTRL, it is suitable for use in the proposed model of on-line handwriting recognition.

On-line handwriting refers to the case where the pen coordinates as a function of time are known and not only a static image of words written on paper. The quality of on-line handwriting recognition has been a great obstacle to the success of pen computing. This task is considered to be one of the fuzzy real-world problems because the input patterns contain potentially fuzzy features; this is something very common in on-line handwriting recognition. Specifically the problems are considered to be writer independence, time calculation, the variable sequence of the length of the example patterns and the large vocabulary which contains (at least 25- 50) common words; this vocabulary will be able to use handwriting for text entry applications.

The differences between the new model and the SRN are firstly, that the network has an MCL, which can keep more states of the history and accelerate the training sessions. The experiments described by Wilson (1993, 1995) demonstrated that, for the training task used, more state vectors meant faster learning [13]. Secondly, the feed forward connections from the multi-context layers to the output layer also speed up the learning of the network and reduce the number of neurons in the hidden layer [4]. The performance of our network is compared to the Elman network, and the experimental results are shown in section (4).

2 Network Architecture

The network architecture consists of an MCL, with feed forward connections from MCL to the output layer. See Figure 1.

In order to explain the functionality of this network, let us introduce some basic notations and definitions. The Output function $f(x)$ can be chosen to be the logistic function:

$$f(x) = \frac{1}{1 + e^{-x}}$$

Where x represents the net input. The derivative of the activation function is:

[1] Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland, email:Tarik.rashid@ucd.ie, bingquan.huang@ucd.ie, tahar.kechadi@ucd.ie

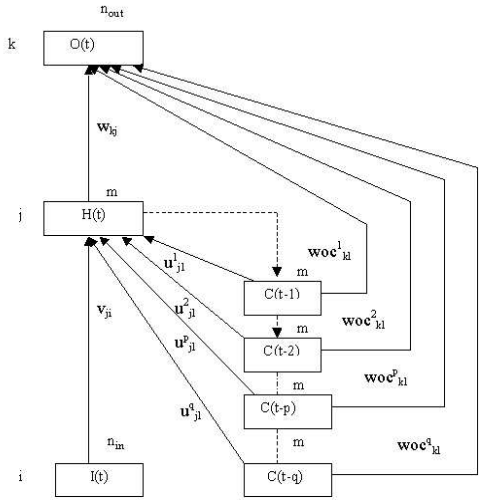


Figure 1. The modified network structure is composed of the SRN and the MCL with feed forward connections to the output layer.

$$f'(x) = (1 - f(x))f(x)$$

In the following, we introduce the Kronecker delta to be:

$$\delta_{ab} = \begin{cases} 0 & \text{when } a \neq b \\ 1 & \text{when } a = b \end{cases}$$

Some basic notations:

1. Neurons and layers
   - i, i' are the indices for the input neurons.
   - j, j', j'', j''' are the indices for the hidden layer neurons.
   - l, l' are the indices for the context layers neurons.
   - k, k', k'', k''' are the indices for output layer neurons.
   - n_{in} is the number of the input layer neurons.
   - n_{out} is the number of the output layer neurons.
   - q is the number of the context layers.
   - p / p' is the pth context layer.
2. Net inputs and outputs
   - $I_i(t)$ is the input of neuron i in the input layer at time t.
   - $\tilde{h}_j(t)$ is the net input to the neuron j in the hidden layer at time t.
   - $\tilde{o}_k(t)$ is the net input to the neuron k in the output layer at time t.
   - $H_j(t)$ is the output of the neuron j in the hidden layer at time t.
   - C_j(t-p) is the output of the neuron j in the pth context layer at time t.
   - $O_k(t)$ is the output of neuron k in the output layer at time t.
   - $d_k(t)$ is the target of neuron k in the output layer at time t.
3. Connection weights
   - $v_{ji}(t)$ is the weight connection from the input layer to the hidden layer.
   - $u^p_{jl}(t)$ is the weight connection from the pth context layer to the hidden layer.
   - $woc^p_{kl}(t)$ is the weight connection from the pth context layer to the output layer.
   - $w_{kj}(t)$ is the weight connection from the hidden layer to the output layer.

3 Learning Algorithm

A RTRL algorithm is used to train the new model. The procedures of the training follow two directions – Feed Forward Pass and Backward Pass . The computations following "feed forward pass" include calculating all the net outputs, and their activation values of hidden layer neurons, context layers neurons, output layer neurons, and calculating errors. The computations following "backward pass" include calculating the change of the error of the network with respect to the change of net output for each neuron in the output, hidden, and context layers. All the weights will be modified at every time step. The mathematical formulae for RTRL were derived in accordance with [9], [7].

3.1 Feed Forward Pass

We assume that the n_{con} is the total number of the active context layers:

$$n_{con} = \begin{cases} t & \text{when } t < q \\ q & \text{when } t >= q \end{cases} \quad (1)$$

q is the number of the context layers.

1. Calculate the outputs of the hidden layer neurons:
   The net input to the jth hidden layer neuron is the weighted sum of the separate outputs from each of the connected neurons of the input layer plus the weighted sum of the separate outputs from each of the connected neurons of the context layers.

$$\tilde{h}_j(t) = \sum_{i=1}^{n_{in}} I_i(t)v_{ji}(t) + \sum_{p=1}^{n_{con}} \sum_{l=1}^{m} C_l(t-p)u^p_{jl}(t) \quad (2)$$

   The output of the jth neuron in the hidden layer:

$$H_j(t) = f(\tilde{h}_j(t))$$

   Updating the outputs of the p context layers is done after the weights of all connections are updated. This is implemented in two phases: The first involves removing the oldest outputs of the context layers, and the second phase concerns copying the outputs of the hidden layer to the first context layer. Firstly, equation (3) is used to remove the history by copying the outputs of the first context layer to the next context layer, and copying the next context layer to the following one, and so on. Secondly, equation (4) is used to copy the outputs of the hidden layer to the first context layer.

$$C_j(t-p) = C_j(t-p+1) \quad (3)$$

$$C_j(t-1) = H_j(t) \quad (4)$$

2. Calculate the outputs of the output layer neurons:
   The net input to the kth output layer neuron is the weighted sum of the separate outputs from each of the connected neurons of the

hidden layer plus the weighted sum of the separate outputs from each of the connected neurons of the context layers.

$$\tilde{o}_k(t) = \sum_{j=1}^{m} H_j(t) w_{kj}(t) + \sum_{p=1}^{n_{con}} \sum_{l=1}^{m} C_l(t-p) woc_{kl}^{p}(t) \quad (5)$$

The output of the kth neuron on output layer:

$$O_k(t) = f(\tilde{o}_k(t))$$

3.2 Backward Pass

3.2.1 Calculating Error

The difference between the target and its actual output in the output layer will be defined as $e_k(t)$:

$$e_k(t) = d_k(t) - O_k(t) \quad (6)$$

Define the instantaneous sum of squares error at time t as:

$$E(t) = \frac{1}{2} \sum_{k=1}^{n_{out}} (d_k(t) - O_k(t))^2 = \frac{1}{2} \sum_{k=1}^{n_{out}} (e_k(t))^2 \quad (7)$$

The objective is to minimize the total error of the network; it can be obtained by summing $E(t)$ over all past steps of the network and this can be expressed as:

$$E_{total} = \sum_{t=1}^{T} E(t) \quad (8)$$

When the network has correctly learned all the training examples, E_{total} becomes close to zero. More specifically, the total error which determines the performance of the network should be a sum over all patterns in the training data set. However, this summation depends on the nature of the task to be handled and the nature of the training set.

3.2.2 Computing Error Gradients

The following describe how to compute the error gradients:

$\frac{\partial E(t)}{\partial w_{kj}(t)}$, $\frac{\partial E(t)}{\partial woc_{kl}^{p}(t)}$, $\frac{\partial E(t)}{\partial u_{jl}^{p}(t)}$, $\frac{\partial E(t)}{\partial v_{ji}(t)}$

1. Computing $\frac{\partial E(t)}{\partial w_{kj}(t)}$
   Using chain rule.

$$\frac{\partial E(t)}{\partial w_{kj}(t)} = \frac{\partial E(t)}{\partial e_k(t)} \frac{\partial e_k(t)}{\partial O_k(t)} \frac{\partial O_k(t)}{\partial \tilde{o}_k(t)} \frac{\partial \tilde{o}_k(t)}{\partial w_{kj}(t)} \quad (9)$$

   we calculate the error gradients by calculating every sub-part of the equation (9).

$$\frac{\partial E(t)}{\partial e_k(t)} = \frac{\partial \left(\frac{1}{2} \sum_{k=1}^{n_{out}} (e_k(t))^2 \right)}{\partial e_k(t)} = e_k(t) \quad (10)$$

$$\frac{\partial e_k(t)}{\partial O_k(t)} = \frac{\partial}{\partial O_k(t)} (d_k(t) - O_k(t)) = -1 \quad (11)$$

$$\begin{aligned} \frac{\partial O_k(t)}{\partial \tilde{o}_k(t)} &= \frac{\partial f(\tilde{o}_k(t))}{\partial \tilde{o}_k(t)} = f'(\tilde{o}_k(t)) \\ &= (1 - f(\tilde{o}_k(t)) f(\tilde{o}_k(t)) \end{aligned} \quad (12)$$

$$\begin{aligned} \frac{\partial \tilde{o}_k(t)}{\partial w_{kj}(t)} &= \frac{\partial}{\partial w_{kj}(t)} \left(\sum_{j=1}^{m} H_j(t) w_{kj}(t) + \sum_{p=1}^{n_{con}} \sum_{l=1}^{m} C_l(t-p) woc_{kl}^{p}(t) \right) \\ &= H_j(t) \end{aligned} \quad (13)$$

We assume $LG_k(t)$ is the local gradient of the kth neuron in the output layer, and can be expressed as follows:

$$LG_k(t) = e_k(t) (1 - f(\tilde{o}_k(t))) f(\tilde{o}_k(t)) \quad (14)$$

Substitute the equations (10), (11), (12), and (13) in (9), so $\frac{\partial E(t)}{\partial w_{kj}(t)}$ can be obtained as follows:

$$\frac{\partial E(t)}{\partial w_{kj}(t)} = -e_k(t) (1 - f(\tilde{o}_k(t)) f(\tilde{o}_k(t)) H_j(t) \quad (15)$$

Place (14) in the above, $\frac{\partial E(t)}{\partial w_{kj}(t)}$ can be written in this way:

$$\frac{\partial E(t)}{\partial w_{kj}(t)} = -LG_k(t) H_j(t) \quad (16)$$

2. Computing $\frac{\partial E(t)}{\partial woc_{kl}^{p}(t)}$
   To compute the value of $\frac{\partial E(t)}{\partial woc_{kl}^{p}(t)}$, we follow the same procedure, which is used to compute the value of $\frac{\partial E(t)}{\partial w_{kj}(t)}$.
   The simplified form:

$$\frac{\partial E(t)}{\partial woc_{kl}^{p}(t)} = -e_k(t) (1 - f(\tilde{o}_k(t))) f(\tilde{o}_k(t)) H_l(t-p)$$

   The variable $H_l(t-p)$ in the above can be represented by the variable $C_l(t-p)$.

$$= -LG_k(t) C_l(t-p) \quad (17)$$

3. Computing $\frac{\partial E(t)}{\partial u_{jl}^{p}(t)}$

$$\frac{\partial E(t)}{\partial u_{jl}^{p}(t)} = \sum_{k=1}^{n_{out}} e_k(t) \frac{\partial e_k(t)}{\partial u_{jl}^{p}(t)} \quad (18)$$

   By taking the term $\frac{\partial e_k(t)}{\partial u_{jl}^{p}(t)}$ of the above equation and implementing the chain rule.

$$\frac{\partial e_k(t)}{\partial u_{jl}^{p}(t)} = \frac{\partial e_k(t)}{\partial O_k(t)} \frac{\partial O_k(t)}{\partial \tilde{o}_k(t)} \frac{\partial \tilde{o}_k(t)}{\partial u_{jl}^{p}(t)}$$

   Substitute $\frac{\partial e_k(t)}{\partial u_{jl}^{p}(t)}$ in equation (18) to get equation (19).

$$\frac{\partial E(t)}{\partial u_{jl}^{p}(t)} = \sum_{k=1}^{n_{out}} e_k(t) \frac{\partial e_k(t)}{\partial O_k(t)} \frac{\partial O_k(t)}{\partial \tilde{o}_k(t)} \frac{\partial \tilde{o}_k(t)}{\partial u_{jl}^{p}(t)} \quad (19)$$

   We assume that the $LG_j(t)$ is the local gradient of the jth neuron in the hidden layer. It can be obtained from the sum of all the local gradients of the neurons on the output layer multiplied by

their connection weights from the hidden to the output layer.

$$LG_j(t) = \sum_{k=1}^{n_{out}} LG_k(t) w_{kj}(t) \quad (20)$$

We assume that the $LG_l^p(t)$ is the local gradient of lth neuron in the pth context layer. It can be obtained from the sum of all the local gradients of the neurons on the output layer multiplied by their connection weights from the context layer to the output layer.

$$LG_l^p(t) = \sum_{k=1}^{n_{out}} LG_k(t) woc_{kl}^p(t) \quad (21)$$

if we can obtain the result of $\frac{\partial \tilde{o}_k(t)}{\partial u_{jl}^p(t)}$, the error gradient can also be obtained.

$$\begin{aligned} \frac{\partial \tilde{o}_k(t)}{\partial u_{jl}^p(t)} &= \sum_{j'=1}^{m} \frac{\partial H_{j'}(t)}{\partial u_{jl}(t)} w_{kj'}(t) + \\ & \sum_{p'=1}^{n_{con}} \sum_{r=1}^{m} \frac{\partial H_r(t-p')}{\partial u_{jl}^p(t)} woc_{kr}^{p'}(t) \end{aligned} \quad (22)$$

The context variable $\partial C_r(t-p')$ in the second term in (22) was represented by the hidden variable $\partial H_r(t-p')$.

if we substitute equations (11), (12), and (22) in (19), then the $\frac{\partial E(t)}{\partial u_{jl}^p}(t)$ can be obtained as below:

$$\begin{aligned} \frac{\partial E(t)}{\partial u_{jl}^p}(t) &= \sum_{k=1}^{n_{out}} e_k(t)(-1)\left(1 - f(\tilde{o}_k(t))\right) f(\tilde{o}_k(t)) \times \\ & \left(\sum_{j'=1}^{m} \frac{\partial H_{j'}(t)}{\partial u_{jl}^p(t)} w_{kj'}(t) + \right. \\ & \left. \sum_{p'=1}^{n_{con}} \sum_{r=1}^{m} \frac{\partial H_r(t-p')}{\partial u_{jl}^p(t)} woc_{kr}^{p'}(t) \right) \end{aligned}$$

Place (14), (20), and (21) in the above.

$$\begin{aligned} \frac{\partial E(t)}{\partial u_{ji}^p(t)} &= -\sum_{k=1}^{n_{out}} LG_k(t) \left(\sum_{j'=1}^{m} \left[\frac{\partial H_{j'(t)}}{\partial u_{jl}^p(t)} w_{kj'}(t) + \right. \right. \\ & \left. \left. \sum_{p'=1}^{n_{con}} \sum_{r=1}^{m} woc_{kr}^{p'}(t) \delta_{rj'} \frac{\partial H_{j'}(t-p')}{\partial u_{jl}^p(t)} \right] \right) \\ &= -\sum_{j'=1}^{m} \left[LG_{j'}(t) \frac{\partial H_{j'}(t)}{\partial u_{jl}^p(t)} + \right. \\ & \left. \sum_{p'=1}^{n_{con}} \sum_{r=1}^{m} LG_r^{p'}(t) \delta_{rj'} \frac{\partial H_{j'}(t-p')}{\partial u_{jl}^p(t)} \right] \end{aligned} \quad (23)$$

We need to clarify $\frac{\partial H_{j'}(t)}{\partial u_{jl}^p(t)}$

$$\frac{\partial H_{j'}(t)}{\partial u_{jl}^p(t)} = \frac{\partial H_{j'}(t)}{\partial \tilde{h}_{j'}(t)} \frac{\partial \tilde{h}_{j'}(t)}{\partial u_{jl}^p(t)}$$

$$= f'(\tilde{h}_{j'}(t)) \frac{\partial (\sum_{i=1}^{n_{in}} v_{j'i}(t) I_i(t) + \sum_{p'=1}^{n_{con}} \sum_{l'=1}^{m} u_{j'l'}^{p'}(t) H_{j'}(t-p'))}{\partial u_{jl}^p(t)}$$

The context variable, $C_{j'}(t-p')$ in the above was replaced by the hidden variable, $H_{j'}(t-p')$.

$$\begin{aligned} &= f'(\tilde{h}_{j'}(t)) \left(\delta_{j'j} \sum_{p''=1}^{n_{con}} \delta_{pp''} H_l(t-p'') + \right. \\ & \left. \sum_{p'=1}^{n_{con}} \sum_{j''=1}^{m} \sum_{l'=1}^{m} u_{jl'}^{p'}(t) \delta_{l'j''} \frac{\partial H_{j''}(t-p')}{\partial u_{jl}^p(t)} \right) \end{aligned} \quad (24)$$

The initial condition (time $t = 0$), the value of $\frac{\partial H_{j'}(0)}{\partial u_{jl}^p(0)} = 0$. By having the initial value of $\frac{\partial H_{j'}(t)}{\partial u_{jl}^p(t)}$, we can obtain all the values recursively.

4. Computing $\frac{\partial E(t)}{\partial v_{ji}(t)}$

To compute the value of $\frac{\partial E(t)}{\partial v_{ji}(t)}$, we follow the same procedure, which is used to compute the value of $\frac{\partial E(t)}{\partial u_{jl}^p(t)}$.

The simplified form:

$$\begin{aligned} \frac{\partial E(t)}{\partial v_{ji}(t)} &= -\sum_{j'=1}^{m} \left[LG_{j'}(t) \frac{\partial H_{j'}(t)}{\partial v_{ji}(t)} + \right. \\ & \left. \sum_{p'=1}^{n_{con}} \sum_{l=1}^{m} LG_l^{p'}(t) \delta_{lj'} \frac{\partial H_{j'}(t-p')}{\partial v_{ji}(t)} \right] \end{aligned} \quad (25)$$

and

$$\begin{aligned} \frac{\partial H_{j'}(t)}{\partial v_{ji}(t)} &= \left((1 - f(\tilde{h}_{j'}(t)) f(\tilde{h}_{j'}(t)) \right) [\delta_{j'j} I_i(t) + \\ & \left. \sum_{p'=1}^{n_{con}} \sum_{j''=1}^{m} \sum_{l=1}^{m} u_{j'l}^{p'} \delta_{j'j} \frac{\partial H_{j''}(t-p')}{\partial v_{ji}(t)} \right] \end{aligned} \quad (26)$$

The initial condition (time $t = 0$), the value of $\frac{\partial H_{j'}(0)}{\partial v_{ji}(0)} = 0$. By having the initial value, we can obtain all the values recursively.

3.2.3 Computing the Change of the Weights

To speed up the training of a network, normally the momentum technique is used in a training algorithm, because it can avoid the local minima error. So the equations (16), (17), (23), and (25) can be rewritten to be (27), (28), (29), and (30), respectively.

$$\triangle w_{kj}(t) = \mu \frac{\partial E(t)}{\partial w_{kj}(t)} + \beta \Delta w_{kj}(t-1) \quad (27)$$

$$\triangle woc_{kl}^p(t) = \mu \frac{\partial E(t)}{\partial woc_{kl}^p(t)} + \beta \Delta woc_{kl}^p(t-1) \quad (28)$$

$$\triangle u_{jl}^p(t) = \mu \frac{\partial E(t)}{\partial u_{jl}^p(t)} + \beta \Delta u_{jl}^p(t-1) \quad (29)$$

$$\triangle v_{ji}(t) = \mu \frac{\partial E(t)}{\partial v_{ji}(t)} + \beta \Delta v_{ji}(t-1) \quad (30)$$

The initial condition (time $t = 0$), all the change of the weights ($\Delta w_{kj}(t)$, $\Delta woc^p_{kl}(t)$, $\Delta u^p_{jl}(t)$, $\Delta v_{ji}(t)$) are set to zero.

3.2.4 Adjusting the Weights

After the weight changes were computed, according to the equations below, all the weights can be updated for the next time step until the proper weights are obtained. (The proper weights can make the network total error to be zero or close to zero.)

$$w_{kj}(t+1) = w_{kj}(t) + \Delta w_{kj}(t) \quad (31)$$

$$woc^p_{kl}(t+1) = woc^p_{kl}(t) + \Delta woc^p_{kl}(t) \quad (32)$$

$$u^p_{jl}(t+1) = u^p_{jl}(t) + \Delta u^p_{jl}(t) \quad (33)$$

$$v_{ji}(t+1) = v_{ji}(t) + \Delta v_{ji}(t) \quad (34)$$

4 Simulation & Experimental results

Different applications were used to test the new model with RTRL. However, in this paper, only two, namely the shaping of the figure 8 and generalization of the digit 3, will be presented.

1. Shaping of the figure 8
   Shaping of the figure 8 is a prediction task. The network is tested to predict the shape of the figure 8. 16 points in cartesian coordinates (x, y) in the range (0-1) were selected to represent the shape of the figure 8. The patterns were designed as follows: The target of an input pattern is the next input pattern, and the target of the last input pattern is the first input pattern. The pattern (0.5, 0.5) is repeated, and represents the crossing center of the shape. By taking any arbitrary point including the crossing center point from the data set and feeding it to the network, the actual output of the selected point will be feed back again as a new input point to the network: This process continues until the shape of figure 8 has correctly emerged. Various tests were carried out with different parameters to compare the performance of new model with that of an Elman network trained with RTRL. To fit the above task, the new model was structured as follows: 2–8–8*2–2: 2 input neurons, 8 hidden neurons, 2 context layers each one with 8 neurons and 2 output neurons. The complexity of the new model was 28 neurons against an Elman network 2–8–8–2, where the complexity was 20. The new model created the shape 8, more quicker with less recycling of points to the network and convergence performance in terms of square error diminished more smoothly compared to the Elman network. Figure 2, (a) and (b) show the Elman network and the new model trained with RTRL for 20,000 cycles respectively. As can be seen the new model requires less shaping cycles than the Elman. (b), was shaped exactly on the crossing center point, while (a) was slightly biased. (c), shows the new model consisting of 2–8–8*2–2 has learned the task with 10,000 cycles. (d) shows that the new model consisting of 2–6–6*2–2 has learned the task with 15,000 cycles.

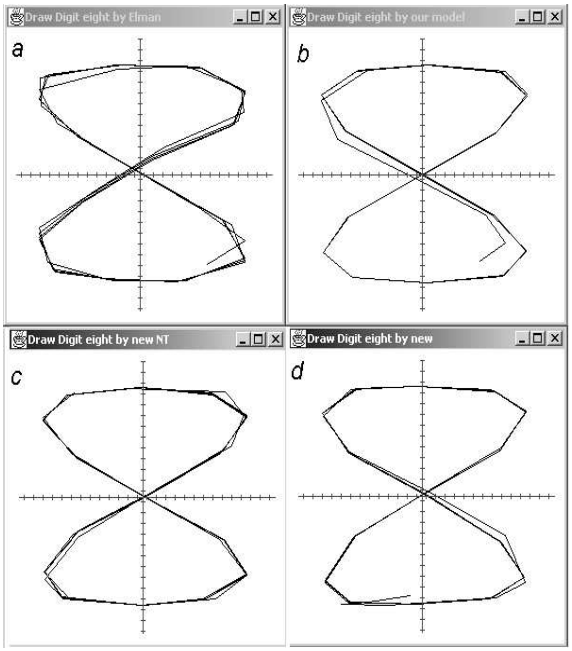


Figure 2. The new model (b) 2–8–8*2–2, which was shaped more accurately and quickly with fewer shaping cycles than Elman network (a) 2–8–8–2. Both networks were trained with 20,000 cycles. (c) and (d) represent the new model, (c) consisted of 2–8–8*2–2 with 10,000 cycles. (d) consisted of 2–6-6*2–2 with 15,000 cycles. Both (c) and (d) have learned the task.

2. Generalization of the digit 3
   This problem described here is very simple, and it demonstrates the network's ability to generalize. The network is able to generalize most effectively when it is tested against slightly corrupted input data. The generalization is affected mainly by the nature of the data set used. Here, the digit 3 was selected to demonstrate generalization. The problem can mainly be solved by multi-layer neural network (MLNN). Whereby the input patterns are spatial and presented independently in a non-dynamical manner. This means that time is irrelevant. Usually the training algorithm and sessions for MLNN are easier and faster than for the RNN. By removing an MCL in fig.1, the network would behave as an MLNN. The purpose of implementing this task on the new model is to show how flexible the new model is at taking spatial patterns and dealing with them in a temporal sequence. 10 different samples were chosen to represent the digit 3. Each sample was created from (4 X 5) pixels and each pixel took the value of either 1 or 0 (on or off). A sample is presented to the network by feeding in 4 input pixels at every time step. The network structure for the new model adopted for this problem was 4–1–1* 3–4: 4 input neurons, 1 hidden neuron, 3 context layers; each one with one neuron and 4 output neurons, so the complexity of the new model was 12 neurons against the Elman network which was consisting of 4–10–10–4 (Elman needs more neurons to learn the problem), the complexity of the Elman network was 28 neurons. The new model trained with RTRL and standard back propagation (SB) learned the task and produced an MSE of 0.003, 0.042 respectively, while an Elman network trained with SB learned the task with an MSE of 0.125. Figure 3 shows that the effect of context layers will favor the power of the new model over the Elman network. Table 1 displays the input patterns that were presented to the network and the

actual outputs that were produced compared with the targets.
Other experiments were carried out to examine the effect of the direct connections from the context layer to the output layer. Again two networks were implemented: One was the new model which consisted of 5–3–3*1–5 trained with SB: This time the new model had one context layer and direct feed forward connection from the context layer to the output layer. The new model has learned the task and produced an MSE of 0.01 against an Elman network 5-3-3-5 trained with SB produced an MSE of 0.205. As can be seen by using 5 neurons instead of 4 neurons in the input and output layer, the training session became faster (less sequence of length was used).
Due to the simplicity of the data set's size and format of the digit 3 task which speeds up the computations, one could argue that the small number of pixels used means that some of the original information about the digit has been lost but the complexity of the problem increases as the network deals with larger data sets of pixels, and so the number of hidden neurons required also increased.

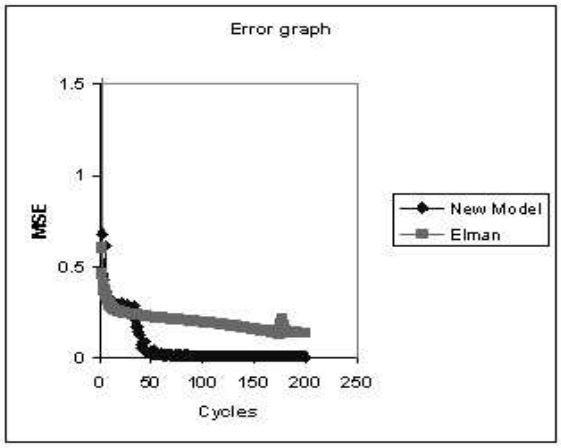


Figure 3. Error graph shows the power effect of the modification to the Elman network over the Elman network by adding the MCL and the direct feed forward from MCL to the output layer. Both networks trained for 200 cycles.

Table 1. This table displays the input patterns, actual outputs, and targets for the digit 3

Input	Actual output	Target
0 1 1 0	0.999 0.989 0.989 0.009	1 1 1 0
0 0 0 1	0.002 0.001 0.001 0.997	0 0 0 1
0 1 1 0	0.010 0.993 0.993 0.006	0 1 1 0
0 0 0 1	0.000 0.000 0.000 0.999	0 0 0 1
1 1 1 0	0.984 0.999 0.999 0.000	1 1 1 0

5 Conclusion & Future work

We introduced a new simple recurrent network based on the Elman network. We generated a full mathematical model for this network trained with RTRL. The most important part in the mathematical model is the error gradients, which has to be minimized using RTRL. Simulation for both modified simple recurrent and Elman were implemented and compared, the results show that:

1. Our network model with RTRL trains faster than Elman model. A suitable number of context layers(2, 3 or 4) might be selected so that the number of the training cycles can be minimized.
2. Mean Square Error decreases rapidly and it is smoother than the Elman network with RTRL. This improves the convergence of the network.
3. Fewer hidden neurons are needed in the new model, while in the Elman it is not possible to reduce the number of the hidden neurons used after a certain threshold. This threshold is much higher in Elman than our model.

Our primary goal is to study this new model in more detail, to carry out various tests, training, and optimize architecture of the network to handle real-world online handwriting recognition.

REFERENCES

[1] A. Cleeremans, D. Servan-Schreiber, and J. McClelland, 'Finite state automata and simple recurrent neural networks', *Neural Computation*, **1**(3), 372–381, (1989).
[2] D.S. Clouse, C.L. Giles, B.G. Horne, and G.W. Cottrell, 'Time-delay neural networks: Representation and induction of finite state machines', *IEEE Transactions on Neural Networks*, **8**(5), 1065, (1997).
[3] J.T. Connor, R.D. Martin, and L.E. Atlas, 'Recurrent neural networks and robust time series prediction', *IEEE Transactions on Neural Networks*, **5**(2), 240–254, (1994).
[4] K.W.Yeung Dit-Yan, 'A locally recurrent neural network model for grammatical interface', in *Infor. processing Proceedings of the International Conference on Neural Information Processing*, (1995).
[5] J.L. Elman, 'Finding structure in time', *Cognitive Science*, **14**(2), 179–211, (1990).
[6] J.L. Elman, 'Distributed representations, simple recurrent networks, and grammatical structure', *Machine Learning*, **7**(2/3), 195–226, (1991).
[7] S. Haykin, *Neural Networks, A Comprehensive Foundation*, Macmillan, New York, NY, 1994.
[8] Yee-Ling LU, Man-Wai MAK, and Wan-Chi SIU, 'Application of a fast real time recurrent learning algorithm to text-to-phone conversion', in *Proceedings of the International Conference of Neural Networks*, volume 5, pp. 2853–2857, (1995).
[9] R.J.Williams and D. Zipser, 'Gradient-based learning algorithms for recurrent networks and their computational complexity', in *Backpropagation: Theory, Architectures and Applications*, eds., Y. Chauvin and D. E. Rumelhart, chapter 13, 433–486, Lawrence Erlbaum Publishers, Hillsdale, N.J., (1995).
[10] P. J. Werbos, 'Backpropagation through time: what it does and how to do it', in *Proceedings of the IEEE*, volume 78, pp. 1550–1560, (1990).
[11] R.J. Williams and D. Zipser, 'Experimental analysis of the real-time recurrent learning algorithm', *Connection Science*, **1**(1), 87–111, (1989).
[12] R.J. Williams and D. Zipser, 'A learning algorithm for continually running fully recurrent neural networks', *Neural Computation*, **1**(2), 270–280, (1989).
[13] William H. Wilson, 'Learning performance of networks like elman's simple recurrent netwroks but having multiple state vectors', *Workshop of the 7th Australian Conference on Neural Networks, Australian National University Canberra*, (1996).
[14] D.M. Himmelblau Y.Cheng, T.W. Karjala, 'Cloosed loop non-liner process identification using internal recurrent nets', *Neural Networks*, **10**(3), 573–586, (1997).

Reciprocity in Open Agent Environments

Enda Ridge[1], **Michael G. Madden**[1], **Colm O' Riordan**[1] and **Gerard Lyons**[1]

Abstract. This paper reports work in progress on validating and extending the research of Sen and Dutta on the evolution of cooperative behaviour in open agent environments [16]. We give a description of the problem domain in which this work was conducted and delineate its merits as a framework for investigating cooperative behaviour. We propose an environment for generalising and extending Sen and Dutta's work to a spatial distribution with neighbourhoods and propose worthwhile experiments in this domain. We also suggest a game theory model of the problem in order to allow more formal analysis.

1 INTRODUCTION AND MOTIVATION

Wooldridge has suggested three broad categories of agent research: theory, architectures and languages [21]. Our work falls within the realm of agent theory in that it aims to contribute to a mathematical formalism for representing and reasoning about agents' tendency to cooperate in open environments.

Autonomous agents are well established as a framework for academic research in artificial intelligence, economics and sociology. Industry is increasingly interested in their potential as demonstrated by the establishment of bodies such as FIPA [4] and AgentCities [1]. The majority of agent research has been conducted in closed, controlled systems where agents can be assumed to be cooperative or to obey some imposed social rules [17].

Certain agent-based systems, such as information assistants and recommenders, will operate in open, heterogeneous, vast and dispersed environments over extended periods. In these circumstances, an agent may not have all the resources or capabilities necessary to achieve its goals and so cooperation may be a useful strategy. Cooperation would involve undertaking another agent's task in the anticipation that the agent would reciprocate. However, the tasks being exchanged will not always be concurrent. For reasons such as an agent's own task scheduling or the nature of the tasks themselves, an agent may be presented with a situation where it must give help with the expectation of receiving help.

Interactions with other agents that have their own goals and selfish interests introduce the possibility of defection and cheating.

There is an inherent conflict between the individual agents, acting out of self-interest, and the performance of the society as a whole. Should we expect open artificial societies to be taken over by parasites rather than mutualists? This question must be addressed if open autonomous software systems are to be a commercial success.

Cooperation is ubiquitous in natural societies and has been studied by biologists [11, 12, 19] and primatologists [6, 18]. Though driven by the 'selfishness' of self-preservation under Darwinian selection, these situations do not have the myriad of laws and punishments that discourage cheating in our modern societies. In many cases, the 'agents' do not even possess the advanced cognitive capabilities needed to estimate the intention and trustworthiness of those with whom they interact. Despite this, cooperation is often a stable strategy for these biological agents in their natural environments.

There is a temptation when designing artificial agent systems to enforce some form of top-down control on the whole society. The need for a central control is an explicit requirement of many traditional multiagent systems. This has been successful within human organisational structures such as corporations and governments. Crucially, these organisations are closed to anonymous participants. Corporations and governments can easily identify their employees and citizens respectively. There are clear rewards (such as promotion) for desirable behaviour and clear punishments (such as imprisonment) for those who break the rules. These hierarchical approaches are impossible to realise in an open artificial environment like the Internet where agents are anonymous, laws and punishments are difficult to enforce and security measures battle to stay ahead of those who would breach them. Furthermore, hierarchies that require interaction through some benevolent intermediary can lead to brittle systems with bottlenecks in communication and computation.

Probst has argued the merits of simplified simulation [13]. If simulations of models are orders of magnitude more complicated than those of formal analysis then the causes underlying the results are difficult to interpret. Results of less complex simulations can extend the limited knowledge of formal theory while still retaining some of its intellectual rigour. By simulating and experimenting with features of open agent environments, we hope to understand the circumstances under which cooperation, rather than selfishness, is an evolutionarily stable strategy [10] for the majority of agents and whether cooperation is of benefit to the society as a whole.

In a real-world setting, agents may represent physically dispersed users. This is true of the V-LAB system [7], where agents interact in a virtual marketplace on behalf of geographically separated hauliers. The modern haulage industry is pan-continental. Hauliers, represented by their agents, could consider sharing and exchanging jobs and resources to reduce costs and improve utilisation of capacity. However, with greater geographical separation and less frequently used routes, there are fewer opportunities for

[1] Department of Information Technology, National University of Ireland, Galway, Ireland. email: enda.ridge@nuigalway.ie; michael.madden@nuigalway.ie; colmor@geminga.nuigalway.ie; gerard.lyons@nuigalway.ie

cooperation between hauliers and consequently less of an incentive to return help. Conversely, neighbouring hauliers, despite being competitors will interact more frequently.

Our research on neighbourhoods models these scenarios. We intend to provide qualitative statements about the effects of various environmental conditions on the stability of cooperation. If positive, these statements can be translated into rules-of-thumb for designers of such open agent systems.

2 RELATED WORK

Cooperative behaviours have been studied in detail in the social sciences and economics. An often cited work is that of Axelrod [2]. The assumptions on which the work was based and the applicability of his claims on the merits of a deterministic tit-for-tat strategy to real world domains have been criticised [3, 8]. Others have looked at the use of morality and incentives to promote cooperative relationships [5].

Sen proposed a probabilistic reciprocity strategy [15] as an alternative strategy that could foster cooperative relationships leading to near optimal societal performance while resisting the selfish incentive of individual agents.

Extensions to this work by Sen and Dutta [16] investigated the stability of the two most sophisticated extremes of this scheme, earned trust reciprocation and collaborative lying in an evolutionary setting. Their claim is that earned trust behaviour dominates collaborative lying across all initial percentages of collaborative liars in a population once the mean number of assigned tasks is greater than about 80 in a population of 100 agents.

There were a number of features of their experiments, identified by the authors [16], which restricted the applicability of their conclusions to true real-world environments. Firstly, the evolutionary selection process sampled from the whole population. The authors identified neighbourhood-based sampling of successful behaviours as a more realistic approach in the evolutionary setting.

Secondly, all agents between tasks could interact to request help with their next task. We propose that the neighbourhood restriction should also apply here. In a large enough environment, it will not be possible for an agent to broadcast its interaction requests to every other agent. Rather each agent will only interact with some subset of the population.

3 PROBABILISTIC RECIPROCITY

3.1 Overview of Probabilistic Reciprocity

We now summarise Sen's probabilistic reciprocity framework as detailed in [14, 15, 16]. Let there be N agents in the multiagent system, each with T tasks assigned to it. The jth task assigned to the ith agent is t_{ij}. Let S_{ki} be the cumulative savings obtained by agent k from agent i over all of their previous interactions. Let W_{ki} be the cumulative work done by agent k for agent i over all of their previous interactions. Then the balance of these exchanges is $B_{ki} = S_{ki} - W_{ki}$. If two agents have never interacted before then the balance of their exchanges is zero. Balances are not symmetrical $B_{ki} \neq B_{ik}$. Sen suggests that the probability that agent k, with an assigned task t_{kl}, will carry out task t_{ij} for agent i is given by:

$$Pr(k,l,i,j) = \frac{1}{1 + exp\dfrac{C_{ij}^{k} - \beta \times C_{avg}^{k} - B_{ki}}{\tau}} \quad (1)$$

β is effectively a measure of an agent's willingness to cooperate and τ determines the steepness of the cooperation curve. C_{avg}^{k} is the average cost of tasks to be performed by agent k. C_{ij}^{k} is the extra cost agent k incurs by accepting agent i's task j.

β and τ are global constants. The values of the other variables come from the dynamics of the system, specifically the agent's history of interactions and the tasks it is randomly assigned. As the exponential term increases, the probability of helping the requesting agent decreases. So a negative balance with the requesting agent can only decrease the probability of helping.

Figure 1 illustrates the probabilistic reciprocity function for two values of β for a fixed average cost and fixed balance with the requesting agent. At a low value of β (bottom) the probability of giving help quickly decreases with the extra cost of that help.

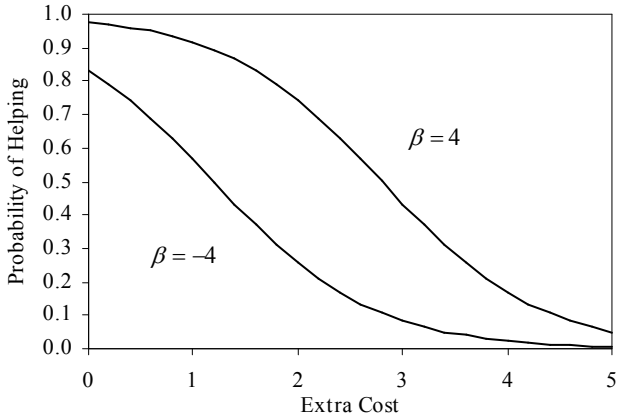


Figure 1. Sample probability functions for accepting a request for help. $\beta = 4$ (top) shows a higher willingness to cooperate and $\beta = -4$ (bottom) shows a lower willingness to cooperate.

3.2 Motivation for using Probabilistic Reciprocity

Reciprocity requires two or more interactions since there is the expectation that any cost incurred will be recouped in future interactions. Thus a reciprocity scheme requires a predictive mechanism. A domain-independent solution to this problem is not known. The alternative is to introduce a stochastic mechanism that incorporates factors that should influence a decision to engage in reciprocative behaviour. In the scheme outlined above, these factors are:

- The difference between the additional cost of a proposed task and some factor times the average cost of all jobs.
- Some measure, either calculated individually or gathered from the group, of the helpfulness of the requesting agent. That is, some measure of the likelihood that the requesting agent will return any help it gets.

There is no claim that a probabilistic scheme produces an optimal solution for evading selfish agents. Rather it may be a useful heu-

ristic for guiding behaviour and agent design in a general open environment.

4 THE PROBLEM DOMAIN

4.1 Overview

Sen's papers were based around research in a package delivery domain [14, 15, 16]. The description of the domain is spread across the three papers that used it. Here, we bring together these descriptions, explicitly stating all assumptions and their implications. We also present a game theoretic description of the interactions in the domain.

4.2 Original Problem Domain

The problem domain consists of one depot to which one or more radial fins are attached (Fig. 2). All fins have the same length. A package delivery task involves bringing a package from the depot to a position on a fin. An agent can undertake more than one task at a time provided all tasks are on the same fin.

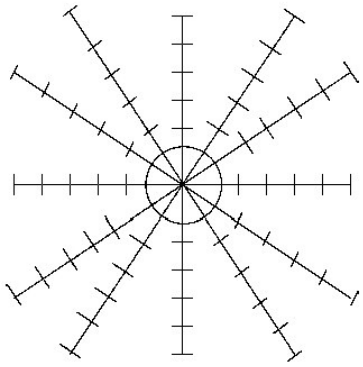


Figure 2. Package Delivery Domain [14, fig. 2]

Each agent is given a randomly generated list of assigned tasks. Agents then proceed to complete their tasks either individually or cooperatively according to certain rules that we explain in the next section.

Consider a package delivery task at a distance d from the depot. The work to complete this task involves travelling a distance d from the depot to the delivery location on the fin and returning back to the depot for the next task. It costs one unit per unit distance travelled per package and one unit per distance travelled (empty-handed) on the return to the depot. The cost of this work for such a task in isolation is $d + d$. However, if two tasks are undertaken at the same time then the savings on the single return journey mitigates the extra cost of bringing the additional package to its destination.

For example, let d_1 be the maximum distance of all tasks the helping agent k is planning to undertake. Let d_2 be the distance of the additional task j that the agent is considering undertaking for agent i . Then the extra cost of the additional task to agent k is,

$$C_{ij}^{k}(d_1, d_2) = \begin{cases} d_2 \text{ if } d_2 \le d_1 \\ d_2 + (d_2 - d_1) \text{ otherwise} \end{cases} \quad (2)$$

For both cases, the additional cost is less than the total cost of both tasks in isolation, $2d_1 + 2d_2$, but the total work completed is the same.

4.3 A Game Theory Description

A game theory description allows us to formally consider interactions from the individual's perspective and make predictions about the possible societal outcomes of these individual actions. A formal specification of any game requires describing four things: the players, the rules, the outcomes and the payoffs [9, p. 219].

4.3.1 The Players

The players are a set of computational agents. Agents differ only in their help-giving behaviour, which falls within one of two categories: reciprocative or selfish. Reciprocative agents give help according to some decision rule, in this case probabilistic reciprocity. Selfish agents never return help. Sen and Dutta showed that the most successful variants of reciprocative and selfish behaviours are what they termed earned trust reciprocative and collaborative lying respectively [16].

In deciding to help another agent, an earned trust reciprocative agent uses the sum of the balances with that agent reported by all *other* agents with which it has a favourable balance.

Collaborative liars spoil the reputations of agents that have helped it in the past and bolster the reputation of agents with which they have a zero or negative balance. A collaborative liar j reveals an untrue balance about agent i as follows [14, p. 190].

$$B'_{ji} = \begin{cases} C \times (-B_{ji}), \text{ when } B_{ji} > 0 \\ \mathcal{P} \qquad \text{otherwise} \end{cases} \quad (3)$$

C is a positive constant determining how much the liar tarnishes a helpful agent's reputation. $\mathcal{P}$ is a large positive constant that the liar reports on fellow liars whose reputation it is bolstering.

We make no claim about the rationality of collaborative lying. It may not make sense strategically for an individual to bolster the reputation of another. The purpose of the collaborative lying strategy is to cause maximum disruption to any cooperative relationships between earned trust agents. In effect, it is the converse of earned trust reciprocation. A player's action set at any time is simply to help or not to help.

4.3.2 Rules

The rules of the game fall into two categories: rules of interaction with other agents and rules of movement within the problem domain.

An agent can only interact for help when it is at the depot. The agents from which it requests help must be at the depot and have an assigned task on the same fin as the requesting agent's assigned task. An agent can only request help with tasks from its assignment set. It cannot agree to do additional tasks and then try to pass them on to other agents. An agent can only request help if the cost it

saves would be greater than the cost incurred by the helping agent. Hence, cooperation improves the society's performance by reducing the cost of completing the work.

When moving in the problem domain, agents cannot cross between fins. All agents move at the same speed, one increment along a fin per simulation time unit. Speed is independent of the number of tasks an agent is currently undertaking. Agents cannot stay still on a fin, they are always either moving out along a fin with a task or returning to the depot.

4.3.3 Outcomes

The outcome of the agent's behaviour strategy is the agent's cost in completing its work.

4.3.4 Preferences

Agents prefer to have the highest performance (lowest cost) possible.

We can represent a single interaction between two agents in normal form as follows. The agents are Agent 1 with assigned task k and Agent 2 with assigned task l.

	Help	Not Help
Help	$t_{1k} - C_{2l}^{1}, t_{2l} - C_{1k}^{2}$	$-t_{1k} - C_{2l}^{1}, t_{2l}$
Not Help	$t_{1k}, -t_{2l} - C_{1k}^{2}$	$-t_{1k}, -t_{2l}$

A reciprocative agent will play a mixed strategy with a probability calculated using the reciprocity scheme. A selfish agent with always play the pure strategy *Not Help*. Of course, a single interaction does not give us the full picture, since a loss to a player in one interaction may be recouped in the subsequent one. We intend to extend our analysis to an evolutionary game theory framework [10] where probabilistic reciprocity is the incumbent strategy and we want to determine its resistance to the mutant selfishness.

4.4 How the game is played

Each game consists of one or more evaluation periods. An evaluation period is a duration for which an agent's performance is measured. At the beginning of the first evaluation period, all agents are assigned their initial behaviours. This allows us to experiment with varying initial proportions of selfish agents in the population.

Each agent is then given a randomly generated list of assigned tasks that must be completed in the given order. For each such task assignment set, the total number of tasks is the same and the total delivery distance of the tasks is the same. This guarantees that the total work assigned to each agent is the same.

An evaluation period ends when all agents have completed all their tasks. An agent's performance is inversely proportional to its total costs incurred in completing all of its work. Because there is a cost saving in delivering packages concurrently on the same fin, agents that cooperate will perform better than those that act individually. The question of interest is whether cooperative agents can complete their tasks at a lower cost than selfish agents.

Behaviours for the next period are chosen according to a tournament selection type scheme that is weighted by performance. The pool from which the selection is made includes all agents in the system. Once new behaviours have been assigned, each agent is given a new task assignment set and the next evaluation period begins.

The game ends once a certain number of evaluation periods have occurred or all agents have adopted the same behaviours and so the population is homogeneous.

4.5 Assumptions in the package delivery domain

The following assumptions about the package delivery problem domain allow us to concentrate on the fundamental variables affecting our experiments.

- **There is no noise.** Requests for help always reach their destination agent and are unambiguous.
- **Agents cannot fail in a task.** If an agent agrees to carry out a task then that task will be carried out successfully. This avoids the complications of replanning when failures arise.
- **Tasks do not require specialisation.** Any agent can complete any task. This avoids giving an advantage to more skilled agents.
- **Helpful agents are honest.** When requesting balances from other agents, we can believe those balances reported by agents with whom we have a positive balance. Sen has suggested that this naïve assessment of trustworthiness could be replaced by a more sophisticated scheme such as reinforcement learning [14]. The critical point is that there must be some feature to allow an assessment of trustworthiness.

4.6 Advantages of the package delivery domain

The package delivery domain is a convenient abstraction:

- It can capture tasks that cannot be undertaken concurrently by assigning them to different fins.
- It captures the costs saved in completing work concurrently.
- Tasks require varying amounts of time to complete, depending on their distance from the depot.
- Tasks are indivisible. Agents cannot do part of a task and delegate the remainder to others.
- The cost of a task is easily calculated. There are no multiple objectives that require advanced techniques such as constraint programming for an optimal solution.
- Tasks are one dimensional. They have only one characteristic — distance along a fin, making it is easy to compare between them.
- It is easy to evaluate success at achieving tasks using the inverse of the total cost incurred.

Although such characteristics could equally well be incorporated in, for example, an information processing metaphor, we continue with Sen's metaphor for easy comparison between our work and his group's.

4.7 Extensions to the Game

Sen and Dutta's original game [16] placed no restriction on the sampling of behaviours at the end of an evaluation period. Agents effectively inhabited a 'soup' where every agent could sample from the total pool of agents in the environment. We believe that this is unrealistic in dispersed environments, for example over

large networks where delays across a network limit an agent's communications.

If the tournament selection scheme is to represent what would happen in such circumstances then the pool it draws from should be some subset of the total population. We define this subset based on neighbourhoods. .

For the concept of a neighbourhood to have meaning, we first require some form of spatial distribution. Consider a square patch of nodes where each node is connected only to those nodes immediately adjacent to it on the horizontal and vertical axes. The edges of the patch are wrapped — an agent can move directly between the top and bottom edges and between the left and right edges. Hence, the agents inhabit a toroidal surface. This is the simplest 2-dimensional connectivity to model.

There are several types of neighbourhood, many of which originate in the study of cellular automata. In our experiments we use the Moore neighbourhood. The Moore neighbourhood consists of the nodes on the horizontal and vertical axes and the nodes between them. Fig. 3 illustrates a Moore neighbourhood with range 2.

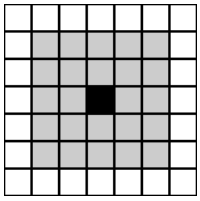


Figure 3. Moore neighbourhood with range 2

The neighbourhood serves to limit both the interactions of the agent while executing its tasks and the sampling of behaviours at the end of an evaluation period. The range then might be a measure of the agent's ability to probe the network for interaction possibilities. In our initial experiments, agents are static on the lattice for the duration of an evaluation period. We leave movement and its associated complexities to later work.

5 PROPOSED EXPERIMENTS

The objective of these experiments is to investigate the effect of neighbourhood sampling on Sen and Dutta's conclusions about the robustness of probabilistic reciprocity in the face of collaborative lying [16].

Their experiments are a specific case of our framework where the neighbourhood range for agents has been extended to cover the whole lattice.

Our experiments will begin with a random distribution of agents on the lattice. Firstly, we plan to test the effect of neighbourhood range on the ability of earned trust reciprocative agents to resist collaborative liars. We expect that increasing an agent's interaction possibilities through a larger neighbourhood range should improve its resistance to exploitation by permitting a more comprehensive reputation scheme. Restricting behaviour sampling should increase the number of evaluation periods required before the population reaches homogeneity.

However, in situations where agents are not dispersed randomly on the lattice there may be disadvantages to a large neighbourhood range for interactions. In particular, consider a cooperative agent in a niche of other cooperative agents. It is foreseeable that such niches would emerge when agents have the ability to move. If such an agent's range extended beyond the boundaries of the niche, the agent would expose itself to harmful interactions. We plan to investigate the relationship between niches and neighbourhood range by biasing the distribution of behaviours on the lattice.

Once we have established the desirable characteristics of niches that resist exploitation, we can allow agents to move on the lattice and search out such niches. Equally, selfish agents, aware of any weaknesses will prey on the niches.

Sen and Dutta's conclusions were valid where:

1. The composition of the group is stable for some number of tasks or some number of interactions.
2. There is a sufficient number of symmetrical cooperation possibilities.
3. Agents have the same evaluation metric. There is no disagreement between agents on the cost of a given task.

All of these conditions are dependent on the nature of the tasks the agents need to complete. The first condition should hold where all tasks must be completed within a time window. In this case the agents must come together for the duration of the window. A rogue agent cannot enter the environment, prey on helpful agents for a small number of tasks and then leave the environment before it is identified by the reputation system.

The second condition depends on there being a sufficient number of tasks and that those tasks are relatively similar. Hence tasks are achievable within most agents' skill sets and their relative costs are easily computable.

Thirdly, large differences in agents' assessments of the cost of tasks should prevent cooperation.

Furthermore, the conclusions we reach will apply to environments with similar characteristics to those modelled by the lattice. That is, environments in which resources, and hence agents, are dispersed among a large number of nodes. Only a subset of the total nodes can be visible to an agent at any time.

6 CONCLUSIONS AND FUTURE WORK

This paper has brought together a comprehensive description of the package delivery domain as used by Sen et al [14, 15, 16]. We have explicitly stated the assumptions in using the domain and the consequent applicability of results obtained from it. We hope that this will encourage a synthesis of the various endeavours to understand cooperation in artificial societies and a framework within which to compare results from various researchers.

A formal analysis is an important grounding for any complex simulation. We present an initial game theory viewpoint of the problem for a single interaction. The intention is to make the domain more amenable to analysis with the considerable body of knowledge and techniques available from evolutionary game theory [20].

We propose an implementation of Sen and Dutta's suggestion of neighbourhood based sampling and identify the parameters and tests of interest within such an implementation.

This research is in its early stages. Our experiments must reproduce Sen and Dutta's results in the extreme case of a 'global' neighbourhood. We expect that once we have some qualitative and analytical results for static agents we can begin to investigate the more complicated scenario of mobile agents. '

ACKNOWLEDGEMENTS

The support of the Enterprise Ireland Informatics Research Initiative is gratefully acknowledged.

REFERENCES

[1] Agentcities Web [HTML]. Agentcities Task Force; 2002 [cited 2003 18th June]; Available from: URL:http://www.agentcities.org/.
[2] R. M. Axelrod, The Evolution of Cooperation, Basic Books, New York, 1984.
[3] K. Binmore, Playing Fair: Game Theory and the Social Contract, MIT Press, Cambridge, MA, 1994.
[4] J. Dale. Welcome to the Foundation for Intelligent Physical Agents [HTML]. The Foundation for Intelligent Physical Agents; 2002 [cited 2002 20/11/2002]; Available from: URL:http://www.fipa.org/.
[5] A. Glass, B. Grosz, 'Socially conscious decision-making', in Proceedings of the Fourth International Conference on Autonomous Agents, ACM, New York, 217-224, (2000).
[6] K. R. L. Hall, I. DeVore, 'Baboon Social Behavior', in Primate Behavior: Field Studies of Monkeys and Apes, Holt, Rhinehart and Winston, New York, 53-110, (1965).
[7] G. J. Lyons, M. Madden, F. Smith, et al, 'Virtual Logistics Multi-Agent Broker', in Enterprise Ireland Informatics Research Initiative 2003, Enterprise Ireland, Dublin, Ireland, 24, (2003).
[8] J. Martinez-Coll, J. Hirshleifer, 'The limits of reciprocity', Rationality and Society, 3, 35-64, (1991).
[9] A. Mas-Colell, M. D. Whinston, J. R. Green, Microeconomic Theory, Oxford University Press, Inc., 198 Madison Avenue, New York, New York 10016, U.S.A., 1995.
[10] J. Maynard-Smith, Evolution and the theory of games, Cambridge University Press, Cambridge, 1982.
[11] M. Milinski, 'TIT FOR TAT in Sticklebacks and the Evolution of Cooperation', Nature, 23, 434-435, (1987).
[12] M. A. Nowak, R. M. May, K. Sigmund. The Arithmetics of Mutual Help. Scientific American 1995 June;272(6):76-81..
[13] D. Probst, review of Evolutionary Game Theory by Weibull J, The Journal of Artificial Societies and Social Simulation, 2 (1), (1999).
[14] S. Sen, 'Believing others: Pros and cons', Artificial Intelligence, 142 (2), 179-203, (2002).
[15] S. Sen, 'Reciprocity: a foundational principle for promoting cooperative behavior among self-interested agents', in Proceedings of the Second International Conference on Multiagent Systems, AAAI Press, Menlo Park, CA, 322-329, (1996).
[16] S. Sen, P. S. Dutta, 'The Evolution and Stability of Cooperative Traits', in Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, ACM Press, New York, 1114-1120, (2002).
[17] Y. Shoham, M. Tennenholtz, 'On social laws for artificial agent societies: off-line design', Artificial Intelligence, 73 (1-2), 231-252, (1995).
[18] T. Struhsaker, 'Social structure among vervet monkeys (Cercopithecus aethiops)', Behavior, 29, 83-121, (1967).
[19] R. L. Trivers, 'The Evolution of Reciprocal Altruism', The Quarterly Review of Biology, 46 (1), 35-57, (1971).
[20] J. W. Weibull, Evolutionary Game Theory, The MIT Press, Cambridge, MA, 1995.
[21] M. Wooldridge, N. Jennings, 'Intelligent Agents: Theory and Practice', Knowledge Engineering Review, 10 (2), 115-152, (1995).

Complex Constraint Domains [1]

Georg Ringwelski[2]

Abstract. A basic means to integrate Constraint Programming (CP) and Object-Oriented software design is presented in this paper. Using Complex Constraint Domains, Constraint Satisfaction or Optimization Problems can be formulated over arbitrary objects instead of variables of a predefined domain. This allows the direct application of software engineering methods in constraint-based applications. The benefit of this is twofold: First, during the Software-Process the constraints related work can be done independently and second the domain specific constraint model is easier to understand for end-users such that they are more likely to provide adjuvant input.

1 Motivation

Constraint Programming has proven to be a very effective means to solve problems of manifold structures. Constraint Satisfaction or Optimization Problems (CSP/COP) are often used very successfully to model and solve real-world or benchmark problems with CP. The basic idea is to formulate problems in terms of variables, their respective domains and constraints and use a reusable constraint solver to find solutions to the thus modeled problems. Much research and software development has been made to successfully improve constraint solvers wrt. their capabilities and efficiency in the last two decades. This knowledge can be applied directly to arbitrary problems without much further implementation effort as it is provided with the used constraint solver. Other constraint processing frameworks, like they are discussed in Operations Research for example, often require a specialized implementation and tuning of the algorithms before they can be successfully applied. But still CP is in many cases not only a good platform for Rapid Prototyping, but can sometimes outperform specialized approaches and is thus an adequate computation model for arbitrary applications.

However, CP has not yet become a standard software development framework as for what it would have the potential. The reason for this is to a wide extent, that it requires a lot of expertise to apply the scheme. One main focus of constraints related research today is thus to make the technique more applicable and easier to use. In the state-of-the-art of CP there are many hurdles to be taken prior to use constraint-based methods in applications. Some examples for this are:

- Understand the CP execution model, and thus the concepts of variable domains, constraint-semantics, propagation and search.
- Become familiar with a programming environment to apply this knowledge. There are many differing libraries (e.g. [13, 15]) and languages [5, 4] for CP available, most of which require quite some effort to get started.
- Decide which solver suits best the specific needs of the current problem wrt. functional capabilities, supported modeling techniques, costs, etc.
- Define a CSP/COP-model that can be solved best within the provided software settings, such as time and memory consumption, privacy issues, etc.

Such problems become even more severe and restricting if considered from the Software Engineering (SWE) perspective. As constraint solvers are rather restrictive wrt. their processable input, software systems have to be built somehow around the chosen CSP/COP model. This makes it necessary for all participating programmers to understand the CP part of the system to be built and will thus force them to take all of the mentioned hurdles. I have observed in many industrial projects, that the starting point for constraint-based applications is a model that is designed according to the capabilities of the used solver. This severely confines the application of other established software development methods. The thus resulting software-architecture is in many cases rather bad wrt. general SWE objectives such as extendability, reusability or maintainability. From the SWE perspective it is thus necessary to encapsulate the constraints-related part of a software system as well as possible. Then standard software development methods, which focus on software with the mentioned general properties can be applied. Constraints, variables and the solver will be exchangeable components with an interface that can also be used by non-CP-experts. Thus CP can be integrated in software systems in a way which is not determined by the used constraint solver, but by general design objectives. This setting will make it unnecessary for most of the participating programmers, to understand the CP-part of the system and thus force them to take all of the mentioned hurdles.

Making the general assumption, that the Software-Process[3] should be rather oriented to the application domain than to

[1] This work was funded by the Embark initiative of the Irish Research Council for Science, Engineering and Technology under grant PD/2002/21. The Cork Constraint Computation Center is supported from Science Foundation Ireland under Grant 00/PI.1/C075.

[2] 4C, UCC, Cork, Ireland. eMail: g.ringwelski@4c.ucc.ie

[3] The notion of Software-Processes is widely used in SWE to refer to the process of software development. This may include several different tasks, which are specified more precisely by the chosen development method. Software Engineering is concerned with the analysis, formalization and improvement of this process for many years. For an overview over some methods to describe Software-Processes, refer to [17].

one of its software components, we propose a means to integrate CP in arbitrary software-models. This is to allow standard software modeling techniques, such as Object-Oriented Analysis and Design, to be used without any further restrictions in constraint-based software. In this setting, a constraints expert is still needed to implement the constraints and variables, but her work can be encapsulated in one SW-component and interfaces to other components will be designed on a more abstract level that can be understood by software developers without CP expertise.

The basic idea is to design constraints according to the application domain instead of just using what is provided by a solver. The arguments of such constraints will then be models of arbitrary entities of the application domain instead of variables of a predefined constraint domain. Provided such constraints, it will be easier for programmers to perform the first of the mentioned hurdles. In addition, for most programmers in a real-world constraint-based application this may be enough knowledge to effectively participate in the respective Software-Process. Only the developers who are to implement the constraints have to know the underlying solver and its capabilities.

Using such application-oriented constraints can not only be very beneficial in the Software-Process of constraint-based applications, but also in the resulting system. It has often been discussed in the constraints community, that users are not able to provide a sufficient amount of domain knowledge to enable systems to find the desired solutions. Consequently there is current research on knowledge acquisition [14], which is to provide a way to get just that kind of input from the user, which is needed to solve her problem with the means of CP. One main aspect of this work is to "ask the right questions". This requires a bi-directional translation of what the user understands and what the solver can handle. This translation can be made much easier with the use of Complex Constraint Domains. Using this technique, constraints can be designed according to the user's knowledge, such that on the one hand the user will understand, what kind of input is requested and on the other hand the system is able to process this input adequately. In this setting, the work of translation is moved from the application to the implementation of the constraints. Once the semantics of the application-oriented constraints is implemented by an algorithm that makes them processable by a solver, they can be used for arbitrary user interaction. I think, this will make user interaction more effective and thus improve the ergonomics of constraint-based software.

The proposed approach is language and (of course) solver independent. The paper provides a general method to use constraints in complex domains. There are some standard requirements to the implementation platform, which are met by all modern programming languages. The constraint solver that is to be used has to provide a feature, which is currently not provided by all systems. However, it is supported in most of today's systems and is a widely accepted basis for CP. This is the possibility to implement constraints by means of propagators [16] or domain reduction functions. These are procedures that are executed whenever a constraint is triggered, i.e. when the solver detected, that the constraint might infer new knowledge in the current state.

2 Traditional Constraint Domains

Constraint Satisfaction Problems (CSP) are defined by a set of variables and a conjunction of constraints over these variables. Constraint Optimization Problems (COP) use an additional objective function which defines a metric on all solutions of the CSP. The declarative semantics of every constraint is usually given by some logic expression, which states restricting properties of the constraint's variables. In CP, this restriction is projected to the variables' respective domain. Domains are given as sets of allowed values for every variable and constraints can eliminate values from these sets, that are proven not to satisfy the constraint. Operationally, constraints can be given either explicitly or implicitly:

1. The explicit definition of a constraint is given by a relation over the Cartesian product of its variables' domains. Thus it explicitly enumerates all allowed combinations of values that will model its declarative semantics: Let $v_1, ..., v_n$ be variables with respective domains $D_1, ..., D_n$, then a constraint c over $v_a, v_b, ...$ is defined as $c \subseteq D_a \times D_b \times ....$
2. The implicit definition of a constraint is normally given by propagators[16]. These are procedures, that may restrict variable domains according to the properties of other variables. In this setting, c defines at least one propagator, which implies a domain reduction function[1] in $D_1 \times ... \times D_n$. The domain reduction has to be done in accordance to the intended semantics of the constraint.

It can easily be seen, that in both cases the domain of every variable is derived from a superset, or universe. This set, together with its typical operations and relations is referred to the **constraint domain** [9]. Or from the other point of view: A constraint domain is a single-sorted algebra with predicates (constraints) and the variable domains are subsets of its carrier. In all constraint solvers I am aware of, variable domains are subsets of some basic datatype that is supported by the host language. For example, the basic data type `integer` or `long` is used as the superset of all variable domains in constraint solvers for finite domains [2, 5, 11].

The constraint programmer can be expected to know the semantics of the basic operations and relations of this underlying datatype, since it is part of the programming language she is using. This knowledge is used to make the constraints, which operate on constraint variables instead of values, understandable for her. For example, every programmer will understand the semantics of the constraint $X + Y < Z$ intuitively from her knowledge of the operation $+$ and the relation $<$ in $\mathbb{Z}$, although the used operations are not the ones, she knows and have other operational and denotational semantics. The idea of declarative programming with constraints is thus to overload these well known operations and implement their domain-counterpart in constraints.

Technically a constructor is used to create constraint variables out of a description of their initial domain (e.g. `in` or `::` in many CLP systems). Over the thus created objects (i.e. constraint variables) the basic operations and relations, i.e. the constraint, can be re-defined containing the intended semantics. This has to be done explicitly for every operation and relation and can be a very difficult task having many pitfalls [3]. There are two requirements to the re-definition of operations on constraint variables that have always to be met:

1. Whenever the variable's domains contains exactly one value, the constraints (i.e. relations and operations) have the same semantics as their underlying operations.
2. The domain reductions a constraint makes are always sound wrt. the declarative semantics. This means, that no values are pruned that cannot be proven not to be part of any solution.

Furthermore, constraints may cause the system to fail, if they detect, that no solution can be found for the specified problem. The second requirement implies the first one in the case of instantiated variables. In most cases this fact leads to an implementation of constraints with their underlying relation. This can also make sure, that the specification of both operations will match in this special case. This formal topic can be discussed in further work, for now the idea is demonstrated in the equation that specifies the `lt`-constraint in Example 1. In that example I also illustrate the idea of the construction of a constraint system upon a basic datatype. It shows part of a specification of a finite domain constraint system that is built upon some given integer type. I used the syntax of Algebraic Specification, which the reader can find a detailed description for in [6].

Example 1 *Let a basic data type for integers with the successor operation and a less-than relation* `lt`*, which is modeled as boolean function be given:*

```
INT =
 sorts: int, bool
 opns: zero: -> int
   true, false : -> bool
   succ: int -> int
   lt: int int -> bool
 eqns: lt(X,succ(X)) = true
   lt(X,zero) = false
   lt(X,succ(Y)) = lt(X,Y)
```

One can build a constraint system with the less-than constraint over finite domain variables upon this datatype by using `INT`*, and a new type for variables, that contains as domain information the upper and lower bounds of the allowed interval of integers.*

```
FD = INT +
 sorts: fd_var
 opns: mkVar: int int -> fd_var // constructor
   lt: fd_var fd_var -> bool // constraint
 eqns: lt(mkVar(X1,Y1),mkVar(X2,Y2)) = lt(X1,Y2)
```

The equation specifying the semantics of the `lt`*-constraint says, that the constraint holds, whenever the lower bound of the first variable is less than the upper bound of the second. This implies the semantics of* `lt` *on integers (as can be seen in the right hand side of the equation), if the variables are instantiated and is thus consistent with the equations of* `INT`*. The pruning, that the* `lt`*-constraint can perform is not represented here.*

In general, the basic datatype has to provide Booleans in order to allow the modeling of relations as boolean functions. Furthermore some structured set of values should be provided, that is to be used as superset for all variable domains. Given these basic prerequisites, a constraint system can be defined upon the basic datatype.

Proposition 1 *A constraint system can be constructed by:*

1. *A new type for constraint variables*
2. *A constructor that creates a constraint variable from some terms uniquely defining a set of values, which is to become the variable's domain*
3. *Boolean constraint-operations that use variables and values as input.*

Any constraint system, given as a parameterised datatype with the proposed properties, can be generated by actualisation with some basic datatype. The basic data type provides all the necessary functionality, to specify the behaviour of constraints in the constructed system. The general module composition technique of actualisation is described in detail in [7], where not only the forward directed construction of the syntax is described, but also an initial (in the category of all algebras with the specified syntax) algebra is constructed from the semantics of the basis and the parameterized data type. To visualize the basic idea I show the scenario of an actualization in Figure 1. In the diagram the required specification-morphisms, i.e. structure-preserving mappings of the symbols of the specification are denoted by arrows.

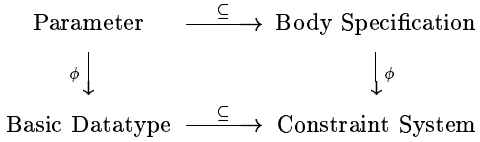


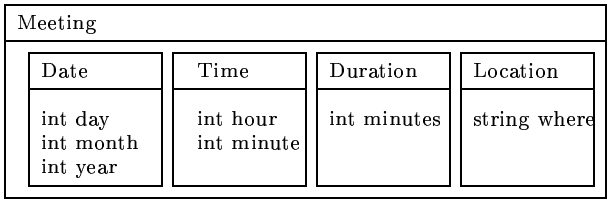
Figure 1. Composition of a constraint system out of a basic datatype and a parameterized specification. ϕ denotes a specification morphism that identifies names. The other specification morphisms are simple set-wise inclusions without any renamings. The body specification is to contain a constructor for constraint variables and at least one constraint.

3 Complex Constraint Domains

For benchmarking or illustration purposes, often mathematical problems or problems that use very compactly describable data are used. For most of such problems (many of them are collected at CSPlib[10]) no modeling of values is necessary, because the searched solutions consist of basic values such as numbers. The modeling effort, which is extensively discussed in the community for these problems concentrates on the identification of variables and constraints, such that some given constraint solver can handle the problem. The art or craft of modeling with constraints is often considered as to find variables, their respective domains and constraints, such that a problem can be solved fastest. This paper addresses a modeling issue that is often only marginally handled: The modeling of values. This task of modeling is not often considered in CP research, because it is obvious for the most of the mentioned problems or it is determined by the constraint solver that is to be used. However, as I want to integrate CP into standard Object-Oriented Software-Processes the constraint domains will in most cases have a structure that does not allow the use of standard types of variable domains and constraints.

The constraints in this framework will normally restrict properties of objects that model real-world objects and will thus have a more complex internal structure.

Example 2 *During the Object-Oriented Analysis for a meeting scheduling tool, meetings are likely to be modeled as objects. As I discussed in [12] the following could be one reasonable class for them:*



Constraints over such objects could be for example:

1. *Meeting A must be before meeting B*
2. *No meetings overlap*
3. *Meetings should preferably be before lunch*
4. *Meeting A must not exceed 20 minutes*

Just as in traditional CP, constraints can be hard (1,2,4) or soft (3), unary (4), binary (1) or global (2) and they can be user-defined (1,3,4) or inherent to the application domain(2).

As can be seen in the above example, in the Object-Oriented modeling with constraints it is adequate to use constraints over entities with a complex internal structure. The constraints restrict the properties of these structures wrt. one or more of their features. The superset of all these structures makes up the constraint domain of the respective application. The handling of such structured constraint domains is the actual contribution of this paper. In traditional approaches, only flat datatypes are considered as constraint domains. I will show in this paper, that also complex constraint domains can be handled by traditional, re-usable constraint solvers, if the implementation of the complex structures provides some required functionality.

Following the CP approach, the variable domains are derived from the constraint domain. Thus, I assume that the constraint domain contains all possible entities that can be described by the available basic datatypes and the provided constraint systems. As I have stated earlier, a constructor has to exist for every type of constraint variable.

Definition 1 *Let be given a set of basic datatypes D with respective sets of sorts $sorts(D)$ and a set of constraint systems with the set constructors(CS). The set $Dom_{D,CS}$ of all* **complex constraint domains** *over D and CS is the closure of $Dom_{D,CS}$ wrt. the following implication: $\tau_1, ..., \tau_m \in (sorts(D) \cup Dom_{D,CS}) \wedge con : \tau_1 ... \tau_m \rightarrow \sigma \in constructors(CS) \Rightarrow \sigma \in Dom_{D,CS}$*

This possibly infinite set contains all types of variable domains that can be created by the constructors. In difference to the traditional approach, where all values have the same type, we cannot consider variable domains to be subsets of the constraint domain, but have to use a different universe for every type. Consequently, in our approach every object can only be instantiated with a restricted set of the possible values, namely those that have the correct type.

Definition 2 *Every* **complex constraint variable** *$x \in X$ in a complex constraint domain $Dom_{D,CS}$ is associated to its domain type by the function $dom_t : X \rightarrow Dom_{D,CS}$.*

The actual domain of a complex constraint variable is a set, which is in most cases a Cartesian product. This set has to have the same structure as the variable's domain type. As the domain type is associated by a total mapping to every variable, it will always exist and be computable. To refer to the domain values themselves, which is performed by the constraints, the different components of the variable domain must be accessible. This can be done by the established concept of selectors. Selectors are functions that map complex structures to their internal components and are a built-in feature of most programming languages.

Proposition 2 *Given a set of constraint systems CS (as described in Prop. 1), that define the set $Dom_{D,CS}$ of complex constraint domains. A complex constraint system can be built upon these systems by the definition of a constraint system with the constructor $con : \tau_1 ... \tau_n \rightarrow \sigma$, if $\tau_1, ..., \tau_n \in Dom_{D,CS}$ and the additional requirement that*

4. *$\forall 1 \leq i \leq n : sel_i : \sigma \rightarrow \tau_i$ is defined*

With these prerequisites, it is not necessary to actually handle the domains of complex variables. There is no need to construct the variable domains as Cartesian products explicitly, because the internal sets can be accessed whenever values are required. Thus in difference to traditional constraint variables the domains of complex constraint variables are not represented in the respective application. The constraints, which access the variable domains, use selectors to access a certain attributes of the complex variables and perform domain operations on the thus returned simple variable domains. The domains of the basic constraint variables are stored in a traditional way in the used constraint solver. This idea will become more clear, when I show, how constraints over complex variables can be implemented in the next Section. But first, I want to illustrate, how complex constraint domains can be constructed in the Algebraic Specification framework.

Example 3 *The complex constraint domain of meetings (see Example 2) can be constructed from the constraint systems* `DATE, TIME, DUR` *and* `LOC`.

```
MEETING = DATE + TIME + DUR + LOC +
 sorts: meeting
 opns: //constructor
   mkMeeting: date time dur loc -> meeting
  // selectors
  getDate: meeting -> date
  getTime: meeting -> time
  ...
   //constraints
  before: meeting meeting -> boolean
  shorter_than: meeting int -> boolean
  before_lunch: meeting -> boolean
```

The global constraint (2) from the list in Example 2 cannot be implemented in this framework, because there is no datatype for lists or other accumulating structures available. The other constraints (1,3 and 4) can be implemented with some algorithm, that interprets the respective functions. The first three

parameter systems (`DATE, TIME, DUR`*) are constructed from* `FD` *(Example 1) and thus the basic type* `INT`*, I illustrate this in one case:*

```
TIME = FD +
 sorts: time
 opns: mkTime: fd_var fd_var -> time //constructor
  getHour: time -> fd_var //selectors
  getMin: time -> fd_var
  before_time: time time -> boolean //constraint
```

The constraint system `LOC` *is also required for* `MEETING`*, it is constructed from standard strings in a similar way as the system* `FD`*.*

4 Constraint Solving in Complex Domains

Constraints over complex variables have, just as regular constraints, a declarative semantics, that restricts the allowed values for their variables. As the domains of complex variables are not explicitly constructed from the underlying sets, constraints over these variables can not be explicitly defined (as in 1. on page 2). Thus constraints over complex variables must be implemented with propagators (as described in 2. on page 2), that perform the intended domain restrictions. These propagators are to use the selectors of complex constraint variables to access the underlying data. Using this projection in every execution of the propagators, we omit the handling of complex constraint domains and can use the basic constraint systems as provided in many constraint solvers. Regular objects or other complex structures can then use constraint variables as attributes and the constraints will refer to the appropriate sub-structures. To illustrate the construction of such complex constraint variables I show in Example 4, how basic finite domain constraint variables can be used in SICStus Prolog to make up the complex constraint variables for meetings as introduced in previous examples.

Example 4 *As Prolog does not provide constructor functions, we use term construction for this purpose. The following predicate will produce a term representing one meeting, where all the attributes are constraint variables with initial domains. The predicate is implemented in SICStus Prolog and makes use of the provided library for constraint processing in finite domains, which defines the constructor* `in` *for constraint variables.*

```
mkMeeting(((Day,Month,Year),(Hour,Min),Dur,Loc)):-
   Day in 1..31,
   Month in 1..12,
   Year in 2000..3000,
   Hour in 0..23,
   Min in 0..59,
   Dur in 1..1440, % max one day
   all_places(Locs),  % get domain from database
   dom(Loc,Locs). % link to variable as domain
```

Such complex structures can be easily fitted in any Object-Oriented model and still do not restrict the capabilities of constraint modeling. As mentioned before, constraints over such complex structures use selectors to get a reference to the required sub-structure and can then make domain restrictions on these attributes. In Example 5, we show, how the `before`-constraint, mentioned in previous examples can be implemented this way. I used the language CHR [8] to implement the constraints, which is integrated in the SICStus Prolog system. However, any implementation language and host system could be used, if it provides a means to implement constraints by propagators.

Example 5 *The* `before/2` *constraint enforces the meeting represented by its first argument to be scheduled, such that it will be finished before the meeting in its second argument. As Prolog does not provide selector functions, pattern-matching is used in the heads of the rules instead.*

```
before((D1,T1,Dur1,_),(D2,T2,_,_))
   <=> ground([D1,D2]),D1=D2  % if same date
   | before_time(T1,Dur1,T2). % time must be earlier
before((D1,_,_,_),(D2,_,_,_))
   <=> ground([D1,D2])  % otherwise,
   | before_day(D1,D2).   % date must be earlier
```

This constraint is reduced (by simplification rules) to constraints over the internal structures representing the date and the time. The constraint `before_time(T1,D1,T2)` *enforces, that* `T2` *is later than* `D1` *minutes after* `T1`*. It is implemented with SICStus' builtin-constraint* `#<`*:*

```
before_time((H1,M1),D1,(H2,M2))
   <=> ((H1 * 60) + M1) + D1 #< ((H2 * 60) + M2).
```

5 Conclusion

Complex Constraint Domains are motivated by the fact, that the restrictive capabilities of today's constraint solvers often have negative impact on the general software properties of constraint-based systems. This drawback can be made up with the use of constraints over complex constraint variables. These variables are models of real-world entities as opposed to traditional variables which are models of a predefined flat data structure.

The paper proposes a means to use Complex Constraint Domains in the Software Process of constraint based applications. This allows for standard software modeling methods, such as OOA or OOD, to be applied without restricting the capabilities of Constraint Programming. The software systems produced this way are more likely to meet general software requirements such as reusability or maintainability while preserving the powerful problem solving techniques of CP. Furthermore, this allows for better software ergonomics since the input format can be designed in accordance to the user's knowledge instead of the abstract mathematical model that can be handled by the used constraint solver.

REFERENCES

[1] Krzysztof R. Apt, 'The essence of constraint propagation', *Theoretical Computer Science*, **221**(1-2), 179–210, (1998).

[2] Philippe Codognet and Daniel Diaz, 'Compiling constraints in clp(fd)', *Journal of Logic Programming*, (1996).

[3] Alain Colmerauer. Multiplication constraint in several approximation spaces. Invited talk at CP01/ ILCP01 conference, Paphos, Cyprus, November 2001. `www.lim.univ-mrs.fr/~colmer/Transparents/Paphos01/paphose.ps`.

[4] The Mozart Consortium. `http://www.mozart-oz.org`.

[5] M. Dincbas, P. van Hentenryck, H. Simonis, A. Aggoun, T. Graf, and F. Berthier, 'The constraint logic programming language chip', in *Proceedings of the International Conference on Fifth Generation Computer Systems (FGCS88)*, (1988).

[6] Hartmut Ehrig and Bernd Mahr, *Fundamentals of Algebraic Specification 1*, Springer-Verlag Berlin Heidelberg, 1985.

[7] Hartmut Ehrig and Bernd Mahr, *Fundamentals of Algebraic Specification 2*, Springer-Verlag Berlin Heidelberg, 1990.

[8] Thom Frühwirth, 'Theory and practice of constraint handling rules', *Special Issue on Constraint Logic Programming (P. Stuckey and K. Marriot, Eds.), Journal of Logic Programming*, **37**(1-3), 95–138, (1998).

[9] Thom Frühwirth and Slim Abdenadher, *Constraint-Programmierung*, Springer, Berlin Heidelberg, 1997. in german.

[10] Ian Gent and Toby Walsh. CSPLib, a problem library for constraints, `http://www-users.cs.york.ac.uk/~tw/csplib/`.

[11] J.-F. Puget, 'A c++ implementation of clp', Technical report, ILOG, (1994). In Ilog Solver Collected papers.

[12] Georg Ringwelski, *Asynchrones Constraintlösen - Ein generisches Ausführungsmodell zur inkrementellen und adaptiven Constrsaintverarbeitung*, Ph.D. dissertation, Technical University Berlin, 2003. `edocs.tu-berlin.de/diss/2003/ringwelski_georg.htm`.

[13] ILOG S.A. `http://www.ilog.fr`.

[14] Eugene C. Freuder. Sarah O'Connell, Barry O'Sullivan, 'Query generation for interactive constraint acquisition', in *Proceedings of the 4th International Conference on Recent Advances in Soft Computing (RASC-2002)*, pp. 295–300, (2002).

[15] Hans Schlenker and Georg Ringwelski, 'Pooc - a platform for object-oriemnted constraint programming', in *Recent Advances in Constraints*, ed., B. O'Sullivan, 159–170, Springer LNAI 2627, (2003).

[16] Christian Schulte and Mats Carlson. Finite domain constraint programming systems. Tutorial at CP02 conference, Ithaca, USA, September 2002. `www.it.kth.se/schulte/talks/FD\%20Systems.pdf`.

[17] Ian Sommerville, *Software Engineering, 6th Edition*, Addison-Wesley, 2000.

Social Minded Commitment Management

Robert Ross[1] and **Rem Collier**[2] and **G.M.P. O'Hare**[3]

Abstract. Over the past 30 years Artificial Intelligence has fragmented from one broad subject into a cluster of narrow but deep individual disciplines. During this time we have also seen the development of increasingly complex software systems for application domains such as robot control, mobile computing, and expert system interfaces. Many of these designs use elements from the branches of AI, but pay little attention to the integration of these elements in an intelligent way. This paper presents an approach to this intelligent integration problem, based on a community of Intentional Agents. Each of the agents within the community uses a Social Minded Commitment Manager (SMCM) to allow it to reason and cooperate in order to achieve goals when individual execution has failed. An implementation of the SMCM that has been developed for AgentFactory is presented, and its use then motivated through the description of a robust, redundancy tolerant robot control architecture named MARC.

1 Introduction

Over the past 30 years there has been a fragmentation of Artificial Intelligence into a multitude of deep specialised subdisciplines. Each of these individual branches are undoubtedly very important, providing us with useful algorithms for data extraction, natural language processing, planning and so forth. Little attention has however been given to the question of how individual components can be intelligently integrated in complex system designs.

The need for intelligent integration of AI techniques and algorithms is perhaps nowhere more manifest than in the production of intelligent service robots. Whereas simple control algorithms sufficed in the 80s, modern robot control architectures must integrate a diverse range of components that deliver support for tasks such as motor control, dialog management, and object recognition. Despite this large increase in the complexity of control systems, we have not seen a significant change in the approach taken to the integration of the individual components within these architectures.

Static, brittle, tightly-coupled architectures are still the order of the day in the large software designs. Although some level of disjunction is possible through the use of standards such as the Distributed Component Object Model (DCOM), these are merely communication protocols and do not improve the intelligence of the individual component or the larger system. The creation of large software systems using C like monolithic architectures, object oriented or DCOM models leads both to multiple points of critical failure, and a tightly coupled design which is not suitable for extension.

The authors reject this static design approach in favour of a dynamic metaphor based around a community of Intelligent Intentional Agents. In this community, agents are strong software entities which have inbuilt reasoning and plan execution ability. Such abilities allow for communication and cooperation within a larger disjoint software architecture. To improve this feasibility of this approach, we introduce a Social Minded Commitment Manager (SMCM) which improves on the basic cooperative skills of Intentional Agents. The SMCM is based around a formal Intentional Agent model. Details of this formal model are beyond the scope of this paper, and instead, the reader is directed to [2]. However, in order to facilitate the description of the SMCM and its implementation, we start with an informal review of the concept of an Intentional Agent.

2 Intelligent Intentional Agents

The notion of an Intentional Agent is broadly based on the work of the philosopher Daniel Dennett. In [4] Dennett introduces the concept of the "Intentional Stance" as a more appropriate way of modelling complex systems. Specifically, Dennett argues that, through the ascription of folk psychological notions such as beliefs, hopes, and goals, people are more easily able to understand behaviour of complex systems that, through the more traditional physical and design stances. It is this notion of the Intentional Stance, as applied from an internal perspective as a tool for modelling both the agent and its environment that categorises an Intentional Agent.

Initial work on Intentional Agents led to the design of a number of agent architectures that define the data structures and algorithms that are required to implement such an agent [5] [1]. For a review of some of the more prominent Intentional Agent architectures see [2]. Many of these architecture were based upon earlier theoretical work on Intentional Agents that employed three mental notions: Beliefs, Desires, and Intentions. Specifically, beliefs are taken to represent the agents current subjective knowledge of itself and its environment; desires represent the agents ideal state of the environment; and the intentions represent a chosen subset of those desires that the agent is committed to bringing about. Architectures that employ these mental notions, or variant of them, have become known as Belief-Desire-Intention (BDI) architectures. Similarly, theories based upon beliefs, desires and intentions have become known as BDI theories [6] [9] [11].

A chief concern underlying the area of Intentional Agents is the identification of a clear link between the various BDI theories and the associated BDI architectures. This issue

[1] Universitat Bremen, Germany, email: robertr@tzi.de
[2] University College Dublin, Ireland. email: rem.collier@ucd.ie
[3] University College Dublin, Ireland. email: gregory.ohare@ucd.ie

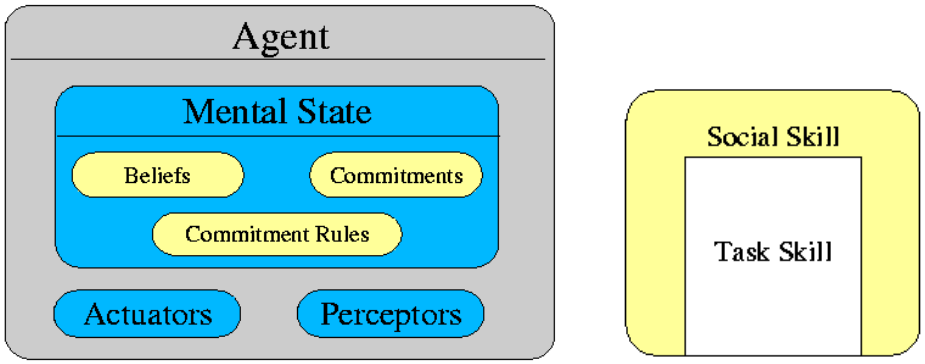


Figure 1. Two views of a Social Intentional Agent. On the left hand side we see an agent broken down in terms of its individual attributes e.g. beliefs, perceptors. On the right hand side we see an Intentional Agent as an entity with two distinct ability layers. This agent has high level reasoning and social abilities at the intentional level, while possessing task specific abilities to perform actions on data or the environment. An Intentional Agent Implementation provides the high level layer to all agents, while task specific abilities are 'plugged' into the core at application design time.

has been the subject of much research, and has led to the emergence of a class of programming language, known as Agent-Oriented Programming (AOP) languages [10]. These languages attempt to provide a strong link between the BDI theories, which are often logical and based upon the computationally intractable Possible Worlds semantics, and the syntax and semantics of the associated language. Some of the more prominent AOP languages include Agent-0 [10], AgentSpeak(L) [8], 3APL [3], and AF-APL [2].

At a high-level, these languages provide constructs for representing mental attitudes such as beliefs, goals, and commitments, together with sophisticated reasoning engines which relate and revise these mental attitudes. Conversely, at a low level the agents abilities are task related, and must be provided through application specific actuators and perceptors. Essentially the high level model provides control of what to do and why, while the low level provides a means of doing things (See figure 1).

A large number of BDI/Intentional Agent implementations centre around a formal model of commitment. Whereas classical systems reasoned about GOAL and ACTION, the intentional agent also reasons about the more abstract notion of commitments. Commitments may be understood by their common language meaning in that they are promises made by one agent to another (or oneself). By reasoning and managing these commitments, a more flexible approach to agent control is possible then would be available through the management of actions alone. Commitments between agents allow for a basic level of cooperation to take place between these agents. In terms of performing complex actions, the vast majority of Intentional Agent implementations acknowledge the need for an agent to use plans to achieve complex tasks. Unfortunately however many implementations choose to execute the plan directly. An alternative approach is to extended the commitment model to handle plan constructs directly. Collier [2] presented a commitment model and implementation where complex plans were resolved to constructs of commitments at runtime. Although increasing the complexity of the agents execution model, such an approach adds great amounts of flexibility over the direct plan execution approach.

In any Intentional Agent implementation, the management of commitments is one of the core processes within each agent. Furthermore the commitment management methodology is one of the most distinguishing features between different Intentional Agent formulations. Probably the best known variants on commitment management are concerned with the maintenence condition for a commitment i.e. under what conditions a commitment is adopted, maintained and dropped [9]. Although these commitment management approaches and their successors, recognise external agents as those who make requests for commitments, the external agents are ignored in the process of achieving commitments. The next section introduces a social commitment manager which although broadly based on the underlying logic of existing commitment managers, uses the agents social environment to help in the commitment management process in times of adversity.

3 Social Minded Commitment Management

A Social Minded Commitment Manager is now presented. The essential difference between this commitment manager and its predecessors is the inclusion of basic social skills in the commitment management cycle. Through the use of these skills the intent is to bridge a gap between tradition Intentional Agent based Agent Oriented Software Engineering, and the long promised emergent qualities of reactive Multi-Agent Systems.

3.1 The Approach

Commitment Managers have traditionally had a limited intelligence in how they attempt to manage and achieve their commitments. Although Collier's [2] run-time adoption of commitment structures to achieve plans was a substantial improvement over more standard direct plan execution, there is still little intelligence in failure handling. When an action within a plan fails then the agent/commitment manager can

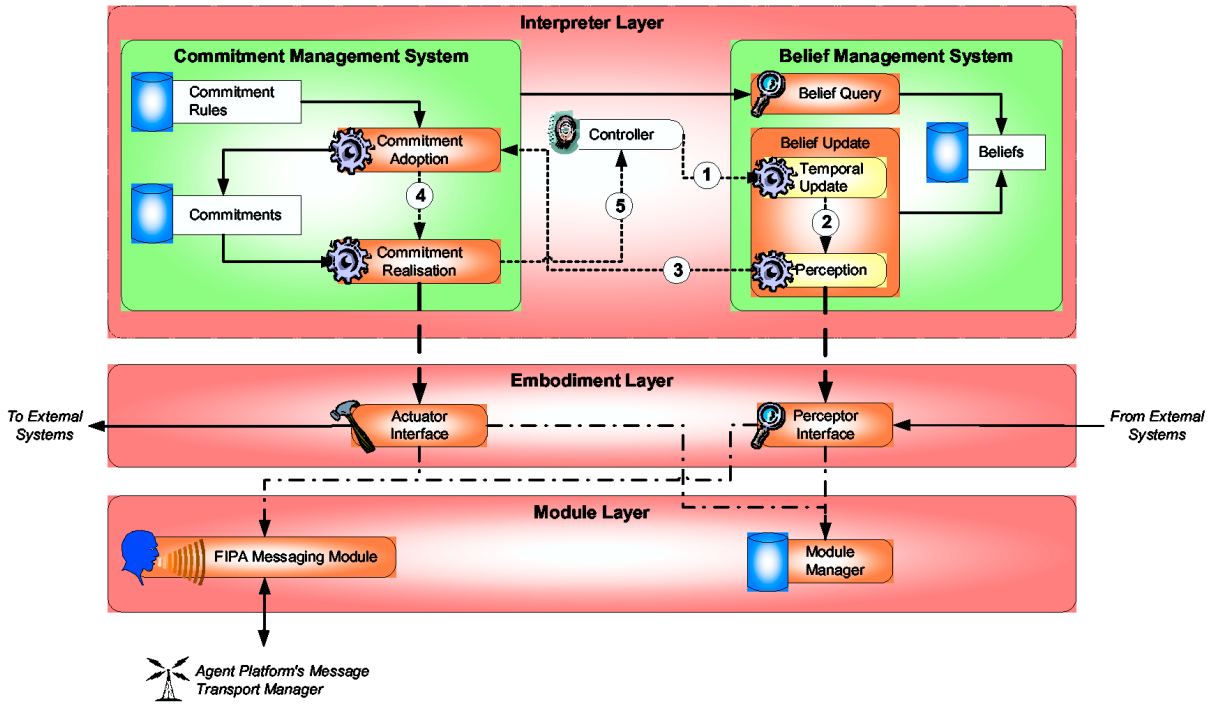


Figure 2. The AgentFactory Agent Programming Language Interpreter. The AF-APL comprises three layers. At the bottom a Module Layer for resource management, followed by an embodiment layer which governs perceptor and actuator execution. At the top layer, the Social Minded Commitment Manager sits alongside the Belief Management System, providing all high level reasoning for the agent.

only fall back on the contents of the plan. If the plan was not formulated in a way to deal with this situation explicitly, then the agent's commitments can fail easily.

Social Minded Commitment Management (SMCM) is different from other approaches in that the commitment manager is specifically designed to incorporate social skills to compensate for an agent's initial failure to achieve a commitment. To put it in other words, if an agent initially fails to achieve a task, then the SMCM is explicitly designed to try to achieve the task with the aid of *any* agents within it's environment. Rather than being an application specific ability, this behaviour is encoded within the intentional agent framework, thus endowing all such agents with these social abilities (See figure 1).

This loosely coupled connection used in failure handling is used at other times in SMCM based agents. For example acquaintances should not be hard-coded into design files or initialisation scripts. Instead all acquaintances are acquired at run-time with a dynamic *'Yellow Pages'*approach which allows an agent to initiate basic communication with any other agent in its environment. Whether communication continues, or is in any way productive is entirely dependant on the needs of individual agents. To improve openness this acquaintance acquisition process takes place on a regular basis, allowing newly inserted, copied or spawned agents to communicate and cooperate with the most appropriate agents within the community.

An agent using the SMCM can be seen in very abstract form in figure 1. The core of the agent are low level task capabilites. These capabilites are entirely application dependent and concern anything from sorting algorithms, to speech generation, to basic movement behaviours. Sitting on top of these task skills are the social and reasoning capabilites provided by the SMCM. Many of these qualities are provided by default by the SMCM and do not need to be designed for specific applications. These qualities of the social intention agent allow for the easy fabrication of communities of agents to perform tasks in the production of complex control systems. The loose coupling of this community of agents brings us close to the flexibility and robustness of a MAS, while the inherent social and reasoning skills allow for the creation of individually powerful agents capable of planning and deliberation. The implementation of the SMCM was based on a commitment management model originally given in [2]. The updated model is now discussed with emphasis on those features which were necessary to the production of the *social* aspects of the commitment manager.

3.2 Implementation - Extending AgentFactory

The Social Minded Commitment Manager approach discussed above is abstract in that it could be implemented on many different Intentional Agent Frameworks. In practice the system has already been implemented as an extension to the the

AgentFactory - Agent Prototyping Environment [4]. AgentFactory provides many other important constructs and resources needed to build intelligent intentional agents. These components include a *Belief Management* and a framework for the definition of plans, actuators and perceptors (see fig 2). AgentFactory also provides low level communication facilities which allow agents to communicate both on the same and multiple platforms. Communication is FIPA compliant, which means that AgentFactory agents can communicate in a meaningful way with FIPA compliant agents based on other systems.

The implementation of SMCM required a number of extensions to the AgentFactory development environment, and a reworking of the formal commitment model. Some changes were relatively trivial such as the introduction of mechanisms to allow an agent to become acquainted with all agents in it's environment at runtime. Other extensions included the expansion of the planning language to allow for universal operations over a set of elements within the agents mental state. Further extensions included the introduction of agent introspection, and the implementation of the Social Minded Commitment Manager algorithm. These two items will now be discussed in more detail.

Introspection Introspection essentially allows an agent to answer *what if* questions from another agent and from itself. More specifically an agent (Agent-A) can examine its own mental state to determine in advance the probable outcome of a request by another agent (Agent-B). Introspection is typically used when Agent-B makes an inquiry as to how the Agent-A would hypothetically respond to some request. Based on the results of its own introspection, Agent-A can then inform Agent-B of the possible result of the request (i.e. whether Agent-A would commit to the request or not). This information on the run-time capabilites of the agent, can then be used in the formulation of initial joint plans. Naturally either the agent or world state can change in between the initial introspection request and a subsequent actual request for the action. However the initial introspection result can often allow for the creation of plans which are successful in many cases.

Introspection is modelled as a core agent actuator which operates on the mental-state of the agent. During execution of this actuator a clone of the agent's mental state is made, and this mental state is run as if an actual request for GOAL from Agent-B had been received. The results of this hypothetical request to the agent can then be used to formulate a response to the hypothetical question from Agent-B.

The Social Minded Commitment Manager

The Social Minded Commitment Manager algorithm was built on-top of an improved plan description language, introspection, and the ability to dynamically acquire acquaintances within the environment. As mentioned earlier the implementation and model used are broadly based on that presented in [2]. In practice the implementation of the commitment manager is extremely complex, therefore a highly simplified view of it is presented in figure 3.

During any given execution cycle of an agent, a previously held commitment to some activity might be attempted by the agent. Traditionally an invalidation of the pre-condition or a problem with the direct execution of an actuator would cause a commitment to fail, and inevitably being dropped. Instead the SMCM commits the agent to a social plan to achieve the action through the help of other agents within the environment/platform. The typical structure of such a social plan is presented in figure 4.

[4] See www.agentfactory.com

```
manageCommitments()
{
   foreach(commitment_to_primitive)
   {
     // attempt to achieve commitment

     // if commitment fails due to invalid
     // pre-condtions on the actuator
     // or plan being attempted
          // then commit to a social plan
          // to get help from other agents
          // to achieve the goal.
   }
}
```

Figure 3. Simplistic view of the SMCM algorithm.

```
PLAN get_help(?goal);
BODY
  SEQ(acquire_acquantances,
      FOREACH(BELIEF(friend(?agent)),
           XOR(SEQ(request(capable(?goal),?agent)
               await_response(capable(?goal),?agent),
               request(?goal,?agent),
               await_response(complete(?goal),?agent),
               adopt_belief(?goal)   )   )   )
      );
```

Figure 4. A Social Plan which may be committed to in order to get help from another agent to achieve a goal ?goal automatically.

The plan is typical of a set of social plans used by the SMCM to achieve social goals. It is a social plan simply, in that it is a plan which is particularly concerned with social interaction. The plan is initialised with *?goal* which is some state of the world or action which must be achieved by an agent in order to facilitate the achievement of the original commitment. The first step of the plan involves an attempt to become acquainted with all agents contactable on the agent platform. This *acquire_acquaintances* is implemented by another plan which uses the agent platform to get a list of all agents which are interested in potentially giving aid to this agent. Each potential helper is then listed in Agent-A's mental state as a friend(?agent) where ?agent is a variable which resolves to a unique identifier of the friend agent.

The next step in the plan is a FOREACH term which operates over all of the friends which are held by the agent at that time. The second term of FOREACH will be expanded out for each ?agent which was resolved against friend(?agent). This section of the plan to be expanded is a XOR operation, which operates on a more basic plan segment which uses basic speech acts and introspection abilities to find one agent which

is capable of achieving ?goal for AGENT-A. If any agent is found, they will be requested to achieve ?goal, and if they report they were successful in that undertaking,then the ?goal will be added to the agents mental state. Such a successful outcome will then allow the agent to fulfil its original requirements.

This use of social ability to achieve commitments during failure conditions, is a unique feature of the Social Minded Commitment Manager. This is in contrast with other Commitment Managers which would give up on the commitment at that point, and instead resort to complete re-planning to achieving the high level goal. This in-built social skill allows a community of intentional agents operates more like a MAS, providing robustness through very loose coupling. A key point here is that these are basic skills which come out of the use of the commitment manager, and do not have to be explicitly considered by a designer in the process of fabricating individual agents. To demonstrate the SMCM approach, an application in the area of mobile robot control will now be described.

4 Application - Multi-Agent Robot Control

The SMCM has been successfully deployed in the field of autonomous robot control. The field of robot control architecture design has been a fruitful field of study for AI over the past 30 years, with an evolution of approaches to control. The first planning based *Sense-Plan-Act* architectures, gave way to the new school of reactive architectures in the mid 80s. Purely reactive architectures then gave way for the emergence of hybrid architectures in the early 90s. These hybrid architectures in principle combine the best parts of both the Sense-Plan-Act and reactive approaches.

Hybrid architectures are however not without fault, and hybrid architectures to date suffer from the deficiencies of static and monolithic design. Often the top layers of these systems are built around one *all powerful* agent [7]. This rigid methodology provides not only problems in initial integration, but also leeds to a lack of system robustness, since the failure of any one component can lead to a cascading failure of the entire system. One approach might be to rigidly model and formalise the design, to the extent that all behaviour can be explicitly predicted and analysed against requirements. However such an approach is unrealistic in systems using a vary large number of individual software components. A more dynamic approach to the construction of these systems is necessary. A loosely coupled intelligent integration framework can help to reduce many of these issues. Not only that but a loose MAS like coupling, leaves the door open for the emergent behaviour to meet a myriad of situations which were not pre-built into the system design.

To this end MARC the Multi-Agent Robot Control architecture was developed. MARC is shown in abstract form in figure 5. The architecture is a true hybrid architecture with functional, reactive, sequencing and planning capabilities. The architecture differs from other approaches though in that reactive, sequencing and planning capabilities are modelled as a community of social agents which vary in their deliberative and reactive capabilites. All agents within this community have been built using AgentFactory. Those not requiring reactive control use the full SMCM, while those requiring reactive support, forgo the full SMCM for a reactive commitment management model model. Essentially these reactive agents can answer requests from other agents, but are incapable of using SMCM mechanism for failure recovery.

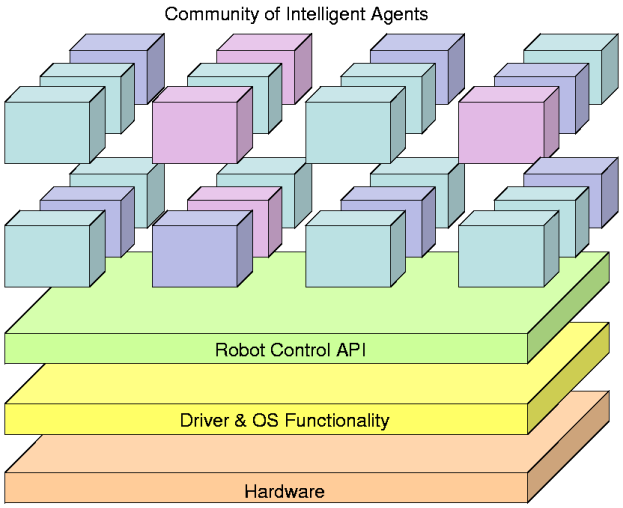


Figure 5. MARC - The Multi-Agent Robot Control architecture - Simplistic Layered View. The Social Minded Commitment Manager is used in the production of Deliberative Agents at the Community of Agents Level of Control.

MARC is being implemented as a control architecture for highly complex humanoid style robots. To this end many natural language processing agents have been developed in addition to the standard movement and command processing abilities normally associated with a mobile robot implementation. The architecture has been successfully deployed on Nomadic Scout II robots in University College Dublin, and is currently being deployed on the Rolland, the autonomous wheelchair in the University of Bremen, Germany.

5 Related Work

Haddadi has recently addressed the question of how intentional agents can form social relationships to achieve complex tasks [6]. Her formulation mainly focused on the production of basic plans through commitment negotiation. The approach did not however deal with the realities of failure and negotiation to recover from a failure which has already taken place.

6 Conclusions & Future Work

This paper presented the Social Minded Commitment Manager as an extension of traditional Intentional Agent Commitment Management approaches. The commitment manager uses social plans and run-time cooperation to attempt to maintain commitments in times of adversity. Such a design leads to a more dynamic Multi-Agent System based approach to complex software architectures, while maintaining the inherent computational power of Intentional Agents. The SMCM has been successfully integrated into AgentFactory,

and has been subsequently used in the production of a robust robot architecture. It is intended that the SMCM implementation brings us one step closer to the intelligent integration of complex software systems.

Specific future work on the SMCM includes the extension of dynamic planning capabilites available to the SMCM. With relation to this, extensions will be provided to allow for true joint planning based on introspection and dialog. Non development work on the SMCM includes the derivations of experimental scenarios which allow for its quantitative evaluation against more traditional commitment managers. This however is non-trivial since the SMCM is intended to be most useful in highly complex software architectures, which are inherently difficult to quantitatively evaluate.

6.0.1 Acknowledgements

We gratefully acknowledge the support of Enterprise Ireland through grant No. IF/2001/02, SAID.

REFERENCES

[1] M. Bratman, D.J. Israel, and M.E. Pollock. Plan and resource-bounded practical reasoning, 1998. Computational Intelligence 4(4), pp349-355,.

[2] Rem W. Collier, *Agent Factory: A Framework for hte Engineering of Agent Oriented Applications*, Ph.D. dissertation, University College Dublin, 2001.

[3] M. Dastani, F. Dignum, and J.J. Meyer. 3apl: A programming language for cognitive agents. ercim news, european research, 2000. Consortium for Informatics and Mathematics, Special issue on Cognitive Systems, No. 53,.

[4] Daniel C. Dennett, *The intentional stance*, The MIT Press, Massachusetts, 1987. 388 pages, 1987.

[5] M.P. Georgeff and F.F. Ingrand, 'Decision-making in an embedded reasoning system, proceedings of the international', in *Joint Conference on Artificial Intelligence (IJCAI'89), pp 202-206, Detroit, Michigan, USA, 1989*, (1989).

[6] Afsaneh Haddadi, *Communciation and Cooperation in Agent Systems: A Pragmatic Theory*, number 1056 in Lecture Notes in Computer Science, Springer-Verlag: Heidelberg, Germany, 1996.

[7] K. Konolige, K. L. Myers, E. H. Ruspini, and A. Saffiotti, 'The Saphira architecture: A design for autonomy', *Journal of experimental & theoretical artificial intelligence: JETAI*, **9**(1), 215–235, (1997).

[8] A. Rao. Agentspeak(l): Bdi agents speak out in a logical computable language, seventh european workshop on modelling, 1996. Autonomous Agents in a Multi-Agent World, Institute for Perception Research, Eindhoven, The Netherlands.

[9] Anand S. Rao and Michael P. Georgeff, 'Modeling rational agents within a BDI-architecture', in *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, eds., James Allen, Richard Fikes, and Erik Sandewall, pp. 473–484. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, (April 1991).

[10] Yoav Shoham, 'Agent oriented programming', *Artificial Intelligence*, **60**, 51–92, (1993).

[11] M. Wooldridge, *Reasoning about Rational Agents*, Intelligent Robots and Autonomous Agents, The MIT Press, Cambridge, Massachusetts, 2000.

Creating a Meta-Controller Using Adaptive Fuzzy Control with GAs

Finlay S. Smith[1]

Abstract. A technique for creating an adaptive meta-level controller is outlined in this paper. These controllers will act as a meta-level controller in non-deterministic and evolving environments.

Approaches to such a problem, such as Neural Networks and Fuzzy Logic suffer from disavantages, techniques that use fuzzy logic tend to suffer from an explosion in the number of rules whilst neural networks lack an ability to explain their reasoning. Further, both of these techniques tend to lose histrical data and reflect the more recent adaptations.

The technique presented in this paper performs the adaptive process on a fuzzy rule base of fixed size through the application of a Genetic Algorithm, for a strictly limited number of generations. This approach overcomes the disadvantages in existing adaptive controllers. The results of this technque are illustrated, firstly with a simple control engineering problem and then to the upredictable environment of game playing. The results show that the technique can be successfully used to adapt to unpredictable environments.

1 GA Adaptive Control

The technique described in this paper, combines the adaptive features of self-organising fuzzy logic controllers [7, 9, 11, 12, 18] is the adaptation of the rulebase based upon perceived discrepancies in the controller's output. In such a controller the rules are adjusted when the controller is perceived to have produced an inadequate output.

There are essentially two components in a self organising fuzzy logic controller, the object level rule-based controller and the self organising component (which adapts the rulebase). The rulebase of the object level controller consists of rules of the form (the example here is for a simple controller that has only one input and one output): $E_k \rightarrow C_k \rightarrow U_k$ where E_k is the fuzzy subset in rule k representing the measured error in the behaviour of the system under control (the difference between the actual value of a system variable and its ideal value), C_k is the fuzzy subset in rule k representing the change in the measured error (compared to the previous measurement of the error) and U_k is the fuzzy subset in rule k representing the output of that rule (or the control adjustment that is to be applied to the system under control).

Two limitations of self-organising fuzzy logic controllers are that the rule base can grow exponentially, and that the adaptation process can erode previously learnt knowledge.

One approach to overcome these problems, would be to use a GA to learn the rulebase [4, 5, 13]. This idea has also been applied to the adapive control of physical systems [1, 3, 6, 10]. The work presented in this paper, differs from these approaches in that it is designed to control non-conventional processes, such as card playing or meta-level control.

These processes are different from conventional control problems as there is no definite solution to a given problem. For example in card games, each player only (generally) knows which cards are in their own hands. The correct action to take depends not only upon the cards in the hand but also what is in the other players hands. It is therfore impossible for a player to make a perfect decision. As a result the decisions to be made are not as determinable as in conventional controllers.

The approach adopted in this paper is to adapt a fixed size fuzzy rule-base by running a genetic algorithm for a strictly limited number of generations.

Each member of the population in the GA is respresented by a series of numbers which represent the 'centre' of each of the triangular fuzzy sets used in each of the rules. In addition to this an additional element is used to represent the width of the fuzzy sets. As the GA evolves, it is effectively moving the centre of its fuzzy sets to try and find a better solution.

The strict limitation on the number of generations has two distinct advantages (over running the GA for longer):

- Limiting the number of generations, results in a fast modification of the rulebase. This is particularly important if time is relevent, for example in card playing it would not be acceptable for the adpatation process between hands took more than a few seconds.
- As there is no definite solution to these problems, running the GA to convergence may result in the rulebase 'overtraining' on the data available to it and losing the ability to generalise. By limiting the number of generations the possibilty of overtraining is avoided. There is a possibility that the number of generations is too small. This may not be significant as every time the rulebase is revised the historical data is all used in the fitness function, therefore the rulebase will adapt with respect to all of the data everytime it evolves. This inverts the limitation of self-organising fuzzy logic controllers losing historical data, in fact the rulebase will tend to reflect the older data rather than the newer data.

The result is a technique that allows the rulebase to be modified whenever it is required to do so, without the limitations of self-organising fuzzy logic controllers.

2 Test Systems

To illustrate the effectiveness of the technique outlined in this paper, a relatively, simple test case will be used. It should be noted that the intended application domains for the techniques described in this paper are not for conventional control domains, rather for less

[1] National University of Ireland, Galway

certain, or high level control tasks. In particular this technique is intended to be used as the basis for the automatic creation of a card playing system [15] and for learning meta-heuristics to solve the Vehicle Routing Problem [16]. The initial test case given in this paper is intended to illustrate the effectiveness of the techniques, before applying them to these more complex domains. Initial results from applying the technique to learning the game of Hearts are also presented.

2.1 The Cart-Pole System

The non-linear cart-pole system [2, 14] is chosen to act as the system under control as it is commonly used as a test case for fuzzy logic controllers [8]. This system involves trying to balance a pole on a moveable cart. Both the pole and the cart are restricted in their movements to one dimension (the cart can only move forwards and backwards and the pole has a hinge pivot so that it can only move in a one dimensional space.

The system is controlled by a force that is applied to the cart, either to try to push the cart forward or to pull the cart backwards. The pole is required to be kept almost upright, and the cart is required to stay close to a fixed point. Diagrammatically the system under control is shown in figure 1.

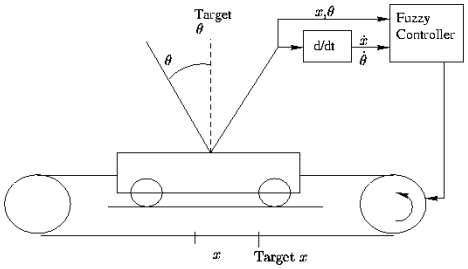


Figure 1. The Cart Pole System

The system consists of four state variables (the inputs to the controller) and one input variable (the result of the controller), defined as follows:

x = the position of the cart relative to its normal position.
$\dot{x}$ = the velocity of the cart.
θ = the angle of the pole with respect to the vertical (dotted) line.
$\dot{\theta}$ = the angular velocity of the pole.
u = the force applied to the cart, which is the only output variable of the FLC.

The cart-pole system can be modelled by the following non-linear differential equations [10]:

$$\ddot{\theta} = \frac{gsin\theta + \frac{cos\theta[-u-ml\dot{\theta}^2 sin\theta+\mu_c sign(\dot{x})]}{(m_c+m)} - \frac{\mu_p\dot{\theta}}{ml}}{l[\frac{4}{3} - \frac{(mcos^2\theta)}{(m_c+m)}]}$$

$$\ddot{x} = \frac{u + ml[\dot{\theta}^2 sin\theta - \ddot{\theta}cos\theta] - \mu_c sign(\dot{x})}{m_c + m}$$

The meaning of the parameters appearing in the system is presented in table 1.

The fuzzy logic controllers that will be generated to control this system will be based upon a relatively simple representation. Each

[htb]

Table 1. The Parameters of the Cart-Pole System

Parameter	Meaning
g	acceleration due to gravity $(9.8ms^{-2})$
m_c	mass of the cart in kilograms
m	mass of the pole in kilograms
l	half the length of the pole in meters
μ_c	coefficient of friction of the cart on the track
μ_p	coefficient of friction of the pole on the cart

controller will consist of 10 rules, each of which has 4 inputs ($\theta, \dot{\theta}, x\ and\ \dot{x}$) and one output (the force to be applied). Each of these inputs and outputs uses the simplest of representations, the symetrical triangle.

As each input and output is represented as a triangle, only the value of the apex needs to be generated by the GA. The chromosome consists of 50 numbers representing the apexes of each of the triangles in the 10 rules and 1 number representing the width of the triangles, giving an overall chromosome size of 51 elements. The widths of the traingles could have been set to some constant value, but this may have had an artificial impact on the effectiveness of the controller. The widths of each triangle could also have been generated by the GA, but it was felt that this would unnecessarily increase the complexity of the GA. Allowing one width to be generated by the GA was a comporomise between these two positions.

The width of the traingles in the rules was also used in the fuzzification process, again it was felt that this would simplify the GA. Defuzzification was carried out using the centre of gravity defuzzifier.

The other stages in the process are the normalisation and denormalisation. For the purposes of this paper, the rather simplistic approach, of linear normalisation and denormalisation was adopted. This may disadvantage the controllers, but it allows the performance of the controller to be fully dependant on the GA rather than partially on the normalisation.

The fitness function to be used by the GA, is simply the absolute sum of the cart displacement and the pole displacement throughout the simulation. As a result the controllers that have the lowest value of fitness function are those that keep the pole most upright and the cart nearest its set point.

To test the adaptivity of the technique, a starting point of a working controller would be desirable. The process could start with no controller and have to learn from scratch, however, as the technique is designed to adapt existing controllers it was decided that starting from an existing controller was reasonable. To create the initial controller, the GA process was run for 1000 generations, with a simulation that started with an initial pole displacement of 10 degrees. The 1000 generations was used to ensure that the GA found a controller that was able to control the cartpole system with this initial displacement.

The results of this intialisation process can be seen in figures 2 and 3 which show the behaviour of the pole and the cart following an initial displacement of 10 degrees.

To test the adaptive process, this initial controller was then adapted by running the GA for a strictly limited number of generations with new initial conditions. The fitness function would then be based upon the abilty of the controller to control the system over a variety of different initial conditions as described in the next section.

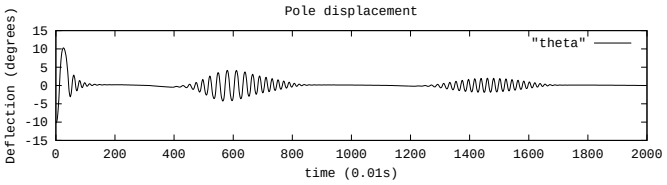


Figure 2. The Behaviour of the Pole Following an Initial Pole Displacement of 10 Degrees

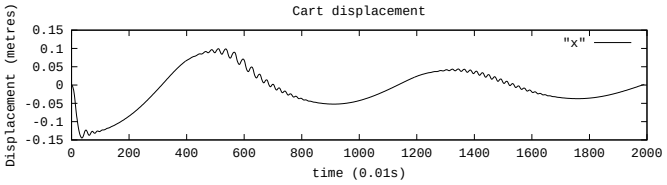


Figure 3. The Behaviour of the Cart Following an Initial Pole Displacement of 10 Degrees

The symmetry of the cart-pole system could have been utilised in this simulation, greatly simplifying the chromosome required for the GA. The number of rules generated by the GA could have been halved, and the full rulebase created by creating a mirror image of each of the rules. This would not only have simplified the chromosome, but it would also have improved the overall performance of the controller, as a behaviour learnt for a negative pole displacement could be inverted and applied to a positive displacement. As the purpose of the work presented here, is not to create the optimum controller, but rather to create a generic technique, the use of this symmetry information was not used, making the resultant process more generic.

2.2 The Game of Hearts

This section introduces the game of Hearts and describes how the use of the technique outlined in this paper can be utilised to learn strategies.

2.2.1 Overview of Hearts

Hearts is a trick based game, for four players[14], where the aim is to score as few points as possible. At the start of the game each player starts with no points and the game ends when a player has scored more than one hundred points. The player with the fewest points at this stage is the winner.

Every card in the Hearts suit carries a penalty of 1 point and the Queen of Spades carries a penalty of 13 points. In addition the Jack of Diamonds carries a reward (which reduces their score) of 10 points. The aim of the game is therefore to avoid taking any Hearts cards and the Queen of Spades, whilst trying to take the Jack of Diamonds.

An additional element of the game is called Shooting the Moon. To Shoot the Moon a player is required to take all of the Hearts cards and the Queen of Spades (they do not need to take the Jack of Diamonds). The reward for Shooting the Moon is that each of the players opponents receive a 26 point penalty (or the player can choose to give themselves a 26 point reward). A failed attempt at Shooting the Moon can result in a player taking almost all of the points, so players have to be careful when attempting to Shoot the Moon.

The game consists of a series of hands, for each of the hands the full pack of 52 cards is dealt to the players so that each of them has 13 cards. Once the cards have been dealt each player chooses three cards to pass to one of the other players. The passing alternates as follows, for the first hand cards are passed to the left, for the second hand cards are passed to the right, for the third hand cards are passed across and for the fourth hand no cards are passed. For the fifth and subsequent hands this cycle is repeated.

When choosing which cards to pass each player needs to consider not only the effects of passing cards, but also which cards they are likely to be passed. For example a simple strategy may be to pass any high Spades (to attempt to avoid taking the Queen of Spades) or any high Hearts. An alternative strategy is to pass low Hearts cards if a player is considering trying to Shoot the Moon.

Once the cards have been passed, the player that holds the 2 of Clubs starts the first trick by playing the 2 of Clubs. Each of the players them plays a card in sequence, always following suit (playing a card in the same suit as the first card) if they can. If a player cannot follow suit they can play any other card (except in the first trick when no penalty cards can be played). The trick is won by the player who played the highest card. Play proceeds with the player who won the previous trick plays the first card in the next trick.

When no cards remain the hand is over and each players points are added (or subtracted) to their score. Play continues in this manner until the game is won.

The game is mainly fairly straightforward, however there are several features that make the game more complex and therefore interesting from a learning perspective:

- Passing cards. This adds an initial element of uncertainty and complexity as care has to be taken to try and avoid situations where a player is forced to play a penalty card that wins a trick.
- The Queen of Spades. As the Queen is not the highest card in its suit, it is very possible to be forced to take the Queen by playing the Ace or the King of Spades. The penalty for taking the Queen is so high that it is often the primary concern of players to avoid taking it.
- The Jack of Diamonds. Some versions of Hearts do not reward players for taking the Jack, but including the Jack significantly increases the strategic element of the game. If the Jack did not carry a reward, a reasonable strategy for a player to adopt would be to try to avoid taking any tricks at all (by passing all of their highest cards). By rewarding players for taking the Jack of Diamonds, this strategy is no longer such a strong one due to the benefit of holding onto high (Jack, Queen, King and Ace) Diamonds. However the risk of holding onto these high cards is that a player may be forced to take a trick containing the Queen of Spades without any guarantee of taking the Jack of Diamonds.

The complexities of the game of Hearts mean that, whilst it is a relatively easy game to learn, the complexities make it a relatively difficult game to master. It is easy to see how a system that can learn how to play Hearts could be modified to allow it to learn to play other card games.

2.2.2 Learning to Play Hearts

The card playing system is based upon ideas developed as part of a Diagnostic Strategist [17, 14]. The purpose of a diagnostic strategist is to guide a fault diagnosing process during its attempts to diagnose faults, in an effective and efficient manner. This can be achieved by

abstracting the effects of previous faults in order to provide meta-level guidance. The more experience that a diagnostic strategist has had, the more likely it is to provide accurate guidance to the diagnostic system, as the diagnostic strategist is more likely to have encountered a similar enough situation before. The guidance given to the diagnostic system may be used, by the diagnostic system, to adapt the current model of the system under diagnosis, and ultimately to make a diagnosis of the faults in the physical system. The guidance that was given by the diagnostic strategist may not have been the best possible guidance that could have been given, in which case the diagnostic strategist will need to be updated to reflect the better guidance.

The diagnostic strategist developed is the self organising component of a self organising fuzzy logic controller with a few alterations. A self organising fuzzy logic controller was chosen to perform this task as it can easily perform unsupervised iterative learning. This diagnostic strategist alters the rulebase of the object level controller which acts as the candidate proposer in the diagnostic system. Two of the problems with this approach are the exponential growth in the number of rules (which can make this approach intractable) and the tendancy of the rulebase to reflect more recent data at the expense of historiacl data. Work has been carried out to reduce the number of rules [18], but this does not address the loss of historical data (indeed the reduction in the number of rules can exacerbate the loss of such data).

Building upon the work of the Diagnostic Strategist, and utilising GAs to adapt the rulebase should overcome both of the identified limitations of the Diagnostic Strategist. The number of rules is fixed so the rulebase does not grow expontentially, and as a record is kept of all of the hands played (and the points taken) the revised rulebase can still reflect the entrire available knowledge, rather than favouring the most recent data. The card player system will use several related controllers to control different aspects of the game playing process (only the first controller has been developed so far):

- Selecting the cards to be passed. This controller could have been developed using 52 inputs (one for each possible card) and 52 outputs, however this would have led to an unacceptably complex controller. In addition the training of the controller would require a vast set of test data. In order to significantly reduce the complexity of this controller the cards that have been dealt will instead be represented as a set of metrics that indicate the kind of cards in the hand. The chosen metrics were:
  - Low Clubs - 2 to 7 of Clubs
  - High Clubs - 8 to Ace of Clubs
  - Low Diamonds - 2 to 10 of Diamonds
  - High Diamonds - Jack to Ace of Diamonds
  - Low Spades - 2 to 10 of Spades
  - Queen of Spades - Queen of Spades
  - High Spades - King and Ace of Spades
  - Low Hearts - 2 to 7 of Hearts
  - High Hearts - 8 to Ace of Hearts

  The values of the metrics are calculated by multiplying the face value of each card in that metric. For example if a player held the King and Ace of Spades the value of the metric for High Spades would be $12 \times 13 = 156$. This greatly simplifies the controller as there will be 9 inputs and 9 outputs. The outputs will use the same ranges as the above metrics, but their value will represent the perceived benefit of passing cards in that metric. The metric with the highest value will have their cards passed. The learning will be achieved by recording the number of points taken for each pass, if any points are taken the controller will adapt itself to avoid suggesting a similar pass being made in the future.
- The second controller will control the general game play, that is which cards to play. This controller will be different from the previous controller as there will generally be some knowledge of the contents of the opponents hands (either direct knowledge from the cards the player passed or inferred knowledge from the cards the other players have played). Additionally, information about which cards have been played, in particular which penalty cards have been played, will be available. This controller will use the same metrics as the previous controller, but will be augmented by additional information regarding the state of the game. The learning in this controller will be performed in a similar manner to the previous controller.
- Individual player strategy. This controller will allow the system to distinguish between individual players. This will allow the card player to adopt different strategies to different players whilst still improving its general performance.
- The final controller will be used to record the current state of the game. Information about which cards have been played will be recorded and deductions will be made about the cards remaining in opponents hands. This controller will be different from the previous ones as it is mostly recording information rather than making suggestions about which card to play. It will use card-counting to make its deductions.

The controllers will not always be required, for example if a player has only a single card in the led suit it can just be played without using any of the controllers.

The selection of the metrics used in the controllers was based upon the relative importance of each suit. Further work to automatically determine the metrics is required. In particular the number of metrics for the Hearts suit may be insufficient.

3 Results - Cart-Pole System

For all of the test results in this section, the starting controller will be the one that was trained for an initial pole displacement of 10 degrees as given in the previous section. The initial population will simply be the final population generated during the initialisation process, this has the advantage of having an initial population that contains some variation, whilst still being a reasonable solution. This approach, of using previous populations, is also used in the tests where the rules are adpated more than once.

3.1 Fitness Function

In each of these test cases the fitness function will again be the sum of the pole and cart displacements, though this time the sum will be across each of the simualtions in the current test. As a result, the fitness function is aiming to minimise the cart and pole displacement over all of the simulations. A consequence of changing the fitness function each time the adaptive process is called is that the fitness function effectively evloves along with the controllers.

3.2 Inverting the Pole Displacement

Figure 4 shows the behaviour of the pole following a displacement of -10 degrees, whilst figure 5 shows the same test following limited adaptation.

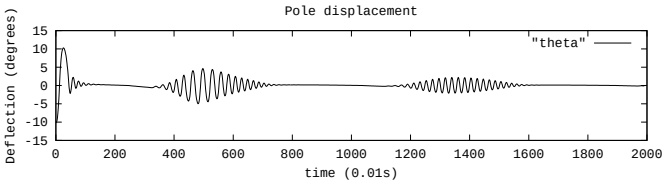


Figure 4. The initial Behaviour of the Pole with an Initial Displacement of -10 Degrees

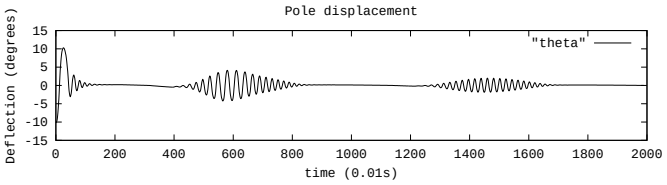


Figure 5. The Behaviour of the Pole with an Itial Displacement of -10 Degrees After Short Evolution

The initial behaviour of the pole is good though they only slightly improved the behaviour of the pole. This behaviour is not unexpected, due to the symmetry of the cart pole system, which has resulted in the controller that was trained on an initial displacement of 10 degrees being very good at controlling an initial displacement of -10 degrees. It is therefore difficult for the limited evolution to greatly improve the performance of the controller.

This simple test shows that even running the GA for only 10 generations has had a positive effect on a reasonably well trained controller.

3.3 A Large Initial Pole Displacement

This test started with the same initial controller as the previous test, this time however the pole was displaced by 15 degrees. Before further evolution the controller was unable to keep the pole upright, indeed it fell down after less than 1s.

Running the GA for a further 10 generations, using a displacement of 15 degrees resulted in the controller being able to successfully keep the pole upright as can bee seen in figure 6.

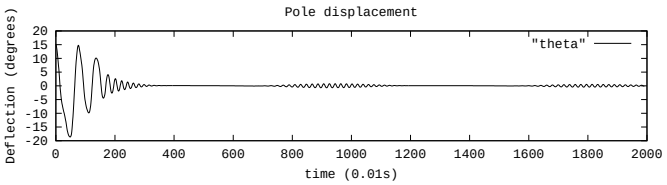


Figure 6. The Behaviour of the Pole with an Initial Displacement of 15 Degrees After Short Evolution

This result shows that after a limited number of generations, the Fuzzy Logic Controller has been successfully adapted to allow it to control a situation where, previously, it had been unsuccessful.

4 Results - Card Passing in Hearts

This section summarises the results of applying the technique to one aspect of the game of Hearts, card passing.

4.1 Fitness Function

The fitness function used for these results is the Euclidean seperation between the number of points taken and the number of points predicted by the rulebase. The closer the predicted values are to the historiacl values the more the rulebase reflects the data. By using the Euclidean seperation, those data items that are most different will have most impact on the fitness function and should therefore be the measurements that the GA is attempting to reduce.

4.2 Results

The results were an improvement on previously recorded results [15].

In these previous tests, the controller was able to suggest which cards to pass. After only 5 hands the passed cards were almost as expected and after 25 hands the quality of passing was such that the human player won every game (though not necessarily every hand).

The results this time were better overall. The output from the controller is now an estimation of the number of points, rather than a recomendation of which card to pass. This simplifies the process as the historical record of cards passed and points taken can be used directly to evaluate the performance of each controller.

Initially, as expected the controller simply picked cards at random. However, from the second hand onwards the results improved. The controller consistently suggested that passing the queen of hearts would minimize the number of points taken, this is due to the severe penalty for taking the queen, which creates a large incentive to dispose of the card. This is a naive approach, as it can often be beneficial to hold onto the queen, especially if you also hold a good number of other spades. After about 15 hands, however, the reliance on wanting to pass the queen every hand began to reduce and the controller only began to suggest passing the queen when it was reasonable to do so.

During the training, it was noticed that the version of Hearts software being used had a habit of passing clubs whenever it can. This makes it fairly safe to pass certain combinations of cards, as it is unlikely to pass mainly high hearts and spades for example. When a human plays against this software, they can quickly adapt their game to the style of the computer players. It was interesting to note that after about 12 hands, the controller began to consistently take advantage of the computer players tendancy to pass ceratin cards, showing that it had adapted itself to suit the way the game was being played by the other players.

5 Conclusions

The results presented in this paper, indicate that using a GA to adapt a fuzzy rulebase has the potential to create a controller that can adpat in an uncertain environment.

One perceived limitation of the results is that a human player still plays the game once the cards have been passed. It is therefore possible that the human player is adapting their style to suit the performance of the card passing controller. It is therefore desirable to take the existing work and extend it. Considerable work is still required to entirely automate the game playing process. The most obvious work that still needs to be completed is the development of the remaining controllers in the game to allow entire games of Hearts to be played automatically. The current work relies on a human to actually play the game and so a more consistent performance could be expected from an automated player which could improve the way that cards are selected for passing.

Further work is also required to refine the metrics that are used to represent the hands. The current metrics were selected in a fairly

arbitrary manner to attempt to reflect the importance of each suit. An improvement to this would be to use historical passing data to determine clusters and hence create new metrics.

The same approach could also be applied to other card games to determine if the approach was general enough to work in a broader context. Similarly work could be undertaken to apply the same techniques in other fields such as meta-level heuristics.

REFERENCES

[1] P. Arabshahi, R. Oh, T. Caudell, and J. Choi, 'Pointer adaptation and pruning of min–max fuzzy inference and estimation', *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing, vol. 44*.

[2] H. R. Berenji, 'A reinforcement learning-based architecture for fuzzy logic control', in *Readings in Fuzzy Sets for Intelligent Systems*, ed., R. Yager D. Dubois, H. Prade, pp. 368–380. Morgan Kaufmann, (1993).

[3] L. Chen and C. Chiang, 'New approach to intelligent control systems with self-exploring process', *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, **Vol. 33, No. 1**, 56–66, (2003).

[4] F. Cheong and R. Lai, 'Constraining the optimization of a fuzzy logic controller using an enhanced genetic algorithm', *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, **Vol. 30, No. 1**, 31–46, (2000).

[5] M. G. Cooper and J. J. Vidal, 'Genetic design of fuzzy logic controllers: The cart and jointed-pole problem', *Proceedings Third IEEE International Conference on Fuzzy Systems*, 1332–1337, (1994).

[6] C. L. Karr, 'Adaptive process control using biological paradigms', *Journal of Network and Computer Applications*, **19**, 21–44, (1996).

[7] C. H. Lee and S. D. Wang, 'A self-organizing adaptive fuzzy controller', *International Journal of Neural Systems*, **80(3)**, 295–313, (1996).

[8] C. J. Lin and C. T. Lin, 'Reinforcement structure/parameter learning for neural-network-based fuzzy logic control systems', *IEEE Transactions on Fuzzy Systems*, **2**, 46–63, (1994).

[9] C. J. Lin and C. T. Lin, 'Adaptive fuzzy control of unstable nonlinear systems', *International Journal of Neural Systems*, 283–298, (Vol. 6 No. 3 1995).

[10] M. Mahfouf, M. F. Abbod, and D. A. Linkens, 'Online elicitation of mamdani-type fuzzy rules via tsk-based generalized predictive control', *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics*, 465–475, (Vol. 33, No. 3 2003).

[11] A Patrikar and J Provence, 'Control of dynamic systems using fuzzy logic and neural networks', *International Journal of Intelligent Systems*, 727–748, (vol 8 1993).

[12] T. J Procyk and E. H Mamdani, 'A linguistic self-organizing process controller', *Automica*, 15–30, (vol 15 1979).

[13] T. L. Seng, M. B. Khalid, and R. Yusof, 'Tuning of a neuro-fuzzy controller by genetic algorithm', *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 226–236, (Vol. 29, No. 2 1999).

[14] F. Smith, 'Selection of fuzzification, inference and defuzzification techniques for fuzzy logic control', *Society of Manufacturing Engineers, Technical Report RP01-145*, (2001).

[15] F. Smith, 'Towards creating an adaptive intelligent card player', in *Proceedings of the 2001 UK Workshop on Computational Intelligence*, ed., Q. Shen, pp. 144–149, (2001).

[16] F. Smith, A. Tighe, and G. Lyons, 'Towards a multi-algorithm vehicle routing problem solver', in *Proceedings of the 2002 UK Workshop on Computational Intelligence*, ed., J. A. Bullinaria, pp. 177–184, (2002).

[17] F. S Smith, 'Combining symbolic conflict recognition with markov chains for fault identification', *PhD Thesis*, (2002).

[18] F. S Smith and Q Shen, 'Towards guiding fault diagnosis based on experience', *Proceedings of the Thirteenth International Conference on Artificial Intelligence in Engineering*, 161–164, (1998).

Collaborative Ranking in Community-Based Web Search

{**Barry Smyth** and **Jill Freyne** and **Maurice Coyle** and **Peter Briggs** and **Evelyn Balfe**} [1,2]

Abstract. As the search engine arms-race continues, search engines are constantly looking for ways to improve the manner in which they respond to user queries. Given the vagueness of Web search queries, recent research has focused on ways to introduce context into the search process as a means of clarifying vague, under-specified or ambiguous query terms. In this paper we describe and evaluate a novel approach to using context in Web search that seeks to manipulate the results of a generic search engine for the needs of a specialist community of users. This collaborative search method enjoys significant performance benefits and avoids the privacy and security concerns that are commonly associated with related personalization research.

1 Introduction

The Web is rapidly becoming a victim of its own success. As it continues to grow, users are finding it more and more difficult to locate the right information at the right time. Even with the help of the most advanced search engines we regularly fail to locate relevant information in a timely manner. Many factors contribute to this access problem. Certainly, the sheer quantity of Web information, and its growth rate, tax even the most advanced search engines. For example, various estimates indicate that even the largest search engines cover only a fraction of the available information space [13]. Their spidering and indexing systems simply cannot keep up with the growth of the Web with respect to their indexing of newly created documents or the re-indexing of recently updated documents. However, this search engine coverage issue is just part of the problem, and indeed can be relieved by using meta-search methods [5, 18].

Another problem stems from the fact that the average Web user is not an information retrieval expert. Even when they choose reasonable query terms to guide search, the resulting queries are rarely complete in the way that they reflect the search needs of a given user; their queries are often vague and imprecise. For example, a query might include terms that identify the primary information target, but might exclude terms that usefully describe the search *context*. For example, a simple query for "*cbr*" does not indicate whether the user is interested in Case-Based Reasoning or the Central Bank of Russia, and queries for "*D. McSherry*" do not distinguish between the University of Ulster lecturer and the well-known novelist, Frank D. McSherry. Thus, many researchers have recently focused on ways to exploit context in Web search as a means of resolving ambiguity (eg. [11, 6, 7, 4, 8, 17]).

In this paper, we describe a novel, simple, yet powerful technique to exploit context during search (Section 3). This *collaborative search* method acts as a post-processing service for existing search engines and re-ranks results based on the learned preferences of a community of users; see also [19]. We describe its implementation in the I-SPY system (http://ispy.ucd.ie) and show how I-SPY achieves this level of personalization in an anonymous fashion, without storing individual user profiles, thus relieving many of the usual privacy issues associated with personalization techniques. In Section 4 we discuss the results of a new large-scale evaluation of I-SPY across a number of different topic domains. We show how I-SPY's collaborative ranking engine can significantly improve result precision and recall when compared to benchmark search engines, and we argue that these advantages make collaborative search particularly well suited to device-limited information retrieval tasks, for example, search on mobile devices such as WAP phones and PDAs.

[1] Smart Media Institute, University College Dublin, Dublin, Ireland email:{firstname.lastname}@ucd.ie

[2] The support of the Informatics Research Initiative of Enterprise Ireland is gratefully acknowledged

2 Background

For the most part, recent search engine advances have focused on improving existing indexing and ranking techniques (eg. [3, 9]). However, vague queries remain a significant problem leading to a growing body of research looking at ways to supplement such queries with missing context terms (see also [11]). Context information can be generated according to two basic approaches: either it can be explicitly provided by the user or search engine or it can be implicitly inferred from the local search environment.

2.1 Explicit Context

Perhaps the simplest way to capture explicit user context is to ask users to provide context terms as part of their search query. For example, Inquirus 2 [6] asks users to select from a set of categories such as "research paper", "homepage" etc. and uses the selected context categories to choose target search engines for the user's query; as such Inquirus 2 is a meta-search engine. The category information can also be used for query modification (eg. a query for research papers on "*web search*" might be modified to include terms such as "*references*").

The second option for introducing context into Web search is to use a specialised search engine whose index has been designed to cover a restricted information domain, essentially fixing the context prior to searching. For example, CiteSeer [14], focuses on searching scientific literature, and DEADLINER [10] targets conference and workshop information.

2.2 Implicit Context

Since many users are unwilling to provide explicit context information, alternative approaches are needed. What if context could be automatically inferred? This question is being answered by a wide range of research focusing on different techniques for capturing different types of context. In fact two basic approaches have become

popular depending on whether *external* or *local* context sources are exploited.

Users rarely perform searches in isolation. It is much more likely that the search will be related to some other task that they are currently performing. Perhaps they are reading a Web page, replying to an email, or writing a document when they need to search for some associated piece of information. By taking advantage of a user's activity immediately prior to the search it may be possible to determine a suitable search context. This is the goal of systems such as Watson [4], the Remembrance Agent [17], and Letizia [15].

Watson and the Remembrance Agent provide just-in-time information access by deriving context from everyday application usage. For example, as a Watson user edits a document in Microsoft Word, or browses in Internet Explorer, Watson attempts to identify informative terms in the target document by using a heuristic term-weighting algorithm. If the user then searches with an explicit query, Watson modifies this query by adding these newly derived terms. Similarly, Letizia analyses the content of Web pages that the user is currently browsing, extracting informative keywords using similar term-weighting heuristics, and proactively searches out from the current page for related pages. In this sense, Letizia is more of a browsing assistant than a search assistant but it does exploit context in a similar manner; incidentally, Watson can also operate in this mode by continually searching the Web for related documents based on query terms extracted from the current document that the user is working on. [8] describes a method that uses categories from the Open Directory Project (ODP) (www.dmoz.org) as a source of context to guide a topic-sensitive version of PageRank [3]. Briefly, the URLs below each of the 16 top-level ODP categories are used to generate 16 PageRank vectors that are biased with respect to each category. These biased vectors are used to generate query-specific importance scores for ranking pages at query-time that are more accurate than generic PageRank scores. Similarly, for searches performed in context (eg. when a user performs a search by highlighting words in a Web page), context-sensitive PageRank scores can be computed based on the terms and topics in the region of the highlighted terms.

The above refer to the use of external sources of context. Techniques also exist for the exploitation of local sources of context. These techniques attempt to use the results of a search as the basis for context assessment, extracting useful context terms that can then be used to supplement the user's original query. Typically these context terms are those terms that are highly correlated in the initial search results. For example, the technique proposed by [16] extracts correlated terms from the top-ranking search results to focus context on the most relevant search results as opposed to the entire set. This idea of using the local search context can be extended beyond a single search episode. Many users will perform a sequence of searches on a specific topic and their response to the results can provide valuable context information. Thus, by monitoring and tracking queries, results and user actions it may be possible to model search context over an extended search session or even across multiple search sessions. For example [1] describes the SearchPad system which extracts context information, in the form of useful queries and promising result-lists, from multiple search sessions. Similarly, [2] describes the CASPER search engine for job advertisements, which maintains client-side user profiles that include job cases that users have liked and disliked in previous searches. These profiles are used to classify and re-rank the results of future searches. CASPER can learn that a given user is interested in Dublin software-engineering jobs that require more than 5 years experience because in the past they have liked job cases in the Dublin region and consistently avoided jobs with lower experience requirements.

3 Collaborative Search & I-SPY

Collaborative search is motivated by two key ideas. First, specialised search engines attract communities of users with similar information needs and so serve as a useful way to limit variations in search context. For example, a search field on an AI Web site is likely to attract queries with a computer-related theme and queries such as "*cbr*" are more likely to relate to Case-Based Reasoning than the Central Bank of Russia. Second, by monitoring user selections for a query it is possible to build a model of query-page relevance based on the probability that a given page p_j will be selected by a user when returned as a result for query q_i.

The collaborative search approach combines both of these ideas in the form of a meta-search engine that analyses the patterns of queries, results and user selections from a given search interface. This approach has been fully implemented in the I-SPY search engine and will be detailed and evaluated in the following sections.

3.1 The I-SPY System Architecture

The I-SPY collaborative search architecture is presented in Figure 1. It presents a meta-search framework in which each user query, q, is submitted to base-level search engines (S_1 - S_n) after adapting q for each S_i using the appropriate adapter. Similarly, the result set, R_i, returned by a particular S_i is adapted for use by I-SPY to produce R'_i, which can then be combined and re-ranked by I-SPY, just like a traditional meta-search engine. I-SPY's key innovation involves the capture of search histories and their use in ranking metrics that reflect user behaviour. The unique feature of I-SPY is its ability to person-

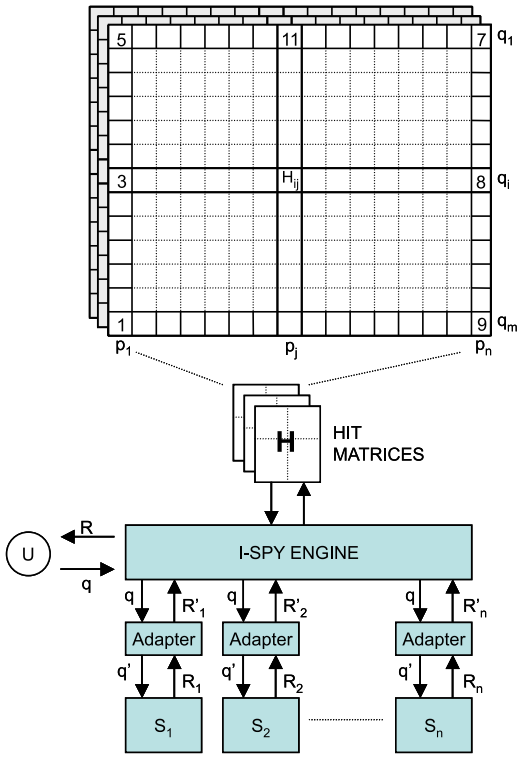


Figure 1. The I-SPY system architecture.

alize its search results for a particular community of users without relying on content-analysis techniques (eg. [2, 12]). To achieve this, I-SPY borrows ideas from collaborative filtering research to profile the search experiences of users. Collaborative filtering methods exploit a graded mapping between users and items and I-SPY exploits a similar relationship between queries and result pages (Web pages, images, audio files, video files etc.). This relationship is captured as a *hit matrix* (see Figure 1). Each element of the hit matrix, H_{ij}, contains a value v_{ij} (that is, $H_{ij} = v_{ij}$) to indicate that v_{ij} users have found page p_j relevant for query q_i. In other words, each time a user selects a page p_j for a query term q_i, I-SPY updates the hit matrix accordingly. I-SPY maintains its hit table using a relational database and an efficient encoding for result URLs and query terms.

3.2 Collaborative Ranking

I-SPY's key innovation is its ability to exploit the hit matrix as a *direct* source of relevancy information; after all, its entries reflect concrete relevancy judgments by users with respect to query-page mappings. Most search engines, on the other hand, rely on *indirect* relevancy judgments based on overlaps between query and page terms, but I-SPY has access to the fact that, historically, v_{ij} users have selected page p_j when it is retrieved for query q_i. I-SPY uses this information in many ways, but in particular the relevancy of a page p_j to query q_i is estimated by the probability that p_j will be selected for query q_i (see Equation 1).

$$Relevance(p_j, q_i) = \frac{H_{ij}}{\sum_{\forall j} H_{ij}} \quad (1)$$

Figures 2 and 3 show two screen-shots of the I-SPY system and serve as a simple example of the system's potential. Each presents part of the results page for a query by a computer science student for the single term query "*cbr*" (meaning "case-based reasoning"). Figure 2 shows the result-list returned before I-SPY has built-up its hit-matrix data and so the results are ordered using a standard meta-search ranking function, giving preference to results that are highly ranked by I-SPY's underlying search engines; in this case, Google, Yahoo, AllTheWeb, Teoma, WiseNut, HotBot and AltaVista. Clearly not all of the results presented are relevant and in fact only the 3rd result is on-target. Since "*cbr*" is a vague query it is no surprise that these results lack precision.

In contrast, Figure 3 shows the results for the same query, but after I-SPY has been *trained* by a community of computer science students; that is, the query-result patterns of a set of computer science students have been used to build the hit-matrix. The results are now ranked by I-SPY's relevance metric, as discussed above, rather than by the standard meta-search ranking function. The point is that this time the results are far more relevant; all of the top 3 results now refer to Case-Based Reasoning rather than other interpretations of the "*cbr*" query. For example, the third ranking result, *"Home Pages of ML and CBR Folks"* is a well known page listing CBR researchers and has an I-SPY relevance value of 12.5%. In other words, this page has been selected 12.5% of the times that *cbr* has been used as a query. This page previously would have been ranked in 7th position by the standard meta-search ranking function.

3.3 Community-Based Filtering

A key point to understand about this relevancy metric is that it is tuned to the preferences of a particular set of users - the community of I-SPY users - and the queries and pages that they tend to prefer. Deploy I-SPY on a wildlife Web site and its hit matrix will be populated with query terms and selected pages that are relevant to wildlife fans. Over time the value-space of the relevancy metric will adapt to fit the appropriate query-page mappings that serve this target community. For example, queries for "*jaguar*" will tend to result in the prioritisation of sites about the wild cat, as opposed to sites related to cars, because previously when users have submitted this query term they will have selected these wildlife sites. The other sites may still be returned but will be relegated to the bottom of the result-list. In fact I-SPY can deploy multiple I-SPY search agents,

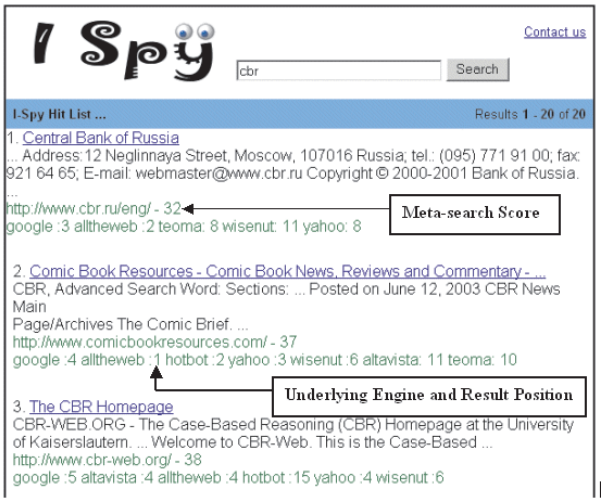


Figure 2. I-SPY search results before training

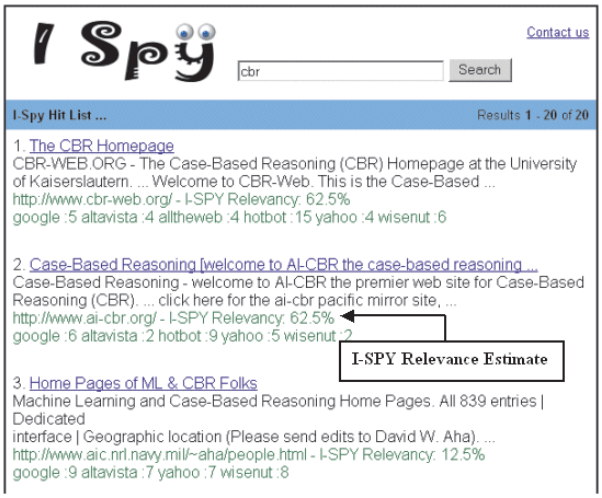


Figure 3. I-SPY search results after training

each with its own separate hit table. Thus the central I-SPY engine can be used to service many different search services across a range of portals, for example, each one adapted for the needs of a particular user group through its associated hit matrix. Alternatively, different hit matrices could be associated with different regions of the same site to bias search with respect to different topics. Placing a

search box on a "*programming languages*" directory page will naturally tend to capture queries from this domain. Consequently, the behaviour of the users providing these queries will gradually adjust I-SPY's relevancy metric and ranking function in favour of programming languages pages.

4 Evaluation

The basic premise of I-SPY is that it is possible to learn implicit search context information by monitoring the selection behaviour of users, and that it is possible to leverage this context information to re-rank standard search results in a useful way. In this experiment we use HotBot (www.hotbot.com) as the basic underlying search engine and we demonstrate how HotBot's raw results can be re-ranked by I-SPY as implicit context information is learned from the selection behaviour of search users. It is important to highlight that the evaluation is conducted by using an artificial model of user search behaviour. The artificial user model is informed by the real search behaviour of live users and since the results of this study are in broad agreement with recent live-user trials, we argue that this artificial user study is useful and informative.

4.1 Setup

The evaluation is conducted over four different search domains, each corresponding to a different subject area (*topic*) with a set of selected query terms and known context terms. In the following sections we describe these different domains plus the generation of query and context terms, the establishment of result relevance, and the role of an artificial user model to simulate user search behaviour.

4.1.1 Topic Domains & Query Generation

We focus on four distinct topic domains, each of which roughly corresponds to a community of Internet users that are interested in a particular subject or topic area. For each domain a set of sample queries are generated; these are the *raw* or uncontextualised query terms. In addition, for each domain we agree on a set of *context terms* which, when combined with the raw query terms, provide a set of *contextualised* queries. For example, in the "*programming languages*" topic domain we generate 74 raw query terms (eg. "*java*", "*pascal*", "*perl*", etc.) from which we derive 74 contextualised queries (eg. "*programming language java*", etc.).

- **Mammals**
  No. of Queries: 211
  Type of Queries: Names of mammals
  Source: Mammals subdirectory in Yahoo.
  Context: "mammal"
- **Travel**
  No. of Queries: 202
  Type of Queries: Country names
  Source: Family Education Network's Countries of the World page.
  Context: "travel"
- **CBR and ML Researchers**
  No. of Queries: 69
  Type of Queries: People involved in CBR and ML research.
  Source: David W. Aha's CBR and ML Researchers page.
  Context: Affiliation (eg. "*University College Dublin*").
- **Programming Languages**
  No. of Queries: 74
  Type of Queries: Names of programming languages
  Source: Programming Languages subdirectory in Yahoo.
  Context: "programming language"

4.1.2 Establishing Relevance

Each query term is used to generate two lists of search results from HotBot. The first list, called the *raw results*, corresponds to the results returned by HotBot for each of the raw queries; HotBot returns up to 1000 results per query. The second list, called the *context results*, corresponds to the results returned by HotBot for the contextualised queries.

The essential point is that, for the purpose of our evaluation, the context results are assumed to be those results that are actually *relevant* to the user. For example, consider the query "*jaguar*" in the "*mammals*" domain. The raw results from HotBot (arising from the "*jaguar*" query) contain a diverse set of results including pages that are related to cats, cars, and operating systems. The results returned for the query "*mammal jaguar*" are assumed to be relevant for this topic domain and make it possible to identify a subset of the raw HotBot results as relevant to the user. Thus, for each list of raw results we have a way of identifying which of these results are likely to be relevant to a user searching in a given topic domain.

4.1.3 A User Selection Model

Whether a user is likely to select a search result in a given search session depends on whether the result is relevant, but also on the position of the result in the result-list (earlier/higher results are far more likely to be selected than later/lower results). Our user selection model is

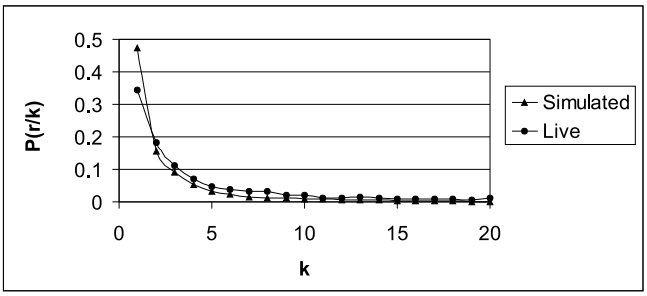


Figure 4. Observed user search behaviour shows a sharp decline in selection probability with increasing result position (k). The artificial selection model is tuned to closely match this selection behaviour.

informed by the search behaviour of 179 real users, observed over a period of 8 weeks and approximately 1500 search sessions. From this data we are able to model the probability that a given user is likely to select a relevant search result r given that it is ranked in position k in the result-list; we assume that the majority of user selections are for relevant results. For example, Figure 4 illustrates this probability distribution and indicates that users are very likely to select relevant results that occur in the top 3 to 5 positions, but that this probability quickly degrades with increasing result position. Our user selection model also includes a small random component to allow for the selection of irrelevant results by users, and during each search session the artificial user is limited to the selection of a predefined number of results, in this case 3 ± 3 results, as informed by our live-user data.

4.1.4 Methodology

Our evaluation is carried out in the following way. For each topic domain, each query is submitted to HotBot between 100 and 200 times. Each raw result-list is processed, using our user model, to simulate user selections based on those results that are known to be relevant for the target query (according to the context results). These simulated user selections are used to populate an I-SPY hit-matrix for the appropriate topic domain. The outcome is a hit-matrix for each topic domain based on the selection behaviour of users in a given search context.

Next, the queries are re-run (again between 100 and 200 times) but this time the result-lists are re-ranked by I-SPY, using the appropriate hit-matrix to drive I-SPY's collaborative ranking engine. We calculate the precision and recall characteristics for these re-ranked I-SPY results, for different levels of k (result-list size), based on the known relevant-results data. Comparable precision and recall values are also computed for the raw results from HotBot as a benchmark.

4.2 Results

Figure 5 presents the results for each of the four domains, for both I-SPY and HotBot, as a graph of precision versus recall for each result-list size (k=5-150). For clarity, the graphs have been partially annotated to indicate the k-values for individual data-points.

Overall the results demonstrate that there is a significant benefit to be derived from I-SPY's collaborative search technique - I-SPY's contextualised, re-ranked results have significantly higher precision and recall values (for a given k) than HotBot's original results. For example, for $k = 5$, the "*mammals*" results (see Figure 5(a)) indicate that I-SPY delivers a precision of more than 0.8 and a recall of 0.12, as compared to a precision of less than 0.2 and recall of approximately 0.01 for HotBot. In other words, in this domain, for I-SPY, approximately 4 out of the top 5 results are relevant. For HotBot, on average, only about 1 out of the top 5 results are likely to be relevant.

We find that precision tends to fall for I-SPY for increasing values of k. This is to be expected because many relevant results will occur low down in the original HotBot results lists and so have a low probability of being selected by users. Such results, although relevant, are unlikely to make it into the hit-matrix. The precision values for I-SPY and HotBot tend to converge at about $k = 100$ indicating that most of I-SPY's promoted results were originally ranked within the top 100 HotBot results. Perhaps the most important feature of these precision results is that I-SPY's maximum benefit tends to occur at low values of k. This is particularly important, and useful, in the context of Web search, and other consumer search applications (eg. mobile-phone search applications) where only limited-size result-lists can be presented. On average the recall results are low, as expected, especially for low values of k. For instance, in the "*mammals*" domain there are an average of about 30 relevant results per query, so the maximum recall at $k = 5$ is 0.166 (that is, $\frac{5}{30}$). I-SPY achieves more than 75% of this maximum recall value at $k = 5$, whereas HotBot achieves only 6% of this maximum. Similarly, at $k = 10$, there is a maximum recall of 0.33 and I-SPY achieves nearly 47% of this (0.155 recall) compared to HotBot, which again reaches only 6% of this (0.02 recall).

In the "*mammals*" and "*researchers*" domains, I-SPY's recall characteristics begin significantly ahead of HotBot's. For example, I-SPY achieves a recall of 0.27 in the "*researchers*" domain for $k = 5$. HotBot only achieves a recall of 0.06 for this k value, and in fact requires the retrieval of about 50 results to match I-SPY's recall. Why

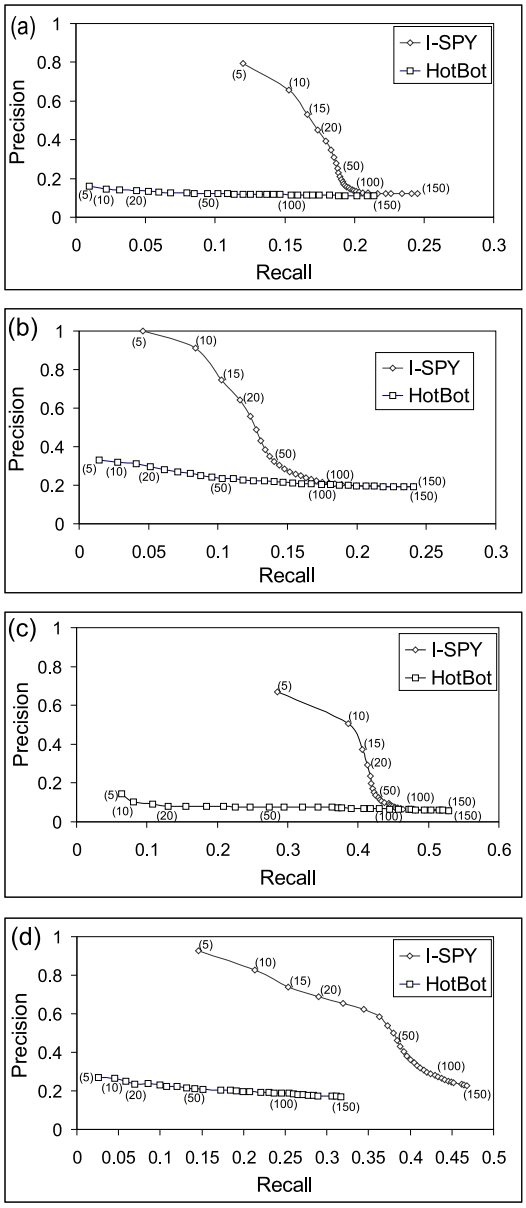


Figure 5. Precision vs. Recall for domain: (a) Mammals, (b) Travel, (c) CBR and ML Researchers, (d) Programming Languages.

do these two domains offer I-SPY improved recall from the outset? Both domains are characterised by a high level of ambiguity in their raw queries, leading to lower numbers of relevant results from the outset. We can estimate query ambiguity in terms of the average number of relevant results per query in the raw result-lists; if all of the raw results are relevant then the raw queries are not ambiguous, but if very few raw results are relevant then the raw queries must have high ambiguity. For example, in the "*mammals*" domain, on average only 3% of results per query are relevant and for "*researchers*" only 1.2% of results per query are relevant. This is in contrast to 10% and 18% of results per query being relevant for the "*travel*" and "*programming languages*" domains, respectively. Thus there is a strong negative correlation between I-SPY's recall and the number of relevant results per query and thus I-SPY's benefits are likely to increase with the level of ambiguity in a typical user query.

4.3 Discussion

In summary, we have shown that I-SPY benefits from improved precision and recall characteristics when compared to HotBot. In particular, I-SPY enjoys vastly superior precision characteristics (>x2) for result lists up to $k = 20$. In addition, I-SPY can also achieve significantly superior recall, especially for ambiguous queries. Of course,

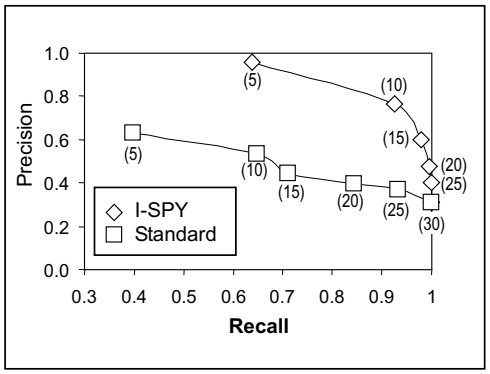


Figure 6. Precision vs. recall for live-users in the "programming languages" domain.

the above evaluation has been conducted using an artificial model of user search behaviour, and although this model has been designed with reference to the search behaviour of real users, doubts will naturally remain. In response it is worth drawing attention to an earlier evaluation of I-SPY ([19]) that does take advantage of live-user behaviour. The results of this earlier study are in broad agreement with the results presented here. For example, Figure 6 presents the precision-recall graph produced from this earlier study and the level of agreement with the current results should be clear. If such agreement was not present then there would be grounds to question the evaluation approach taken here, or at the very least, the user model used to simulate search behaviour. The fact that the results are so similar suggests that the user model is reasonable and the results valid.

5 Conclusions

The collaborative search idea attempts to discover patterns in the activity of a community of searchers in order to determine general search context and prioritise search results accordingly. It makes no strong assumptions about the form of the underlying search engines and is generally applicable across a range of content types. The proposed ranking metric is computationally efficient ($O(k)$ in the number of search results) and requires no additional parsing of result pages. Finally, the ability to personalize search results for the needs of a community is achieved without the need to store individualised search histories; no individual user profiles are stored and no user identification is necessary. This has significant security and privacy advantages compared to many more traditional approaches to personalization.

In this paper we have described and evaluated the I-SPY implementation of collaborative search. The results indicate a clear potential for significant precision and recall improvements, when compared to traditional Web search engines, highlighting the potential for collaborative search to add value to existing Web search engines. Moreover, the benefits are particularly significant for low values of k (result-list size), which in turn suggests that the collaborative search approach is especially well suited for search on devices such as mobile phones and PDAs, with their limited display and input capabilities. Future research will focus on a number areas including: the ongoing, live-user evaluation of I-SPY; data-mining the hit-matrix to generate page similarity and query similarity knowledge; leveraging user feedback for index revision; the role of I-SPY in multi-media search applications.

REFERENCES

[1] K. Bharat, 'SearchPad: Explicit Capture of Search Context to Support Web Search.', in *Proceedings of the Ninth International World-Wide Web Conference*, (2000).

[2] K. Bradley, R. Rafter, and B. Smyth, 'Case-based User Profiling for Content Personalization', in *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, eds., O. Stock P. Brusilovsky and C. Strapparava, pp. 62–72. Springer-Verlag, (2000).

[3] S. Brin and L. Page, 'The Anatomy of A Large-Scale Web Search Engine.', in *Proceedings of the Seventh International World-Wide Web Conference*, (2001).

[4] J. Budzik and K. Hammond, 'User Interactions with Everyday Applications as Context for Just-In-Time Information Access.', in *Proceedings International Conference on Intelligent User Interfaces.* ACM Press, (2000).

[5] D. Dreilinger and A. Howe, 'Experiences with Selecting Search Engines Using Meta Search', *ACM Transactions on Information Systems*, **15(3)**, 195–222, (1997).

[6] E. Glover, S. Lawrence, M. D. Gordon, W. P. Birmingham, and C. Lee Giles, 'Web Search - Your Way', *Communications of the ACM*, (2000).

[7] Eric J. Glover, Gary W. Flake, Steve Lawrence, Andries Kruger, David M. Pennock, William P. Birmingham, and C. Lee Giles, 'Improving category specific web search by learning query modifications', in *2001 Symposium on Applications and the Internet (SAINT 2001)January 08 - 12, 2001 San Diego, CA*, p. 23, (2001).

[8] T. H. Haveliwala, 'Topic-Sensitive PageRank', in *Proceedings of the World-Wide Web Conference.* ACM Press, (2002).

[9] Jon M. Kleinberg, 'Authoritative Sources in a Hyperlinked Environment.', in *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 668–677, (1998).

[10] A. Kruger, C Lee Giles, F. Coetzee, E. Glover, G. Flake, S. Lawrence, and C. Omlin, 'Building a New Niche Search Engine.', in *Proceedings of the Ninth International Conference on Information and Knowledge Management.*, (2000).

[11] S. Lawrence, 'Context in Web Search', *IEEE Data Engineering Bulletin*, **23(3)**, 25–32, (2000).

[12] S. Lawrence and C. Lee Giles, 'Context and Page Analysis for Improved Web Search', *IEEE Internet Computing*, **July-August**, 38–46, (1998).

[13] S. Lawrence and C. Lee Giles, 'Accessibility of Information on the Web.', *Nature*, **400(6740)**, 107–109, (1999).

[14] S. Lawrence and C. Lee Giles, 'Searching the Web: General and Scientific Information Access', *IEEE Communications*, **37(1)**, 116–122, (1999).

[15] H. Lieberman, 'Letizia: An Agent That Assists Web Browsing', in *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI'95*, ed., C. Mellish, pp. 924–929. Morgan Kaufman Publishers, (1995). Montreal, Canada.

[16] M. Mitra, A. Singhal, and C. Buckley, 'Improving Automatic Query Expansion.', in *Proceedings of ACM SIGIR.* ACM Press, (1998).

[17] B. J. Rhodes and T. Starner, 'Remembrance Agent: A Continuously Running Automated Information Retrieval System.', in *Proceedings of the First International Conference on the Practical Applications of Intelligent Agents and Multi-Agent Technologies.*, pp. 487–495, (1996).

[18] E. Selberg and O. Etzioni, 'The Meta-Crawler Architecture for Resource Aggregation on the Web.', *IEEE Expert*, **Jan-Feb**, 11–14, (1997).

[19] B. Smyth, E. Balfe, P. Briggs, M. Coyle, and J. Freyne, 'Collaborative Web Search.', in *Proceedings of the 18th International Joint Conference on Artificial Intelligence, IJCAI-03*. Morgan Kaufmann, (2003). Acapulco, Mexico.

Relevance at a Distance - An Investigation of Distance-Biased Personalization on the Mobile Internet

Barry Smyth, Kevin McCarthy & James Reilly
Smart Media Institute
Department of Computer Science
University College Dublin
Belfield, Dublin 4, Ireland
barry.smyth@ucd.ie

Abstract. The usability of mobile portals has been a major stumbling block since the advent of the mobile Internet and WAP handsets. Indeed poor usability is cited as a major contributing factor to the poor take-up of mobile Internet services amongst consumers. A key problem relates to the amount of time that users spend navigating to content as they browse mobile portals. Recent advances in personalization technology have the potential to solve this problem, and today a number of leading operators already provide their users with access to intelligent portals that are automatically personalized based on subscriber usage patterns. In this paper, we examine this so-called *personalized navigation* technology and propose ways in which it may be enhanced by combining structural properties of a mobile portal (such as the distance to content sites) with the access probabilities of users. We demonstrate that although such distance factors have proven successful in Web personalization, they are less beneficial when it comes to the personalization of mobile portals.

1 Introduction

The mobile Internet (MI) has failed to live up to end-user expectations. Limited bandwidth, unreliable handsets, patchy content and poor usability have all contributed to this state of affairs. And although recent developments have seen significant improvements in bandwidth, handsets and content, usability remains a problem, particularly in relation to the navigation effort faced by users when searching for content in a typical mobile portal. For example, recent studies have highlighted how content services are usually positioned to be more than 16 *clicks* from the portal home page. In other words, to access a typical content service a user can expect to have to make 16 clicks on their mobile phone as they navigate through the portal, scrolling through menus and selecting options enroute [13]. The result is low levels of satisfaction from end-users and lackluster usage levels for mobile operators.

Recently, however, a compelling solution has emerged that has been proven to have a dramatic impact on portal usability by significantly reducing the above navigation problem. This so-called *personalized navigation* solution applies personalization techniques to the navigation task. That is, instead of recommending individual content items to users, menus and menu options are recommended in such a way that users require an average of 50% fewer clicks to locate content, leading to significant increases in mobile usage [15, 16]. Very briefly, this personalized navigation approach estimates $P_u(o|m)$, the probability that a given user, u, currently in menu m, is looking for menu, o. Menu options are promoted to the user based on their past access probabilities.

In this paper we focus on this probabilistic personalized navigation technique in the context of mobile portal personalization. In particular, we investigate whether or not performance improvements can be achieved by extending the basic probabilistic personalization model to take into account an options's distance from the current menu as well as its access probability. For example, consider two menu options, o_1 and o_2, both with the same access probabilities; that is, $P_u(o_1|m)$=$P_u(o_2|m)$. But suppose that the distance from m to o_1 is greater than the distance from m to o_2, that is $Distance(m, o_1) > Distance(m, o_2)$, then shouldn't o_1 be promoted ahead of o_2 because if correct a greater number of navigation clicks will have been saved? We call this *distance-biased promotion* and it clearly has the *potential* to improve the degree to which personalized navigation can save a user navigation effort. In fact evidence from Web personalization suggests that such an extension is likely to pay dividends [1]. However, we are conscious that traditional Web-based portals and current mobile portals are very different and what works on the Web does not always translate well to the mobile space. With this in mind, in this paper we examine the potential of distance-biased techniques in the personalization of mobile portals.

In the next section we discuss the background to this research focusing in particular on recent developments in the mobile Internet and outlining past research related to the issue of navigation effort. Section 3 provides a review of our core personalized navigation strategy as detailed in [15, 16]. It also describes the click-distance model of navigation effort and explains how this can be used to bias personalized navigation with respect to portal distance. Finally, before concluding, in Section 4 we describe a recent evaluation to investigate the benefits of this distance-biased technique, based on a large-scale European portal and live user activity logs.

2 The Mobile Internet

The mobile Internet refers to the delivery of data services across wireless networks for Internet-enabled handsets as implemented through a group of related infrastructure, protocol and device technologies. It allows the end-user to access various types of data services from their mobile handsets, including Web-style information content, email services, games etc. Access devices range

from limited, first-generation WAP (Wireless Application Protocol, see www.wapforum.org) phones to today's sophisticated PDAs (Personal Digital Assistants) and so-called SmartPhones (see www.microsoft.com/smartphone).

In the past the usability of mobile services has been compromised by limited device functionality, bandwidth, and content. Fortunately the new generation of mobile services (so-called 2.5G services) represents a significant improvement. The major bandwidth and content issues have largely been resolved, and the latest phones offer users significant interface and functionality improvements over earlier models. However, key portal usability problems remain, due to poor mobile portal design. Users find that they are spending too much of their time navigating to content because mobile portals are designed as fixed, complex hierarchies of menu options.

2.1 Mobile Internet Devices

From a user experience viewpoint, one of the key features of the mobile Internet is the degree to which existing consumer devices represent a significant step backwards in terms of their functionality, at least when compared to the traditional Internet device (the desktop PC or laptop). In particular, presentation and input capabilities tend to be extremely limited on most mobile devices. For instance, a typical desktop PC, with a screen size of 1024x768 pixels, offers more than 10 times the screen real-estate of a PDA, and more than 20 times the screen space of second-generation Internet phones (eg. I-mode and Vodafone Live! handsets or Microsoft's SmartPhone). Mobile handsets are further limited in their ability to receive user input. The keyboard and mouse functionality of a modern PC are notably absent and the mobile phone numeric keypad makes it extremely difficult for user to input any quantity of information. From a mobile Internet viewpoint, these devices restrict selection features to simple scroll and select keys that allow the user to scroll through menu lists and perform selections. Some improvements are present in most PDAs, which tend to offer touch sensitive screens that are easier to manipulate. Nevertheless data input remains difficult at best.

2.2 Mobile Information Access

These key differences that exist between mobile handsets and more traditional Internet devices, such as PCs and laptops, directly influence the manner in which users access information using these devices. For example, on the Internet today search has largely become the primary mode of information access. It is relatively easy for users to input search queries and search engines have improved significantly in their ability to respond intelligently to user needs. In addition the large screen sizes make it feasible for users to efficiently parse the longs lists of search results returned. In contrast, search is far more problematic on mobile devices. Entering queries is simply too time consuming and complex for the average user to tolerate and small screen sizes make it practically impossible for users to easily process the result lists returned. As a result, browsing is the primary mode of information access on the mobile Internet. Instead of searching for information, users attempt to navigate to information by using mobile portals. Today the vast majority of mobile Internet services are accessed via an operator portal with direct search constituting a small fraction ($<10\%$) of mobile Internet activity.

This distinction between alternative modes of information access on the mobile and fixed Internet is an important one and it sets the scene clearly for our own research. The bottom line is that to help users to locate information and services more effectively on the mobile Internet we must attempt to improve the efficiency of mobile portal browsing or navigation.

2.3 Mobile Portal Navigation

Mobile portals are examples of hierarchical menu systems (HMS), and long before the arrival of the mobile Internet different forms of hierarchical menu systems were studied extensively with respect to their general usability and navigation characteristics [4, 5, 6, 7, 8, 17, 18, 19, 20]. Very briefly, much of this research has focused on the structural properties of hierarchical menu systems, for example their depth and width, as they relate to the ability of a user to easily navigate through the HMS. For example, [8] discovered that for moderate sized menu systems, wide hierarchies are preferable to deep hierarchies due to the short-term memory limitations of end users, which led to a greater number of navigation errors in deep hierarchies; the interested reader is also referred to [5, 17] for related work. Similar observations have been made with respect to the menu hierarchies found in the World-Wide Web [20]. Thus the evidence suggests that the complexity of a hierarchical menu system has a significant impact on its usability and the ability of users to navigate through menu levels. The type of menu hierarchies found on the mobile internet are likely to be subject to similar findings. Indeed the scale of the navigation problem associated with mobile portals today, and the mismatch between user expectations and realities, is highlighted by a number of recent studies. For example, one study claims that while the average user expects to be able to access content within 30 seconds, the reality is closer to 150 seconds [11]. For instance, Figure 1 presents a typical navigation scenario in which a mobile user must navigate through 4 levels of menus, and make 11 separate scrolls, in order to get from the portal home page to her local cinema listings. Of course the time that it takes a user to access a content item is a useful measure of navigation effort and we suggest that the navigation effort associated with an item of content depends critically on the location of that item within the portal structure, and specifically on the number of navigation steps (scrolls and selects) that are required in order to locate and access this item from a given starting position within the portal (typically the portal home page). We will return to this idea in the next section when we introduce the *click-distance* model of navigation effort. We show how it can be used to guide and evaluate the personalization of a portal.

3 A Probabilistic Model of Personalized Navigation

The basic idea behind personalized navigation is that instead of presenting each user with a fixed portal hierarchy, each user is presented with a hierarchy that has been adapted to his or her needs. By *adapted* we mean that individual menu options may be promoted within the portal so that they are more accessible to relevant users. For example, menu options may be reordered within a menu or they may even be promoted from lower levels of the portal to higher levels [1, 2, 3, 9, 10, 12, 14, 16, 15]. In other words, each time a user accesses a given menu, m, this menu is dynamically created given their short and long-term preferences and menu options or content items that the user is likely to be interested in are promoted.

In the following sections we describe a probabilistic model of personalized navigation that drives promotion by computing access probabilities for each user given their current menu. This serves as a review of our recent research and is the benchmark against which

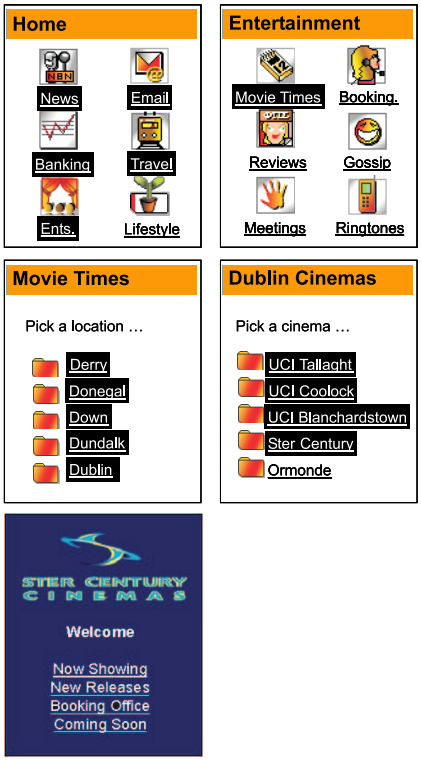


Figure 1. In this sample portal the user must navigate through a series of menu pages to locate their local cinema.

we propose to judge the usefulness of a modified approach to personalized navigation that also considers the distance of items from the current menu when selecting promotion candidates (see also [1]).

3.1 Profiling & Personalization

Tracking user accesses across a mobile portal provides the basis for an effective profiling mechanism. For example, individual menu accesses can be stored in a so-called hit-table, which provides a snapshot of a user's navigation activity over time. For example, Figure 2(a) indicates that a user has accessed option B from menu A 10 times and option C 90 times; of course in reality other activity information including device, temporal and location information is normally stored as part of this evolving profile but a more detailed discussion is outside of the scope of this paper.

In fact two types of hit table can be used: a global, *static* hit table that is initialized with respect to the default portal structure (Figure 2(b)); and a *user* hit table that records each user's individual history. The static table makes it possible to deliver a default menu structure early on that will be over-ridden by the personalized menu once a user's access probabilities build. Moreover, the hit values set in the static table make it possible to control personalization latency - low values mean that personalization takes effect very quickly. To build a personalized menu m for user u we must identify the k most probable options for m (the k options with the highest $P_u(o|m)$ values) by combining the frequency information in the user and static hit tables. Consider the data in Figure 2 and the construction of menu A. The access probabilities can be determined as shown in Figure 3. In descending order of access probability we have C, F, B, G, D, and E. For $k = 3$, C, F, and B are selected, in order, for menu A.

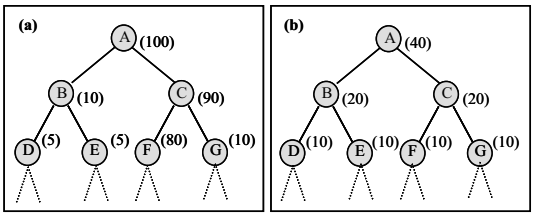


Figure 2. User (a) and static (b) hit-table representations.

P(B|A)		= (20+10)/(40+100)	.214
P(C|A)		= (20+90)/(40+100)	.786
P(D|A)	= P(B|A) P(D|B)	= (30/140)(10+5)/(20+10)	.107
P(E|A)	= P(B|A) P(E|B)	= (30/140)(10+5)/(20+10)	.107
P(F|A)	= P(C|A) P(F|C)	= (110/140)(10+80/20+90)	.642
P(G|A)	= P(C|A) P(G|C)	= (110/140)(10+10)/(20+90)	.142

Figure 3. Sample access probabilities; note that the user subscript has been omitted for simplicity.

The complexity of the proposed personalization method depends on the complexity of the process that identifies the k most probable options for the menu, m. As described this can mean examining not just the default options of m, but also all the options contained in menus that are descendents of m; essentially a breadth-first search from m to the content leaves of the menu tree is required. Fortunately, a more efficient algorithm is possible once we recognize that, by definition, $P_u(o|m)$ is always greater than or equal to $P_u(o'|m)$ where o' is an option of a menu, m', which is itself a descendent of m through o. This means that we can find the k most probable nodes for menu m by performing a depth-limited, breadth-first search over the menu tree rooted at m. We only need to expand the search through an option o' if $P_u(o'|m)$ is greater than the kth best probability so far found. Once again, a detailed description of this issue is beyond the scope of the current paper but the interested reader is referred to [16] for further information.

The approach just described supports two types of menu adaptations: (1) a menu option may be *reordered* within its parent menu by changing its position within its parent menu; or (2) a menu may be *promoted* into an ancestral menu. Such adaptations are side-effects of the probability calculations. In the above example, option F is promoted to A's menu - options can even be promoted from deeper levels if appropriate. If F is subsequently selected from A, it is added to A's hit table entry for that user, so the next time that A is created, the computation of $P_u(F|A)$ must account for the new data on F. Specifically, assuming a single access to F as an option in A, we get:

$$P_u(F|A) = 1/101 + (110/141)(10 + 80)/(20 + 90) = 0.647$$

3.2 Distance-Biased Promotion

The above approach to personalized navigation has been fully developed and deployed in the field through the ClixSmart Navigator^{TM} product by ChangingWorlds (www.changingworlds.com). Large scale deployments by Vodafone and O_2 have attracted millions of mobile subscribers and carefully controlled evaluations prove that this approach to personalization can lead to dramatic improvements in the practical usability of a mobile portal. This in turn can lead to significant increases in mobile portal usage; for example, airtime,

user sessions, and page impressions have all be shown to increase as a direct result of ClixSmart Navigator's personalized navigation solution (see [15, 16]).

Thus there appears to be a compelling link between the practical usability of a mobile portal and the $distance$ that a user must travel in order to access its content; reducing this distance improves usability and drives usage. Therefore it is reasonable to seek out ways of reducing this navigation distance even further. For example, as it stands the above personalization technique is based on access probabilities alone and does not take navigation distance into account in any explicit way; although obviously navigation distance is reduced as a *side-effect* of the personalization process. By considering access probabilities *and* the likely reduction in navigation distance we can further reduce navigation distance as a side-effect of promotion and personalization (see also [1] for related work on the creation of *shortcut* links between Web pages).

3.2.1 Click-Distance

How can navigation distance be usefully measured? [1] suggest a simple model of navigation distance that counts the number of links that must be followed to locate a page in a Web site. However, while this simple model is appropriate in the context of more traditional Web sites, it is not well suited to modern mobile portals. With the current generation of mobile phones, there are two basic types of navigation action. The first is the *menu select*: the user clicks to select a specific menu option. The second is a *menu scroll*: the user clicks to scroll up or down through a series of options. Scroll actions are less important in the context of traditional Web portals and traditional Web access devices such as PCs, laptops and PDAs, with their sophisticated point-and-click user input. In contrast, the input capabilities of most Internet-enabled mobile phones are far more limited and even simple scrolling actions correspond to a significant degree of user effort.

Thus, an item of content, i, within a mobile portal can be uniquely positioned by the sequence of selects and scrolls needed to access it, and the navigation effort associated with this item can be simply modelled as click-distance, the corresponding number of these selects and scrolls (see Equation 1).

$$ClickDistance(i) = Selects(i) + Scrolls(i) \quad (1)$$

Recent studies illustrate the extent of the click-distance problem. For example, a recent analysis of 20 European mobile portals reported an average click-distance in excess of 16 [13]; see Figure 4. In other words, a typical European mobile user can expect to have to make 16 or more clicks (scrolls and selects) to navigate from their portal home page to a typical content target. Moreover, on average European portals are organised such that less than 30% of content sites are within 10-12 clicks of the portal home page; 10-12 clicks corresponds to a navigation time of about 30 seconds, which is expected by mobile Internet users [11]. To put this another way: more than 70% of mobile portal content is essentially invisible to users because of its positioning within its parent portal. Finally it is worth highlighting that although the above click-distance model constitutes a fairly simple model of navigation effort it is nonetheless an effective one in practical terms. For example, a recent analysis of the activity of 3500 users of a major European portal, over the course of a 30-day period in 2002, found a correlation of -0.65 between the click-distance of content-sites (that exist as leaf nodes in the portal hierarchy) and their access frequencies. In other words, all other things being equal, sites with a low click-distance are accessed more frequently than sites with a high click-distance. In one sense of course this result is fairly intuitive, nevertheless the strength of the correlation is significant.

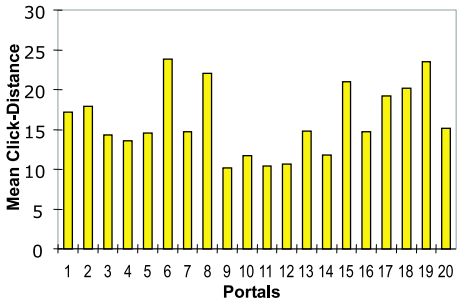


Figure 4. Mean click-distances for 20 European mobile portals.

3.2.2 Expected Click-Distance

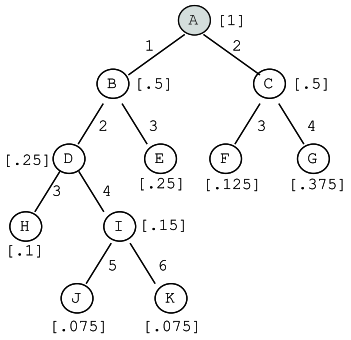
At this stage it is possible to combine the click-distance of a portal item i and its access probability to calculated the *expected click-distance* (ECD) of i. Then, in order to promote those items that have the largest expected click-distances, because such promotions are likely to result in the greatest expected click-distance savings (see also [1]). As a result, during personalization, instead of simply computing the access probabilities for descendants of the current menu, m, we calculate the expected click-distances of these descendants according to Equation 2 (the expected click-distance of item i from menu m, for some user u).

$$ECD_u(i, m) = ClickDistance(i) * P_u(i|m) \quad (2)$$

Figure 5 illustrates a simple example of this concept. A small portal, rooted at home page A, is presented and the expected click-distance values for all of the descendants of A are calculated. For example, menu G has the largest expected click-distance (1.5) based on an access probability of 0.375 and a click-distance from A of 4 (3 scrolls and 1 select). In fact, G has the highest expected click-distance and so would be promoted into menu A ahead of even B or C, A's default descendants. Similarly E would be promoted ahead of B as it has an expected click-distance of 0.75 compared to B's expected click-distance of 0.5. As a result, if we assume that menu A has been configured to allow for 5 options then, in order of their expected click-distances, options G, C, E, I, B would be selected. In contrast, on the basis of access probabilities alone, options B, C, G, E, D would have been selected in order. Note that Equation 2 assumes that there is only one path from m to i. Of course in general there can be multiple paths from m to i each with their own access probabilities and click-distance from m. As such, in practice, the expected click-distance must be summed over these alternate paths in the obvious way.

4 Experimental Evaluation

In previous work we have reported widely on the results of extensive live-user trials of our personalized navigation techniques [15, 16]. These trials prove a strong link between click-distance reduction and



Item, i	CD(i)	$P_u(i \mid A)$	$CD(i)*P_u(i \mid A)$
G	4	0.375	1.5
C	2	0.5	1
E	3	0.25	0.75
I	4	0.15	0.6
B	1	0.5	0.5
D	2	0.25	0.5
K	6	0.075	0.45
F	3	0.125	0.375
J	5	0.075	0.375
H	3	0.1	0.3

Figure 5. A sample portal showing access probabilities (in square brackets) and item click-distances from the portal root. The table shows the expected click-distances for each of the descendants of A in descending order.

increased portal usability. In this paper we are interested in evaluating the likely impact of biasing our personalization technique to include a distance factor as well as its core access probability factor. Ultimately we are interested in understanding if the above distance-biased promotion technique is likely to result in an increased click-distance reduction (compared to the pure probabilistic approach) in real mobile portals. If increased click-distance reductions are proven then this bodes well for the new distance-biased approach because these greater reductions are likely to result in further (or at least more rapid) usability improvements.

4.1 Setup

For the purpose of this experiment we used data from a leading European mobile portal. Thus included the portal structure containing over 450 portal nodes and 14-days worth of user access logs covering the activity of 3,500 individual users. We also made use of two versions of ClixSmart Navigator: the standard version that relies on a pure probabilistic approach to promotion and personalization; and an enhanced version that incorporates the above distance-biased approach.

4.2 Comparative Click-Distance Profiles

The key question to answer is whether there is any significant difference in the click-distance reduction obtained using the distance-biased method when compared to the reduction obtained using the pure probabilistic method. To test this we used the supplied user logs to $replay$ the user activity over two versions of the portal: one that is personalized by the pure probabilistic strategy and one that is personalized according to the distance-biased strategy; incidentally, the logs were replayed using ClixSmart Navigator's simulation engine. This allowed us to evolve two different portals over the 14 day test period: a pure probabilistic portal and a distance-biased portal. At the end of each simulation "day" we calculated the mean click-distance of the two portals, averaged over all user sessions that occurred during that day.

The results are presented in Figure 6 as a graph of mean click-distance against simulation day for each of the two portals corresponding to the two different personalisation strategies. The results clearly show the click-distance reduction capability of each strategy. At the end of the first day both portals have an average click-distance of between 8.4 (distance-biased) and 8.6 (pure probability) and by the end of the 14th day this has dropped to about 5.3. In other words, to begin with it takes users more than 8 clicks to get to a typical content site from their portal home page. Remember, these 8 clicks are made up of a combination of scrolls and selects and in this portal the ratio is over 3 to 1, so on average 2 of these 8 clicks will be menu selections and the remaining 6 will be scrolls. After only two weeks users are able to access content sites in about 5 clicks (approximately 1 menu select and 4 scrolls), an overall click-distance reduction of 40%. Perhaps the most important thing to note is the

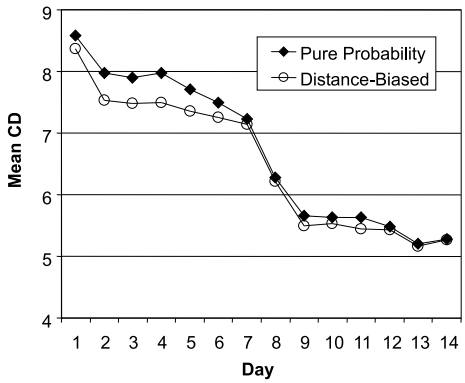


Figure 6. Click-distance results.

lack of any significant difference between the click-distance profiles of the two portals, at least beyond day 6. In other words, although the distance-biased technique is capable of delivering improved click-distance reductions, this benefit is relatively short-lived and appears to disappear after day 6. Moreover, the extent of this improvement for the first 6 days is marginal at roughly 0.5 clicks. The average click-distance for the pure probabilistic portal is 8, over the first 6 days, compared to 7.5 for the distance-biased portal; that is, the pure probabilistic portal suffers from a 7% increase in click-distance when compared to the distance-biased portal.

4.3 Further Analysis

On the face of it then there appears to be relatively little advantage in biasing promotion using distance factors. Why should this be the case? One possibility is hinted at by the actual average click-distances reported in the above results. The evaluation portal contains more than 450 individual nodes and has an average click-distance in excess of 15 across all of its content sites; that is, 15 is the average click-distance from the portal home page to each of the content sites that exist at the leaf nodes of the portal. Nevertheless, the average click-distance reported above, which is based on actual user sessions

rather than a static click-distance analysis of the entire portal, is no more than 8. In other words, although many sites within the portal exist at very large click-distances from the portal home page, the majority of users actually never wander very far from the home page in a typical session. In fact when we further analysed the user logs we found that more than 80% of content accesses were for content sites that were within a click-distance of 10 from the portal home page, sites that were linked to from level 1 or level 2 pages.

This observation has two important implications. First, it means that many of the content sites accessed by users have similar click-distance to begin with and this limits the impact of the distance factor during personalization. For example, the 80% of accesses referred to above are for content sites with an average click-distance of 5.2 and a standard deviation of 3.4. Secondly, and perhaps more importantly, because these sites were positioned on level 1 or level 2 pages the distance that they could be promoted was also limited from the start. For example, if a site is linked to from a level 1 menu then it can only be promoted to the portal home page (level 0) and similarly level 2 pages can only be promoted to level 1 or level 2 menus.

The essential point is that distance-biased promotion is only likely to have a large impact on click-distance when very distant sites are promoted because of the distance bias. The behaviour of mobile portal users is so limited that such distant sites, although they exist, are rarely accessed and thus rarely promoted. Therefore, although distance-biased promotion has the potential to improve our pure probabilistic personalization technique *in theory*, we find that in practice it is not well adapted to the needs of behaviour of real mobile portal users. Contrast this with the work of [1] where a form of distance-biased personalization is used to good effect in the generation of shortcut links between Web pages. Web usage is less limited than mobile portal usage and Web users are more likely to follow long chains of links to their destination content. Therefore, the benefits of distance-biased personalization are more pronounced.

5 Conclusions

In general, limited usability and poor value-for-money are major contributing factors to the low levels of interest in the mobile Internet currently shown by the general public. These problems are closely aligned with the difficulty that users have in locating content on mobile portals. This navigation problem is especially acute on the mobile Internet and in previous work we have demonstrated how personalization techniques can be used as a potential solution [15, 16].

In this paper we have focused on ways to further improve our personalization approach by directly considering the navigation distance to portal items during personalization - to promote more distant items before nearby items on the assumption that distant items are likely to lead to improved navigation savings. However, after evaluating this approach on 3,500 users of a large European portal, we have found that any improvements are marginal and short-term. However, this is not so much a failing of the distance-biasing concept, but rather a side-effect of the usage patterns of mobile users. The simple fact of the matter is that, compared to their Web cousins, mobile users are impatient and rarely tolerate long navigation times. The majority of accesses are to content sites that are within a limited distance of the portal home page and this fundamentally limits the impact of any distance-bias that is introduced into the personalization mechanism.

REFERENCES

[1] C. Anderson, P. Domingos, and D. Weld, 'Adaptive Web Navigation for Wireless Devices', in *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pp. 879–884, (2001).

[2] D. Billsus, M.J. Pazzani, and J. Chen, 'A learning agent for wireless news access', in *Proceedings of Conference on Intelligent User Interfaces*, pp. 33–36, (2000).

[3] X. Fu, J. Budzik, and K. Hammond, 'Mining Navigation History for Recommendation', in *Proceedings of Conference on Intelligent User Interfaces*, pp. 106–112, (2000).

[4] J. Jacko and G. Salvendy, 'Hierarchical Menu Design: Dreadth, Depth and Task Complexity', *Perceptual and Motor Skills*, **82**, 1187–1201, (1996).

[5] J.I. Kiger, 'The Depth/Breadth Tradeoff in the Design of Menu-Driven Interfaces', *International Journal of Man/Machine Studies*, **20**, 201–213, (1984).

[6] K. Larson and M. Czerwinski, 'Web Page Design: Implications of Memory, Structure and Scent for Information Retrieval', in *Proceedings of the CHI'98 Human Factors in Computer Systems*, pp. 25–32. ACM Press, (1998).

[7] E. Lee and J. MacGregor, 'Minimizing User Search Time in Menu Retrieval Systems', *Human Factors*, **27**, 157–162, (1985).

[8] D.P. Miller, 'The Depth/Breadth Tradeoff in Hierarchical Computer Menus', in *Proceedings of the 25th Annual Meeting of the Human Factors and Ergonomics Society*, pp. 296–300, (1981).

[9] M. Perkowitz, *Adaptive Web Sites: Cluster Mining and Conceptual Clustering for Index Page Synthesis*, PhD Thesis, Department of Computer Science and Engineering. University of Washington, 2001.

[10] M. Perkowitz and O. Etzioni, 'Towards adaptive web sites: Conceptual framework and case study', *Journal of Artificial Intelligence*, **18(1-2)**, 245–275, (2000).

[11] M. Ramsey and J. Nielsen, *The WAP Usability Report*, Neilsen Norman Group, 2000.

[12] D. Reiken, 'Special issue on personalization', *Communications of the ACM*, **43(8)**, (2000).

[13] B. Smyth, *The Plight of the Mobile Navigator*, MobileMetrix, 2002.

[14] B. Smyth and C. Cotter, 'Wapping the Web: A Case-Study in Content Personalization for WAP-enabled Devices', in *Proceedings of the 1st International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'00)*, pp. 98–108, (2000).

[15] B. Smyth and C. Cotter, 'Personalized Adaptive Navigation for Mobile Portals', in *Proceedings of the 15th European Conference on Artificial Intelligence - Prestigious Applications of Artificial Intelligence*. IOS Press, (2002).

[16] B. Smyth and C. Cotter, 'The Plight of the Navigator: Solving the Navigation Problem for Wireless Portals', in *Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'02)*, pp. 328–337. Springer-Verlag, (2002).

[17] K. Snowberry, S.R. Parkinson, and N. Sisson, 'Computer Display Menus', *Ergonomics*, **26**, 699–712, (1983).

[18] D. Wallace, N. Anderson, and B. Shneiderman, 'Time Stress Effects on Two Menu Selection Systems', in *Proceedings of the 31st Annual Meeting of the Human Factors and Ergonomics Society*, pp. 727–731, (1987).

[19] J.M. Webb and A.F. Kramer, 'Maps or Analogies? A Comparison of Instructional Aids for Menu Navigation', *Human Factors*, **32**, 251–266, (1990).

[20] P. Zaphirs, 'Depth vs. Breadth in the Arrangement of Web Links', in *Proceedings of 44th Annual Meeting of the Human Factors and Ergonomics Society*, pp. 139–144, (2000).

ACKNOWLEDGEMENTS

We would like to thank ChangingWorlds for their support and assistance. Special thanks must also go to David McSherry for his comments and help during the initial stages of this research.

An Analysis of Feedback Strategies in Conversational Recommenders

Barry Smyth[1] and **Lorraine McGinty**[1]

Abstract. User feedback is a vital source of information for recommender systems that adopt a conversational style of interaction, allowing them to make better suggestions by automatically updating their understanding of user requirements on each interaction. There are many different ways that users can provide feedback but to date there has been little by way of comparative evaluation. In this paper we evaluate and compare the four most common forms of feedback used by conversational recommender systems. We focus on the implications for recommendation efficiency so that the developers of recommender systems can better understand the cost-benefit characteristics of each.

1 Introduction

Conversational recommender systems [1, 5, 8] help users navigate through complex product spaces in pursuit of suitable information items (e.g. products, services, etc.), selecting items for recommendation and eliciting feedback from the user before making the next batch of recommendations. Feedback allows a recommender to make better suggestions by adapting its current understanding of the user's requirements. The recommendation task is somewhat straightforward, although hardly trivial, when the user's requirements are clearly specified - a standard similarity-based search technique will produce a ranked list of recommendations based on their similarity to the user's fully specified requirements. Of course in the real world, and especially in the consumer application domains where recommender systems are commonly deployed, user requirements are rarely fully specified leaving the recommender to work with an partial set of requirements that are unlikely to probe the product space in the region of a suitable product during an initial retrieval.

To combat this, conversational recommender systems make use of two basic strategies [11], *navigation by asking* and *navigation by proposing*, which rely on different forms of user feedback (see Figure 1). In navigation by asking the user is *asked* to provide additional information in the form of specific features; a form of feedback known as *value elicitation*. And although this may appear to be the easy way out of the requirements predicament, the key challenge concerns the selection of an optimal set of features to request from the user [4, 10]. Navigation by proposing solicits requirements information from the user in a less direct manner. Instead of asking for specific feature details, the recommender *proposes* a set of cases to the users and offers them an opportunity to comment on these recommendations. For example, a user may indicate a *preference* for one case ("Show me more like case 1") or they may *critique* a case by indicating that their preference subject to some feature alteration ("Show me more like case 1 only cheaper"). Alternatively, a user may *rate* a set of recommendations ("Show me more given that case 1 is about 20% correct, case 2 is 85% correct, but case 3 is only 30% correct.")

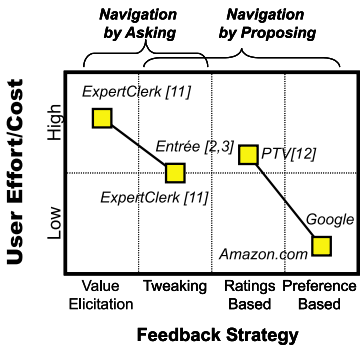


Figure 1. Categorising different user feedback forms under the headings *navigation by asking* and *navigation by proposing*.

Early research has been dominated by navigation by asking and value elicitation feedback. It is, after all, a rich source of feedback for the recommender and likely to result in efficient recommendation dialogs. However, as recommender systems have been developed for a broader range of application scenarios, the shortcomings of value elicitation have become more and more apparent. For example, while it may be a rich source of feedback, it is also a costly form of feedback from the user's perspective. Indeed, in general, there is a trade-off between the ability of a particular form of feedback to efficiently guide the recommender systems and the cost to the user of providing this type of feedback. If value elicitation sits at one extreme, then simple preference-based feedback sits at the other, its low-cost balanced by a limited ability to guide recommendations. In the end the appropriate form of feedback to use will depend largely on the characteristics of a particular application scenario. In some situations the efficiency of navigation by asking with its direct form of value elicitation feedback is warranted, with the high-cost to the user easily out-weighed by its unrivalled recommendation efficiency. But then again, in other scenarios, where cost to the user must be minimised, preference-based feedback strikes an optimal balance even though it is likely to result in protracted recommendation dialogs.

Given the importance of feedback in conversational recommenders it is somewhat peculiar that the research to date has been largely quiet when it comes to critically assessing the pros and cons of different forms of feedback. Indeed while many researchers have described and evaluated specific instances of feedback in the context of particular application domains [7, 8, 9, 12], comparative evalua-

[1] The Smart Media Institute, University College Dublin, Dublin 4, Ireland

tions are conspicuous by their absence. Developers of recommender systems need to be able to better understand the cost-benefit implications of the primary forms of feedback if they are to maximise the value of their solutions. In this paper we attempt to address this gap in the literature by evaluating four different forms of feedback (value elicitation, critiquing, ratings-based, and preference-based). In particular, we highlight a number of critical domain characteristics that have a bearing on the choice of feedback strategy and we empirically evaluate each in terms of its recommendation efficiency.

2 Comparing Feedback Strategies

Feedback from the user is an essential feature of conversational recommender systems, helping the recommender to better understand the user's requirements and so better focus its attention in the *right* region of the recommendation space. And, as highlighted in the introduction, there are four distinct forms of feedback - value elicitation, critiquing, preference-based, and ratings-based feedback - each with its own potential benefits and pitfalls.

In this section we will describe these different types of feedback in more detail and assess their comparative strengths and weaknesses. But first we must consider what factors tend to influence the use of different forms of feedback in a recommender system as a basis for this comparison? For example, in [6] the importance of *user cost* (the effort required of the user to provide a particular type of feedback) was emphasised, and the different forms of feedback were compared along this dimension. But other dimensions must also be considered.

2.1 Influencing Factors

Three important factors contribute to the suitability of a given form of feedback for a particular recommendation task are: the *cost* to the user of providing the feedback; the level of domain *expertise* required by the user to provide the feedback; and the type of user *interface* needed to capture the feedback (see Figure 2).

Feedback	Cost	Expertise	Interface
Value Elicitation	***	***	***
Ratings	**	**	*
Critique	**	**	*
Preference	*	*	*

Key: * Low ** Moderate *** High

Figure 2. A comparison of feedback strategies.

2.1.1 Cost to the User

It goes without saying that different forms of feedback carry very different costs from a user's perspective. For example, one form of feedback might have a high cost because it causes the user to think very carefully about the type of information they are being asked for, or the level of detail they are being asked to provide. For example, a car recommender that asks a user for a precise fuel consumption requirement in terms of kilometers per litre carries a higher cost than a similar recommender that simply asks them to indicate a preference for high, moderate or low fuel consumption. Alternatively, asking a user to provide information that they regard personally valuable translates into a higher cost for the user. In general then, asking users for specific feature values carries a high cost than asking them to indicate a general preference.

2.1.2 Domain Expertise

Domain expertise is another well-known factor that influences the type of feedback that is appropriate in a given recommendation context. The basic idea is that the ability of a user to provide a certain form of feedback depends, naturally enough, on whether the user has a clear understanding of what they are being asked to provide. For instance, in order for a user to provide feedback in the form of a specific answer to a specific question about a specific feature she must know the correct answer to the question as it relates to her needs. Take the PC domain for example. Here, to provide feedback about her preferred memory requirements our user must have a relatively detailed understanding about the PC domain, the range of memory sizes that are available, and their implications for her PC needs. This level of domain expertise adds an additional burden on the user and cannot always be relied upon. Furthermore, in many domains users are able to express a preference without necessarily understanding individual item properties. This is especially true in domains where a specialised vocabulary already exists to describe items (e.g. fashion, jewellery, art, music, etc.), but where users judge the relevance of specific cases by different means, for example, by sight, sound or taste. The essential point here is that for certain forms of feedback to work well system users must have a minimum degree of domain expertise and this must be carefully considered in the light of likely expertise levels within the target user population. If users are unlikely to have the expertise that allows them to answer specific feature questions then value elicitation and critiquing forms of feedback may not be appropriate, for example.

2.1.3 Interface Requirements

The third factor that we believe has an important impact on the form of feedback used by a recommender system has also been largely ignored by the literature to date. It concerns the interface requirements of the feedback. Simply put, different forms of feedback impose different interface requirements on the recommender system. Some forms of feedback require very sophisticated interfaces to capture the detail required, while other forms have must simpler requirements as we will see. In turn, the implementation of a given recommender will offer certain interface capabilities depending, for example, on the type of device that is to be used to deliver the recommender interface. Problems occur when the interface requirements of a form of feedback are not matched by the capabilities of the recommender's interface. For example, forms of feedback that require the user to provide specific feature value information expect the user to enter specific value data from a keyboard. In the past, because most recommenders have been implemented on traditional PC interfaces, this has not been a problem. However, with the advent of mobile devices, and their far more limited input capabilities, this expectation is not always realistic. For example, asking a user to type complex feature value information into a WAP phone, or even a PDA, is unlikely to be acceptable to the user. Similarly, imagine a recommender system developed for use on a voice portal, and assuming the availability of competent voice-recognition services. How can users provide feedback from a simple telephone handset? The point is that many new

devices, such as mobile phones, PDAs, digital TVs etc. are emerging as new platforms for next-generation recommender systems, but these devices are limited in various ways. They often have limited presentation capabilities (eg. many mobile phones have screen sizes that are hundreds of times smaller than the screen of a standard descktop PC). And they nearly always have limited input capabilities that make it difficult for users to provide input that is more complex than a simple selection.

2.2 Forms of Feedback

So far we have described the various factors that influence the form of feedback than can and should be used in a recommender system without describing the various forms of feedback in any detail. In this section we outline the four basic types of feedback that are commonly used in recommender systems today - value elicitation, critiquing, ratings-based and preference-based - comparing them in terms of their user cost, domain expertise and interface requirements (see Figure 2).

2.2.1 Value Elicitation

Value elicitation (e.g. "I want a 1GHz Pentium PC"), is perhaps the most common form of feedback. However, to provide this level of feedback the user needs a high level of domain expertise to specify a reasonable feature value. In addition to this the user must be willing to answer the direct and specialised questions posed. Finally, to provide detailed feature-level feedback demands a sophisticated user interface. Generally speaking the user must type in a specific feature value and this limits its use on mobile devices, for example. Having said this some forms of value elicitation feedback can be implemented using alternative interfaces such as drop-down lists or selections lists, but in general these options are limited to nominal features with a limited range of possibilities.

2.2.2 Critiquing

The FIND-ME group of recommender systems [3, 2] have introduced critiquing (or *tweaking*) as a powerful form of feedback. Briefly, the user is asked to express feedback in the form of a critique. In short the user implicitly selects one of the presented cases and expresses directional preferences for a certain feature. For instance, in the Entree restaurant recommender (see [3]), the user is presented with the single best case during each recommendation cycle and can introduce tweaks over features such as *price* (find me cheaper or more expensive restaurants) or *style* (find me a more casual or less casual restaurant), for example. As a form of feedback critiquing relies on users have at least some minimal understanding of the recommendation feature space in order to specify suitable critiques. Unlike value elicitation however, critiquing benefits from minimal interface requirements due to the limited number of critiques that can be applied in most circumstances; critiquing can be easily implemented using a simple point-and-click type interface that enables feedback to be given at the feature-level.

2.2.3 Ratings-Based

Recently, ratings-based feedback has become a popular form of feedback in recommender systems. The basic idea is to ask the user to rate individual recommendation proposals; as such the user is providing case-level feedback. For example, PTV is a recommender system operating in the TV listings domain [12]; it compiles personalized TV guides for users based on their learned TV preferences and as such is involved in recommending TV programmes to users. PTV encourages users to rate its programme recommendations (using a 5-point scale from strong-negative to strong-positive) as a means of fine tuning subsequent recommendation cycles. Generally speaking ratings-based feedback is a relatively low-cost form of feedback since the user need only express a qualitative (or perhaps quantitative) indication of interest at the level of an individual case. Having said this, the level of effort naturally increases if the user needs to rate many or all of the recommended items - although in most systems, such as PTV, the user can chose to rate as few or as many of the items as she desires. This form of feedback make relative moderate demands in terms of its interface requirements, but once again its use in simple interfaces is potentially challenging if the user must provide many ratings in a given feedback cycle.

2.2.4 Preference-Based

Preference-based feedback (e.g. "I prefer PC1") is a low-cost form of feedback and can be provided by users, through a simple interface [7]. For example, even the simplest mobile phone interface is amenable to this form of feedback because it generally requires little more than point-and-click type interface functionality. It is also suitable for voice-based recommenders in which the user listens to a small number of alternatives and selects one as their preference in much the same manner as a standard voice-portal menu system. Importantly, this form of feedback can be used even when users have only a very limited understanding of the recommendation domain. We saw earlier how in the Whiskey example most users were able to express a simple preference without necessarily understanding the specific taste features of the Whiskeys. Thus, preference-based feedback is suitable in this domain and for these users, but value elicitation is not.

2.3 Ambiguity & Recommendation Efficiency

One feedback dimension that is conspicuous by its absence concerns the level of inherent *ambiguity* associated with a given strategy, which in turn has implications when it comes to recommendation efficiency. Ultimately recommender systems solicit feedback from users in order to better guide the recommendation process. The objective is to bring the user to a satisfactory product/service as quickly, and with as few interactions, as possible. But the ability of a given form of feedback to guide the recommender depends critically on the degree to which the feedback unambiguously identifies a given user's relevant requirements. Accordingly, forms of feedback that solicit direct feature value information from users are likely to be less ambiguous than other forms of feedback that simply ask a user to express a preference for, or rate, a case. Soliciting a direct feature value from a user regarding their price requirements for a PC, for example, provides a clear indication of their needs in relation to price and allows the recommender to prioritise those cases in the appropriate price range during subsequent recommendation cycles. However, a user's preference for a specific PC case does not necessarily provide such a clear indication of their reasons for the preference. Perhaps they like like price of the preferred case, but equally they may be interested in the processing power or perhaps the memory size; indeed they may actively dislike some of the features of the preferred case.

For these reasons, the features of a preferred case are ultimately ambiguous with respect to a user's real requirements and as such provide only limited guidance for the recommender; see [6, 13] for recent advances with this form of feedback.

In summary, latent ambiguity is likely to compromise the efficiency of the recommender system and different forms of feedback have different degrees of ambiguity. But how do these different ambiguity characteristics actually relate to recommendation efficiency, quantitatively speaking? This is the central question to be explained during the remainder of this paper by evaluating the recommendation efficiency of these different feedback strategies in different domains.

3 Feedback in Comparison-Based Recommendation

Comparison-based recommendation is a conversational recommendation framework that supports the navigation by proposing strategy described in section 1. Here each cycle of a user/system conversation consists of 3 key stages: (1) new items are *recommended* to the user based on the current query; (2) the user *reviews* the recommendations and provides feedback in terms of how these suggestions relate to their requirements; (3) this feedback information is then used to automatically *revise* the query for the next cycle. The recommendation process terminates either when the user is presented with a suitable item or when they give up.

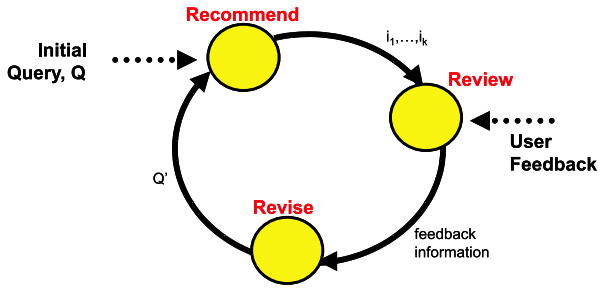


Figure 3. The Comparison-Based Recommendation cycle.

Previous research has looked at ways of maximising performance benefits at the **Recommend** and **Revise** stages. For instance, the interested reader is referred to [6] for a description of a number of highly-effective query revision strategies, which are based on principled ways of selecting and weighting features in the revised query as a result of preference feedback. Similarly, works [8, 13] concentrate on the **Recommend** stage, where a range of selection strategies are evaluated and compared.

In this paper, the focus is on the **Review** stage, whereby a user is invited to provide feedback on a recommendation set. There are a number of different ways a user can do this in a typical navigation-by-proposing setting. Examples include *Preference-Based Feedback*, *Critiquing*, and *Rating-Based Feedback* (see Section 4 for descriptions on these). The comparison-based framework is capable of supporting all of these approaches.

4 An Empirical Analysis of Feedback Strategies

We wish to test four basic conversational recommendation systems, each employing a different form of feedback: (1) VEF - a pure similarity-based recommender employing value elicitation; (2) CBF - a comparison-based recommender that employs critiquing-based feedback; (3) RBF - a comparison-based recommender that employs ratings-based feedback; and (4) PBF - a comparison-based recommender employing preference-based feedback. Note that the comparison-based recommenders are implemented with $k = 3$; that is, 3 alternative cases are presented in each recommendation cycle.

4.1 Data-Sets and Test Queries

Two test case-bases are used for this evaluation. First of all, the *Travel* case-base contains 1024 cases[2], each describing a specific vacation in terms of features such as *location*, *duration*, *accommodation*, *price* etc. The *Whiskey* case-base ([9]) contains a set of 552 cases, each describing a particular Scotch whiskey in terms of features such as *distillery*, *age*, *proof*, *sweetness*, *flavour*, *finish* etc.

For each data set, three different groups of queries are generated of varying degrees of difficulty (*easy*, *moderate*, *difficult*); difficulty is based on the number of cycles required by preference-based feedback. To generate test queries we randomly selected a number of cases from each case-base (200 from the Whiskey case-base, and 300 from travel). These cases served as the basis for each set of queries constructed by taking random subsets of item features. The actual number of query terms selected was varied between 1 and 5.

4.2 Methodology

We used a leave-one-out methodology to solve the test queries, whereby we first identified the *base* case for a given query (i.e. the case it had originally been generated from) in the casebase. This *base* case is temporarily removed and we select the case that is most similar to it to act as the recommendation *target* for the recommendation session. Thus, the *base* represents the ideal query for a user, the generated query is the initial query that the user provides to the recommender, and the target is the *best* available case for the user based on their ideal. Each generated query serves as a test problem for each recommender and in each recommendation cycle the appropriate feedback is provided according to the original *parent* case. For example, in VEF a random feature is chosen and felled with the appropriate value of the *base* case; not we do not use any sophisticated feature selection techniques here to determine which feature should be queries or in what order. For the CBF, RBF and PBF systems, in each recommendation cycle the users preference is assumed to be the case that is most similar to the known target case. Preference-based, ratings-based or critiquing-based feedback is applied to this preference case as appropriate. In the case of critiquing, a random critique is applied to the preferred case in each cycle, based on the values in the *base* case, and for ratings-based feedback, the rating provided corresponds to the similarity score between the rated case and the *base* case.

4.3 Recommendation Efficiency

Recommendation efficiency can be measured in terms of the average number of unique cases that a user must evaluate before finding their ideal target. For both data-sets the leave-one-out method outlined above is used by the four conversational recommenders listed above and the mean number of unique items presented to the user are measured. The results are shown in Figure 4(a) for Travel and in Figure 4(b) for Whiskey, as graphs of the mean unique cases for

[2] This data-set can be downloaded at http://www.ai-cbr.org.

each algorithm by query group. As expected, VEF is the clear winner in both sets of results, followed by CBF in second place, RBF in third, and PBF being the poorest performer of all. However, as discussed in section 2, VEF assumes that the user will be able and willing to provide feature-level feedback on each recommendation cycle. For example, in the whiskey domain the evaluation for VEF assumes that the user will be able to comment on their level of preferred *peatiness*, *sweetness* etc. This will not always be possible, and as such a navigation-by-proposing strategy might be the more appropriate alternative. Looking at the moderate query category for the Travel data-set (Figure 4(a)) we see that CBF requires the user to examine 40 unique cases, RBF 65, and PBF 100. That is, CBF represents a relative reduction of nearly 26% over RBF, and 60% over PBF in terms of the average number of cases examined. Moreover, the relative reductions enjoyed by CBF tend to increase with query difficulty. For instance, for the Travel data-set we find that CBF enjoys a relative benefit over PBF of nearly 97%! Put another way, to satisfy a difficult query, CBF is 3 times more efficient than PBF (i.e., the same system using PBF requires the user to examine 3 times as many items!).

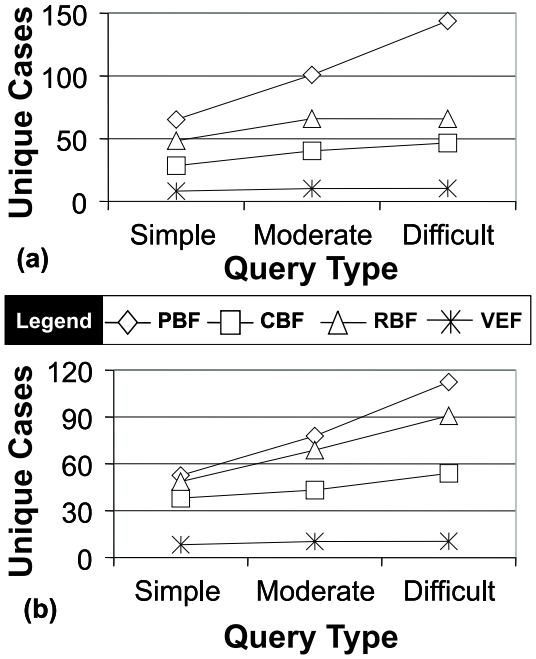


Figure 4. Recommendation efficiency evaluation results for the Travel (a) and Whiskey (b) data-sets.

4.4 Target Tolerance

Of course the previous evaluation assumes that a user is searching for a particular target item, and that the dialog only terminates when this item is returned. In reality, a user may end there search when they are presented with an item that is similar but maybe not an exact to the optimal target. To evaluate this idea we relaxed termination condition by repeating the basic efficiency experiment, terminating each dialog once an item has been recommended that is within some pre-defined similarity of the target. A similarity threshold of 70% means that the dialog terminates when an item that is at least 70% similar to the target has been recommended.

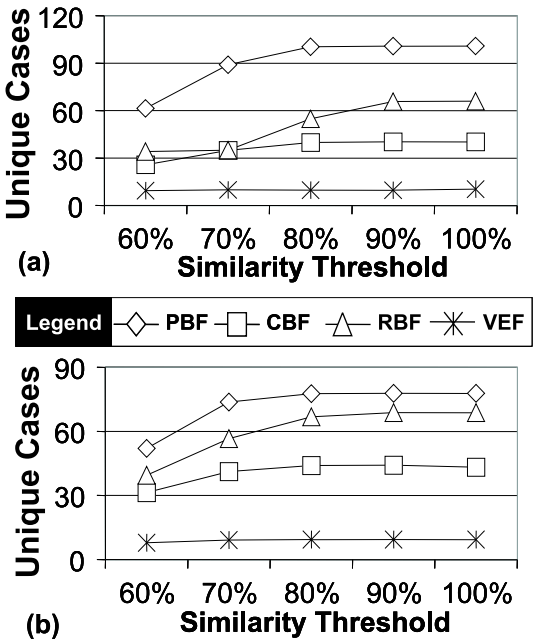


Figure 5. Target tolerance evaluation results for the Travel (a) and Whiskey (b) data-sets.

Figure 5(a) and (b) present the results we found for the Travel and Whiskey domains, by graphing the mean number of items presented to the user versus the similarity threshold for moderate queries. Once again, the performance of VEF is unaffected by this experiment for the very same reason as before. As expected, relaxing the termination condition results in shorter recommendation dialogs for all of the navigation-by-proposing approaches. For example, in Travel (Figure 5(a)) RBF dialogs reduce from just under 66 items at the 100% similarity threshold (where the optimal item must be recommended) to just over 34 items at 60% similarity. Similar results are found for all approaches the Whiskey domain (Figure 5(b), both for simple and difficult queries.

4.5 Discussion

It is very difficult to define a set of hard and fast rules to govern the choice of feedback method. Based on their qualitative differences a number of broad guidelines are worth highlighting:

1. In domains where users are unlikely to have a detailed understanding of the features of recommended items or cases, or where users are unlikely to tolerate being asked detailed requirements questions, then value elicitation and critiquing are unlikely to be useful, and should be avoided in favour of ratings-based or preference-based feedback.
2. In domains where recommendation efficiency is likely to be critical, for example where obtaining a satisfactory recommendation is time-critical, then feedback ambiguity should be minimised suggesting the use of value elicitation or critiquing.
3. When a recommender system is to be deployed on a device limited interface and input capabilities then value elicitation and detailed ratings-based feedback should be avoided. In the extreme case, preference-based feedback is most appropriate but critiquing is also generally feasible.

This analysis has looked at the quantitative differences that exist between feedback strategies. These findings are summarized by Figure 6, showing the average speed-up for preference-based feedback (PBF), critiquing-based feedback (CBF) and rating-based feedback (RBF), over the value-elicitation approach (VEF). There are three very clear (and important) results:

1. Preference-based feedback (PBF) is an order of magnitude less efficient than the value-elicitation approach. For example, it shows a speed-up of 9 over VEF at the moderate query category. That is, on average a user should expect to examine 9 times as many items before they locate their target item using the 'user-friendly' PBF approach rather than a VEF approach.
2. There is a clear distinction between case-level (i.e. PBF and RBF) and feature-level (VEF and CBF) feedback; the former being significantly less efficient than the later.
3. Finally, there is a close relationship between RBF and PBF in terms of their efficiency characteristics, for simple and moderate queries. Notably, PBF seems to be more sensitive to difficult queries than RBF. Thus if, for whatever reason, feedback had to be elicited from a user at the case-level there is an argument that says PBF is a strong contender (given it's low user cost characteristics) over RBF, so long as the queries are not too difficult.

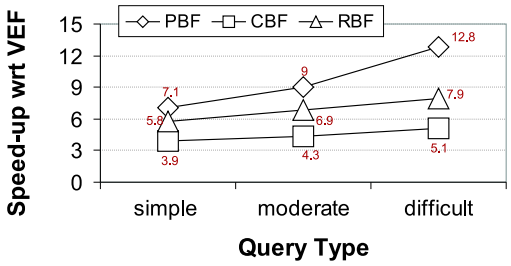


Figure 6. Feedback analysis summary.

5 Conclusions

User feedback is a vital source of guidance information to conversational recommenders. User cost, the users level of domain expertise and interface requirements are well-known factors that influence the type of feedback that is appropriate in a given recommendation context. Level of ambiguity is less well-known but, we believe, this is equally important. In an attempt to better understand the cost-benefit implications of the primary forms of feedback, this paper compares four different feedback strategies (i.e. value elicitation, critiquing, preference-based, and ratings-based feedback) in terms of the level of ambiguity characteristic of each.

For example, value elicitation whereby the user is asked to provide for a specific feature value it is a rich source of requirements information with little inherent ambiguity; knowing that the user is interested in items with a particular feature allows the recommender to eliminate many irrelevant items from consideration. In short, this form of feedback benefits from minimal inherent ambiguity but imposes a high user cost with significant expertise and interface requirements. Critiquing can be viewed as a limited form of value elicitation and as such it enjoys some benefits in relation to its limited ambiguity.

Unlike value elicitation and critiquing, ratings-based feedback suffers from high levels of ambiguity and as such is likely to be limited in its ability to guide the recommendation process. Similarly, preference-based feedback also suffers from high levels of ambiguity. On its own a simple preference for an item is inherently ambiguous with respect to the user's intent since the reasons for the user's preference (in terms of case features) are rarely clear.

Intuitively these observations are no surprise. Instead the significance of this paper is the quantitative comparison of user feedback strategies, where we compare feedback strategies according to the degree to which they *out-do* eachother in terms of their efficiency performance. Ultimately, we believe that this quantitative analysis can at least serve as a guide to recommender system developers, helping them to better understand the tradeoffs that exist between different forms of feedback.

ACKNOWLEDGEMENTS

The support of the Informatics Research Initiative of Enterprise Ireland is gratefully acknowledged.

REFERENCES

[1] D. Bridge, 'Towards Conversational Recommender Systems: A Dialogue Grammar Approach.', in *Proceedings of the Mixed-Initiative Workshop on CBR at the European Conference on Case-Based Reasoning (ECCBR-02).*, ed., D.W. Aha, pp. 8–22. The Robert Gordon University, (2002). Aberdeen, Scotland.

[2] R. Burke, 'Knowledge-based Recommender Systems.', *Encyclopedia of Library and Information Systems*, **69(32)**, (2000).

[3] R. Burke, K. Hammond, and B.C. Young, 'The FindMe Approach to Assisted Browsing.', *Journal of IEEE Expert*, **12(4)**, 32–40, (1997).

[4] M. Doyle and P. Cunningham., 'A Dynamic Approach to Reducing Dialog in On-Line Decision Guides.', in *Proceedings of the 5th European Workshop on Case-Based Reasoning, (EWCBR-00)*, eds., E. Blanzieri and L. Portinale, pp. 49–60. Springer, (2000). Trento, Italy.

[5] M. Goker and C. Thompson, 'Personalized Conversational Case-based Recommendation', in *Proceedings of the 5th European Workshop on Case-based Reasoning, (EWCBR-00)*, eds., E. Blanzieri and L. Portinale, pp. 99–111. Springer-Verlag, (2000).

[6] L. McGinty and B. Smyth, 'Comparison-Based Recommendation.', in *Proceedings of the 6th European Conference on Case-Based Reasoning (ECCBR-02)*, ed., S. Craw, pp. 575–589. Springer, (2002). Scotland.

[7] L. McGinty and B. Smyth, 'Evaluating Preference-Based Feedback in Recommender Systems.', in *Proceedings of the 13th National Conference on Artificial Intelligence and Cognitive Science*, ed., R. Sutcliffe. Springer, (2002). Limerick, Ireland.

[8] L. McGinty and B. Smyth, 'The Role of Diversity in Conversational Systems.', in *Proceedings of the 5th International Conference on Case-Based Reasoning (ICCBR-03)*, eds., D. Bridge and K. Ashley. Springer, (2003). Troindheim, Norway.

[9] L. McGinty and B. Smyth, 'Tweaking Critiquing.', in *Proceedings of the Workshop on Personalization and Web Techniques at the International Joint Conference on Artificial Intelligence (IJCAI-03)*. Morgan-Kaufmann, (2003). Acapulco, Mexico.

[10] D. McSherry, 'Minimizing dialog length in interactive case-based reasoning.', in *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-01)*, ed., Bernhard Nebel, pp. 993–998. Morgan Kaufmann, (2001). Seattle, Washington.

[11] H. Shimazu, A. Shibata, and K. Nihei, 'ExpertGuide: A conversational case-based reasoning tool for developing mentors in knowledge spaces', *Applied Intelligence*, **14(1)**, 33–48, (2002).

[12] B. Smyth and P. Cotter, 'A Personalized TV Listings Service for the Digital TV Age', *Journal of Knowledge-Based Systems*, **13(2-3)**, 53–59, (2000).

[13] B. Smyth and L. McGinty, 'The Power of Suggestion.', in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-03)*. Morgan-Kaufmann, (2003). Acapulco, Mexico.

Making the Constraint Model of Concept Combination More Familiar and Efficient

Georgios Tagalakis and **Mark T. Keane** and **Dermot Lynott** [1]

Abstract. Constraint theory and the C^3 model, which is the computational-level implementation of the Constraint theory, explain how novel word combinations may be understood satisfying three constraints of diagnosticity, plausibility, and informativeness. Two limitations of the C^3 model are less efficient computation procedures that operate over a large space of possibilities in order to identify the best interpretations, and the system's inability to deal with familiarity effects. In this paper we explore the issue of familiarity in the comprehension of noun-noun compounds and we present PUNC - a prototype system for Producing and Understanding Noun-Noun Compounds. PUNC tries to overcome algorithmic limitations of the C^3 model in being more efficient in its computational complexity. It also deals with a wider span of empirical phenomena, such as dimensions of word familiarity.

1 Introduction

Conceptual combination has been a major focus for Cognitive Science research because of its relevance to compositionality, language generativity and change (see [7], [9], [11]). Conceptual combinations are manifested in everyday language through compound phrases (e.g., *blood oak*, *sausage meat*, *jail job*). Some of these compounds are familiar (e.g., *rubber hose*, *rugby ball*) and others are quite unfamiliar (e.g., *meadow carpet*, *organ bowl*). Clearly, any algorithmic treatment of concept combination has to be able to account for the understanding of familiar and unfamiliar compounds. However, most of the literature on concept combination has concentrated on the more exotic, novel compounds. In this paper, we spend some time trying to extend accounts developed for novel compounds to incorporate their more familiar cousins. In the process, we present a new model of conceptual combination called PUNC.

1.1 Background Literature

There are three main theories that endeavor to explain the comprehension of novel compounds; the Dual-Process theory [24], the Competition Among Relations in Nominals theory (CARIN) [13], and the Constraint theory ([4], [5], [17]).

The Dual-Process theory ([24], [25]) is designed to account for the two major categories of interpretation that people produce to compounds; relational and property interpretations. Relational interpretations involve the use of a thematic relation connecting the two words - the modifier and head words - in the compound (e.g., *dawn flight* is "a flight that takes off at dawn"). Property interpretations assert a property of one concept of the other concept (e.g., *bullet train* is "a very fast train", asserting properties of a *bullet* and of a *train*). Dual-Process theory posits a different process to account for each of these interpretation types. Relational interpretations are produced by a concept specialization process, where some slot in the head concept is specialized with the modifier concept. Property interpretations are seen as being produced by an analogical mapping mechanism (cf., [10], [16]). Unfortunately, [24] and [25] have never produced an implemented model of Dual-Process theory and have left open the methods by which processing might switch between the interpretation strategies.

The CARIN model ([13], [12]) prioritizes relational interpretations over property ones. It proposes that the relations used in these interpretations are selected from a set of thematic relations that are associated with the modifier word from previous known compounds. For example, *chocolate dog* is readily understood as "a dog made of chocolate" because *made of* is strongly associated with the word *chocolate* from many previously-known compounds (e.g., *chocolate bar*, *chocolate cake*, *chocolate egg*). CARIN makes predictions of ease of understanding novel compounds by performing a corpus analysis of the relations associated with known compounds and then using this information in a mathematical choice model. While CARIN is very interesting, it has a narrow focus (i.e., relational interpretations) and does not explain how initial or new relational interpretations arise.

The Constraint theory tries to deal with many of the limitations of these other models ([4], [5], [6], [17]). It can account for the production of both relational and property interpretations with a unitary generative mechanism that produces all possible interpretations that are subsequently filtered by the constraints of diagnosticity, plausibility, and informativeness. Furthermore, the theory has been implemented and has been shown to parallel many aspects of people's behavior. In this paper, we consider some of the limitations of the C^3 model and how it might be extended to the understanding of familiar compounds.

1.2 Outline of Paper

The aim of this article is to explore the issue of familiarity in the comprehension of noun-noun compounds and to examine a modified version of C^3 - a C^3lite, if you like - that tries to

[1] Department of Computer Science, University College Dublin, National University of Ireland

deal with familiarity. The rest of the paper is structured as follows. First, we give a short overview of the Constraint theory and the C^3 model. Then, we introduce the idea of a constraint for familiarity, which has been identified as an important factor that exerts a marked effect on word comprehension. Next, we present a new prototype system for representing knowledge and producing interpretations that we developed in an attempt to accommodate the new factor and improve the performance of C^3. We conclude with a discussion and directions for future research.

2 The Constraint Theory and the C^3 Model

The Constraint theory is a computational-level theory for the conceptual combination of novel noun-noun compounds ([4], [5], [6], [17]). It assumes that there is a generated space of possible meanings based on combining the predicates of the two noun concepts in all possible ways. It then assumes that the best set of interpretations are those, which meet the three constraints of diagnosticity, plausibility, and informativeness.

The diagnosticity constraint requires that a good interpretation must, to some extent, contain diagnostic predicates of both constituent concepts of a compound phrase. For example, diagnosticity predicts that "prickly fish" is a better interpretation than "green fish" for the compound *cactus fish*, because *prickly* is a more diagnostic property of the concept *cactus* than its property of being *green* (cf., [11], [21]).

The plausibility constraint ensures that an interpretation contains a group of overlapping elements of the two concepts, which are likely to co-occur on the basis of previous experience, and is, therefore, pragmatically acceptable. For example, it would be plausible to understand an *angel pig* as "a pig with wings attached to its torso", because *pig* occurs with *torso* and *wings* occur with *torso*. "A pig with wings attached to its tail" would be an implausible interpretation, because *wings* and *tails* do not typically co-occur in living beings.

The informativeness constraint requires that an acceptable interpretation should convey some new information above and beyond that in the constituent concepts. For example, no one would ever find the interpretation "pencil made of wood" to be an acceptable interpretation for *bed pencil*, because most *pencils* are already *made of wood*.

The C^3 model is an implementation of the Constraint theory. The model generates a space of possible meanings by forming all possible subsets of key predicates in both concepts, taking into account different unifications of them too. For two concepts, with a handful of predicates, it can generate several thousand possible interpretations. The model then uses constraint-satisfaction techniques to filter out the best interpretations from this set. There are three main stages to the model's processing cycle. First, it generates a set of partial interpretations based on subsets of selected diagnostic predicates of the head and modifier concepts. Second, these partial interpretations are fleshed out by adding further predicates from similar concepts in memory. At this point, each interpretation has an acceptability score based on its combined diagnosticity and plausibility scores. Third, informativeness acts to remove interpretations from this set that do not convey anything new (see [5]). The model is not particularly efficient but it gets there. It tends to generate quite acceptable interpretations for the compounds, considering the inherent incompleteness of its knowledge base. Furthermore, [5] have shown that it does a good job of capturing the sort of interpretations that people tend to produce.

However, C^3 is limited because it does not take into account human-like, efficient reasoning techniques. For example, in computing plausibility C^3 compares each candidate interpretation against every concept in its knowledge base. A good model would have some way of limiting this extensive comparison process, constraining its generativity.

This algorithmic limitation has many practical implications. For example, C^3 cannot heuristically generate a small number of interpretations that will turn out to be among the most acceptable that could be found. It has to go through the full computation of all possibilities to identify the best ones[2].

Furthermore, the model does not have natural extensions to obvious empirical phenomena. For example, a familiar compound (like *chocolate egg*) would be treated exactly the same way as a novel compound (like *water egg*). Yet, it is obvious that people do not act in this way. A familiar compound would be understood very quickly by some sort of fast lexical access, whereas an unfamiliar compound might require considerable reflection. In the next section, we discuss some of our recent findings on familiarity effects in compound understanding before we move on to considering a computational model that might capture such results.

3 Familiarity and its Dimensions

Most studies of conceptual combination use noun-noun compounds that are supposed to be novel. It is generally recognized that familiar compounds would be processed in different ways and not give rise to as much polysemy as novel ones. Recently, [22] have confirmed this intuition in a series of experiments that have carefully controlled the familiarity of the compounds used. They found that familiar compounds are understood much faster than their unfamiliar counterparts and these effects occur irrespective of whether the interpretation involved is a relational or property one. These findings suggest that there is a minimum amount of semantic construction involved in understanding familiar compounds. Rather they appear to be comprehended by a more "automatic" type of memory access (see also [1], [3], [14] for more empirical evidence of the familiarity effects on lexical access and understandability). Consequently, it has been suggested in the literature that only unfamiliar combinations as objects or linguistics expressions should be considered in order to assess processes of conceptual combinations [15].

This may be only part of the story. We believe that there are at least two main types of familiarity that influence comprehensibility; the familiarity of a given compound, and the familiarity emerging from semantically-related concepts to a familiar compound.

[2] For example, a simulation process for producing the 10 most acceptable interpretations of 24 compound phrases took almost 3 weeks running on a Macintosh LC III (25 MHz 68030 processor, main memory size of 33 MB, running Mac OS 7.6 and Macintosh Common Lisp 3.0)

3.1 Familiar Compounds

A familiar compound is one that is frequently used and known to a wide language community. Intuitively, the more frequently a compound is used and cited in different sources the more familiar it becomes (cf., [20]). For example, *bullet train* is a very familiar compound that refers to a high-speed streamlined train that travels fast and is shaped like a bullet. However, there are other dictionary-defined compounds in a language that are less frequently used. For example, *diamond crossing* is a phrase known to railway enthusiasts that refers to the arrangement of a line at the point where two lines cross in the shape of a diamond. This is a known compound but is not very frequently used by the language community at large. Therefore, it might be better considered to be unfamiliar.

Clearly, there are also cases of familiar idiosyncratic, idiomatic compounds that are characterized by a strong connotational meaning (in the sense of [8], p. 111). *Ivy league*, for instance, is a frequently used term to refer to a cluster of eight well-known and prestigious universities in the United States; so-called because of their ivy-covered older college buildings. Such phrases are understood fast without the speakers being necessarily aware of the etymology of the meaning.

A familiar compound itself does not impose any special difficulties in terms of knowledge representation and retrieval. It can be stored in the mental lexicon, like a single word, as a single unit together with its interpretation and can be recalled instantly, on demand. As a matter of fact, its interpretation does not require a conceptual combination at all in the sense of combining semantic elements of the two constituent concepts.

3.2 Familiarity Due to Semantically-Related Concepts

Familiarity may also be manifested when people encounter a compound that uses semantically-related words to those used in a familiar compound (e.g., *bullet car* working off *bullet train*). For example, in the familiar *bullet train* the shape and speed properties of the head train are instantiated to the values *fast* and *streamlined* having being mapped from the modifier *bullet* (see Figure 1). When a novel compound is encountered with the same modifier and a symmetrically semantically-related head, like *bullet car*, a priming effect occurs and the meaning can get rapidly constructed using the known *bullet train* interpretation. Hence, a *bullet car* will tend to be understood as "a very fast car" and probably as "a car shaped like a bullet".

These sort of effects seem to invite some form a distributed account of memory. Typically, theories of distributed memory focus on semantic similarity to explain priming facilitation effects. An underlying assumption is that priming is proportional to semantic relatedness. If there is a feature overlap and the meanings are similar then the process requirements are few (see [23]).

3.3 Putting Familiarity into Computational Models

From what has been said so far it is clear that semantic familiarity is a phenomenon that cannot be ignored by conceptual combination models. In the context of C^3, it is unclear whether this factor should be handled by a new constraint or being an emergent feature of the action of the current constraints. We believe that familiarity is best handled as an emergent feature, as it seems to naturally arise form the action of the known constraints. We have seen already that acceptable interpretations are constructed through the constrained application of diagnosticity, plausibility and informativeness. First, the diagnostic, distinctive predicates of the constituent concepts of the novel compound and the familiar compound are compared and selected, as in the example above. Second, plausibility will be satisfied as it works off prior knowledge; namely, that of the known compound. If a familiar noun-noun compound is plausible then a semantically-related novel noun-noun must be plausible too, by default (i.e., in the absence of any contradictory semantic information). Following a similar line of argument, informativeness would be naturally met too. In the next section, we turn to a model of concept combination that tries to take these sort of factors into account.

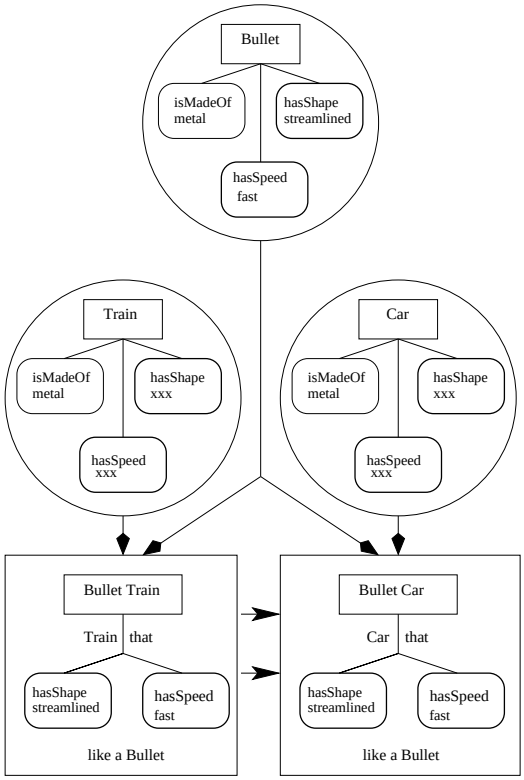


Figure 1. Priming Effects of Familiar Compounds on Novel Compounds Due to Semantic Associations

4 PUNC: A Model for Producing and Understanding Noun-Noun Compounds

With the limitations of the C^3 model in mind we developed a prototype model for Producing and Understanding Noun-Noun Combinations (PUNC) to serve as a test bed for our ideas. Currently, the PUNC model incorporates each of the three constrains of diagnosticity, informativeness, and plausibility not as separate processes but as intrinsic aspects of the

interpretation process. The model also handles familiarity.

4.1 System Structure and Algorithmic Design

PUNC represents concepts using a simple hierarchy (e.g., *vet* → *person* → *creature*) and each concept is represented by diagnostic knowledge associated with it. For example, the concept *cactus* is represented by its diagnostic information, such as *has spikes*, *grows in the desert*, *can conserve water*, and each piece of knowledge is weighted by its diagnosticity. *Cactus* also inherits the diagnostic information of *plant* (e.g., *can photosynthesize*), but this inherited knowledge is less diagnostic of a cactus than it is of a plant, so the diagnosticity weightings are lowered accordingly.

C^3 implements diagnosticity by comparing a given feature to all other concepts in the knowledge base. The frequency of occurrence of this feature in the knowledge base determines its level of diagnosticity (i.e., if it occurs only once for concept A, it would by highly diagnostic for concept A). While C^3 computed all of the diagnosticity weights for the concepts in its knowledge base, PUNC assigns standardized weights on intuition. For example, for the concept *cactus* the feature *has spikes* has been given a high diagnosticity score since it is the generally one of the most important features of cactus. Each concept has its predicates encoded in decreasing order of diagnosticity, thus giving us a partial ordering of the relative goodness of the predicates in the concept. C^3 might seem better than PUNC in this respect but one must remember that C^3 could easily generate odd diagnosticity scores because of the incompleteness of its knowledge base.

In C^3, plausibility is implemented directly through a process that compares the partial interpretation to all known related concepts. Plausibility in PUNC, on the other hand, is an implicit part of the entire combination process. First, the type of interaction between the head and modifier concepts can influence the plausibility of an interpretation. For example, in the two following cases: a *chocolate beetle* is "a beetle that eats chocolate" and *brick beetle* is "a beetle that eats bricks", the first interpretation is seen as being more plausible because *chocolate* is something that *can be eaten*, and a *beetle* is something that *can eat things*. On the other hand, while a *beetle* can still eat things, a *brick* is not something that is normally eaten, so this interpretation is seen as less plausible. So, the goodness with which the roles of the typical relations in the combination are met is an important index of plausibility. Second, the computationally-intensive plausibility stage of C^3 is avoided by employing the knowledge available from the head as the core representation for new interpretations. In other words, PUNC adopts severe restrictions on what is used in the combination process. This short-cut of using the head-knowledge as the basis for the interpretation by-definition ensures that the generated interpretation will already contain plausible information. So, plausibility occurs as a side effect of the combination process.

Informativeness in C^3 is implemented as a distinct stage in the process of constructing interpretations. Once an interpretation has been produced, it is examined for redundant information. For example, if a generated interpretation for *pencil bed* were "a bed that is made of wood", it would be rejected as being uninformative, since *made of wood* is an encoded predicate of *bed*. By contrast, in PUNC, informativeness is implemented as an inherent part of the interpretation process. PUNC does not consider default information (like *made of wood*) as part of its interpretations and so will never produce an uninformative interpretation.

We have already seen that C^3 does not really take familiarity into account in processing compounds. PUNC handles familiarity by by-passing the normal interpretation-production mechanism and directly accessing familiar, lexicalized compounds. Before a new interpretation is produced to a presented compound, the words in the compound are checked against familiar compounds in the knowledge base. Familiar, lexicalized compounds are stored as single units in the knowledge base with hard-coded interpretations that are retrieved instantly "as-is". When there are several familiar interpretations for a given compound all of them are retrieved in a fixed order of preference based on separate empirical analyses of people's interpretation judgements [18]. Each of these interpretations is structured as a sequence of strings in the scope of the head concept. The string sequence contains a number of variable places for diagnostic features, predicates of the head concept bound to fixed values, and a description of the connecting relation between head and modifier. The interpretations are by definition highly plausible and informative. For example, the two interpretations for *bullet train* are represented in our knowledge base as follows:

```
@bullet train:
train[vehicle(hasSpeed_veryfast)=1 {like a bullet}],
train[vehicle(hasShape_streamlined)=2 {like a bullet}]
```

The constituent concepts of familiar compounds are also encoded separately in the knowledge base with diagnostic knowledge associated to them like any other concept, so they can be used as elements for novel combinations as well.

If the compound presented is not found in the knowledge base but there is one or more familiar compounds with the same modifier, a quick construction method is attempted by checking the existence of any semantic relations between the head of the familiar compound(s) stored in the knowledge base and the head of the compound presented. This method tries to create an "interpretation environment" for a new compound by taking into account possible familiarity effects due to concepts semantically-related to it. Familiar idiosyncratic, idiomatic compounds are not considered. If two heads are found to be semantically related, an insertion process takes place. The new head concept replaces the head of the familiar compound(s), and a unification process is initiated. The unification process tries to fill/replace the overlapping, diagnostic variables/values of the head concept of the new compound with those of the head concept of the familiar compound(s), which are bound to concrete values (see again the example of subsection 3.2 for a head insertion and variables unification method between the two symmetrically semantically-related head concepts *train* and *car*.

If the compound is unfamiliar/novel and cannot be related semantically with a familiar compound, an interpretation needs to be constructed from scratch. PUNC does this by meshing the diagnostic knowledge of the head and modifier concept. This meshing of knowledge can happen in different ways, depending on what the diagnostic knowledge of the concept in question and how it can interact with other concepts. For example, the concepts in *cactus beetle* can interact

because *eats things* is something that *beetles* do, and *can be eaten* is something that we know about *plants*. This demonstrates where both the head and modifier can have possible interactions; the information can then be meshed to produce an interpretation like "a beetle that eats cactus". Another such example would be in the compound *idea magazine*. From *magazine* we know that a magazine can be *about* something, so PUNC will produce the interpretation "a magazine that is about ideas". However, an *idea* can also be *about* different things. So, *idea magazine* can be "a magazine that an idea is about".

These examples demonstrate how diagnostic information of the head and modifier concepts can interact to produce different interpretations. Interpretations can be produced even when the interactions between the head and modifier are not two-way, like in the examples above. Consider the combination *cactus magazine*. We already know that one of the most important aspects of the concept *magazine* is that a magazine is generally *about* something. Because a magazine can be *about* something, PUNC produces an interpretation that says *cactus magazine* is "a magazine about cacti". The representation for this interpretation contains all of the diagnostic aspects of *magazine*, which is endowed with the extra knowledge that this particular magazine *is about* cacti. This is an example of where the modifier concept is incorporated into the head concept to form an interpretation.

Interpretations are also produced when specific pieces of knowledge from the modifier are incorporated into the knowledge we have from the head concept. For example, one of the most diagnostic aspects of *cactus* is that it *has spikes*. In the combination *cactus beetle* this knowledge is meshed with the existing representation of *beetle* to produce the interpretation of "a beetle that is spiky". Other interpretations that arise from this meshing of knowledge from the modifier would be "a beetle that is found in the desert" and "a beetle that can conserve water". The compounds are ranked in order of their goodness, which depends on the type of interaction that gives rise to the interpretation and the diagnosticity weightings of the knowledge used to form it. For example, because the interpretation "a beetle that is spiky" uses highly diagnostic information of the modifier it is viewed as a good interpretation, whereas "a beetle that can photosynthesize" is less good, since the information used is less diagnostic of *cacti*.

4.2 Model Testing and Evaluation

For familiar compounds, the model "searches and retrieves" their interpretations. The interpretations have been selected out the most frequent judgments of 25 people ([22], Experiment 3). So, evaluation of the quality of interpretations for familiar compounds retrieved from the knowledge base is not an issue. Also, the system checks for semantic links between concepts that have been predefined according to the intuitions of 3 independent people (with an agreement 90%-100%), and employs, when appropriate, a simple "search and replace" procedure. 93% of the interpretations produced with this method are meaningful.

To examine PUNC's performance for novel, not semantically related phrases, we used two sets of interpretations for novel noun-noun compounds; one from [5] and one set from [18]. In both of these studies, people were asked to provide what they thought would be plausible interpretations for a list of novel compounds. Participants' responses were collated and ranked by their frequency of production. We took the same set of compounds that were presented to those participants (39 in total) and input them into the PUNC model. For each compound PUNC returned an ordered set of interpretations and representations for each interpretation, which we could then compare to participants' responses.

We found that in 77% of cases the most frequently produced interpretation by people was produced by PUNC as the highest ranking interpretation. For example, for the compound *plate paper*, the interpretation that was ranked highest by PUNC was "paper that is used to make plates", which matched the most frequently produced interpretation by the participants. For some compounds, PUNC's highest ranking interpretation was not the interpretation produced most often by people, but was still among the set of interpretations people produced. For example, *bee hat* meaning "a hat that is worn by a bee" was considered the best interpretation by PUNC, but it was only the third most frequently produced interpretation by participants. If we compare all of the interpretations produced by PUNC that were also produced by people, there is a strong correlation between PUNC's goodness score (the lower the goodness score the better the interpretation) and the frequency of production of participants' interpretations (the higher the frequency score the better the interpretation) ($r = -0.74, N = 261, p < .001$).

To examine PUNC's efficiency, the CPU time for algorithms execution was measured on a x86-based PC, with 500 MHz PIII processor, main memory size of 256 MB, running MS Windows 2000 Professional and JDK 1.4.1. We identified three main conditions; interpretations retrieved for familiar composite concepts (Fcc), interpretations produced with a quick construction method when the input phrases were semantically-related to familiar compounds (SRcc), and interpretations generated for novel compounds (Ncc). For each condition, 30 different, random word combinations were processed. The results were recorded and submitted to an one-way ANOVA.

The mean time in milliseconds for processing phrases was: $MD_{Fcc} = 10.43, SD = 1.19$; $MD_{SRcc} = 15.06, SD = 3.88$; $MD_{Ncc} = 56.66, SD = 32.41$. The overall difference across conditions was reliable, $F(2, 87) = 54.68, p < .001$. Fisher's LSD post-hoc test indicated significant difference between the means of Ncc and Fcc ($MD = 46.23, p < .001$). These metrics mirror the results of [22], who report reliably shorter comprehension times due to familiarity. Fisher's LSD post-hoc test indicated also reliable difference between the means of Ncc and SRcc, ($MD = 41.60, p < .001$). These results can probably be explained within the framework of distributed theories of memory, which provide evidence why semantic similarity between two words is a crucial factor for explaining priming effects [23]. This is a topic for future research.

5 Discussion

Constraint theory needs to be developed in several directions. First, it needs to be able to deal with a wider range of empirical phenomena, such as those we have reported here on familiarity. Second, the C^3 model needs to operate more at the algorithmic level in being more efficient in its computa-

tional complexity. The PUNC model is a first prototype that aims to meet these challenges.

It may appear on the surface that there are similarities between the processes employed by the PUNC model and those of the concept specialization model ([19], [2]), but in fact there are radical differences between the two. First, a major shortcoming of the concept specialization (CS) model is that it cannot produce property-based interpretations. CS generates interpretations by filling slots in the head concept with the modifier name; for example, *robin hawk* could be interpreted as "a hawk that preys on robins", by filling the *preys* slot in the schema representation of *hawk* with the modifier name. However, CS does not allow for properties of the modifier to be instantiated into the head representation. This means that while the interpretation "a hawk with a red breast" is readily generated by PUNC, CS cannot produce it. PUNC allows for a much greater role for the modifier and its constituent information, thus having greater scope and variation in the interpretations it produces. Second, the knowledge encoded in CS is very general in nature and does allow for certain aspects of a concept to be more important than any other. Conversely, in PUNC the relative importance of information to a concept (e.g., its diagnosticity) impacts on both the ease with which an interpretation is produced and its overall goodness. PUNC offers a model with far greater scope, yet with processing requirements controlled by the tenets of the Constraint theory.

Future work will be aimed at moving from the current stage of early experimentation to more complete computational solution. This will involve a formal specification of the constraints we want to impose on the system, included the newly introduced familiarity factor, and the interactions between them. Also there are some important points that are not fully elaborated on here. Our framework needs to provide an adequate ontology and knowledge representation of concepts. Inheritance relations, abstract concepts and same-level semantically associated concepts have been partially defined but more work needs to be done in these areas. Certainly, an interesting, relevant issue is to determine how much concept-specific semantic information should be hard-coded in the knowledge base and how much should be computed dynamically by the constraint engine rules.

REFERENCES

[1] D. G. Blasko and C. M. Connine, 'Effects of familiarity and aptness on metaphor processing', *Journal of Experimental Psychology: Learning, Memory, and Cognition*, **19**, 295–308, (1993).

[2] B. Cohen and G. L. Murphy, 'Models of concepts', *Cognitive Science*, **8**, 27–58, (1984).

[3] C. M. Connine, J. Mullennix, E. Shernoff, and J. Yelen, 'Word familiarity and frequency in visual and auditory word recognition', *Journal of Experimental Psychology: Learning, Memory, and Cognition*, **16**, 1084–1096, (1990).

[4] F. J. Costello and M. T. Keane, 'Polysemy in conceptual combination: Testing the constraint theory of combination', in *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*, eds., M. G. Shafto and P. Langley, pp. 137–142, Hillsdale, N.J., (1997). Lawrence Erlbaum Associates.

[5] F. J. Costello and M. T. Keane, 'Efficient creativity: Constraint-guided conceptual combination', *Cognitive Science*, **24**, 299–349, (2000).

[6] F. J. Costello and M. T. Keane, 'Testing two theories of conceptual combination: Alignment versus diagnosticity in the comprehension and production of combined concepts', *Journal of Experimental Psychology: Learning, Memory, and Cognition*, **27**, 255–271, (2001).

[7] S. Coulson, *Semantic Leaps: Frame Shifting and Conceptual Blending in Meaning Construction*, Cambridge University Press, New York and Cambridge, 2001.

[8] U. Eco, *A Theory of Semiotics*, Indiana University Press, Bloomington, 1976.

[9] Z. Estes and S. Glucksberg, 'Interactive property attribution in concept combination', *Memory & Cognition*, **28**, 28–34, (2000).

[10] K. D. Forbus, R. W. Ferguson, and D. Gentner, 'Incremental structure mapping', in *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, eds., A. Ram and K. Eiselt, pp. 283–288, Atlanta, GA, (1994). Lawrence Erlbaum Associates.

[11] B. Franks, 'Sense generation; A "quasi-classical" approach to concepts and concepts combination', *Cognitive Science*, **19**, 441–505, (1995).

[12] C. L. Gagné, 'Relation-based combinations versus property-based combinations: A test of the CARIN theory and dual-process theory of conceptual combination', *Journal of Memory and Language*, **42**, 365–389, (2000).

[13] C. L. Gagné and E. J. Shoben, 'Influence of thematic relations on the comprehension of modifier-noun combinations', *Journal of Experimental Psychology: Learning, Memory, and Cognition*, **23**, 71–87, (1997).

[14] M. A. Gernsbacher, 'Resolving 20 years of inconsistent interactions between lexical familiarity and orthography, concreteness, and polysemy.', *Journal of Experimental Psychology: General*, **113**, 256–281, (1984).

[15] J. A. Hampton, 'Conceptual combinations', in *Knowledge, Concepts and Categories*, eds., K. Lamberts and D. Shanks, 133–159, Psychology Press: MIT Press, Hove, (1997).

[16] M. T. Keane, T. Ledgeway, and S. R. S. Duff, 'Constraints on analogical mapping: A comparison of three models', *Cognitive Science*, **18**, 387–438, (1994).

[17] M.T. Keane and F. Costello, 'Setting limits on analogy: Why conceptual combination is not structural alignment', in *The Analogical Mind: A Cognitive Science Perspective*, eds., D. Gentner, K. J. Holyoak, and B. Kokinov, 287–312, MIT Press, Cambridge, MA, (2001).

[18] D. Lynott and M. T. Keane, 'The effects of context on conceptual combination'. (Manuscript in submission).

[19] G. L. Murphy, 'Comprehending complex concepts', *Cognitive Science*, **12**, 529–562, (1988).

[20] J. T. Sheidlower, 'Principles for the inclusion of new words in college dictionaries', *Journal of the Dictionary Society of North America*, **16**, 32–44, (1995).

[21] S. A. Sloman, B. C. Love, and W-K. Ahn, 'Feature centrality and conceptual combination', *Cognitive Science*, **22**, 189–228, (1998).

[22] G. Tagalakis and M. T. Keane, 'Modelling the understanding of noun-noun compounds: The role of familiarity', in *Proceedings of the European Cognitive Science Conference 2003*, (in press).

[23] S. L. Thompson-Schill, K. J. Kurtz, and Gabrieli J. D. E., 'Effects of semantic and associative relatedness on automatic priming', *Journal of Memory and Language*, **38**, 440–458, (1998).

[24] E. J. Wisniewski, 'Construal and similarity in conceptual combination', *Journal of Memory and Language*, **35**, 434–453, (1996).

[25] E. J. Wisniewski and B. C. Love, 'Relations versus properties in conceptual combination', *Journal of Memory and Language*, **38**, 177–202, (1998).

To Avoid Repeating Checks Does Not Always Save Time

M.R.C. van Dongen (dongen@cs.ucc.ie)[1]
Cork Constraint Computation Centre/Computer Science Department
University College Cork

Abstract. Arc-consistency algorithms are the workhorse of backtrackers that Maintain Arc-Consistency (MAC). This paper will provide experimental evidence that, despite common belief to the contrary, it is not always necessary for a good arc-consistency algorithm to have an optimal worst case time-complexity. To sacrifice this optimality allows MAC solvers that (1) do not need additional data structures during search, (2) have an excellent average time-complexity, and (3) have a space-complexity which improves significantly on that of MAC solvers that have optimal arc-consistency components. Results will be presented from an experimental comparison between MAC-2001, MAC-3_d and related algorithms. MAC-2001 has an arc-consistency component with an optimal worst case time-complexity, whereas MAC-3_d does not. MAC-2001 requires additional data structures during search, whereas MAC-3_d does not. MAC-3_d has a space-complexity of $\mathcal{O}(e + nd)$, where n is the number of variables, d the maximum domain size, and e the number of constraints. We shall demonstrate that MAC-2001's space-complexity is $\mathcal{O}(ed\min(n, d))$. MAC-2001 required about 35% more solution time on average than MAC-3_d for easy and hard random problems, MAC-3_d was faster for 40% of the real-world problems but slower for the remaining real-world problems. Our results are an indication that if checks are cheap then lightweight algorithms like MAC-3_d are promising.

1 Introduction

Arc-consistency algorithms significantly reduce the size of the search space of Constraint Satisfaction Problems (CSPs) at low costs. They are the workhorse of backtrackers that Maintain Arc-Consistency during search (MAC [Sabin and Freuder, 1994]).

Currently, there seems to be a shared belief in the constraint satisfaction community that, to be efficient, arc-consistency algorithms need an *optimal* worst case time-complexity [Bessière *et al.*, 1995; Bessière and Régin, 2001; Zhang and Yap, 2001]. MAC algorithms like MAC-2001 that have an optimal worst case time-complexity require a space-complexity of at least $\mathcal{O}(ed)$ for creating data structures for remembering their support-checks. We shall prove that MAC-2001's space-complexity is $\mathcal{O}(ed\min(n, d))$ because it has to *maintain* these additional data structures. As usual, n is the number of variables in the CSP, d is the maximum domain size of the variables and e is the number of constraints.

We shall provide evidence to support the claim that good arc-consistency algorithms do not always need an optimal worst case time-complexity. We shall experimentally compare five MAC algorithms. The first algorithm is MAC-2001 [Bessière and Régin, 2001]. MAC-2001's arc-consistency component has an optimal $\mathcal{O}(ed^2)$ worst case time-complexity. The second and third algorithms are MAC-3 and MAC-3_d [Mackworth, 1977; van Dongen, 2003a; 2002]. The fourth is a new algorithm called MAC-3_p. It lies in between MAC-3 and MAC-3_d. MAC-3, MAC-3_d and MAC-3_p have a better $\mathcal{O}(e + nd)$ space-complexity than MAC-2001 but their arc-consistency components have a non-optimal $\mathcal{O}(ed^3)$ worst case time-complexity. The fifth and last algorithm is MAC-2001_p. It is to MAC-2001 what MAC-3_p is to MAC-3. Finally, we shall introduce some notation for compactly describing ordering heuristics.

For random and real-world problems and for as far as avoiding the re-discovery of checks is concerned MAC-2001_p and MAC-2001 were by far the better algorithms. For any fixed arc-heuristic and for random problems where checks were cheap MAC-3, MAC-3_p and MAC-3_d were *all* better in clock on the wall time than MAC-2001 and MAC-2001_p, with MAC-3_d the best of all. MAC-2001_p required about 21% more time on average than MAC-3_d, whereas MAC-2001 required about 35% more time. For time and solving real-world problems things were not as clear.

The results presented in this paper are important because of the following. Since the introduction of Mohr and Henderson's AC-4 [Mohr and Henderson, 1986], most work in arc-consistency research has been focusing on the design of better algorithms that do not re-discover (do not repeat checks). This focused research is justified by the observation that, as checks become more and more expensive, there will always be a point beyond which algorithms that repeat will become slower than those that do not and will remain so from then on. However, there are many cases where checks are cheap and it is only possible to avoid re-discoveries at the price of a large additional bookkeeping. To forsake the bookkeeping at the expense of having to re-discover may improve search if checks are cheap *and* if problems become large.

The remainder of this paper is organised as follows. Section 2 is an introduction to constraint satisfaction. Section 3 presents some notation for describing selection heuristics. Section 4 describes related work. Section 5 provides a detailed description of the algorithms under consideration and contains a proof that MAC-2001's space-complexity is $\mathcal{O}(ed\min(n, d))$. Section 6 presents experimental results. Conclusions are presented in Section 7.

2 Constraint Satisfaction

A binary *constraint* C_{xy} between variables x and y is a subset of the cartesian product of the domains $D(x)$ of x and $D(y)$ of y. A value $v \in D(x)$ is *supported* by $w \in D(y)$ if $(v, w) \in C_{xy}$. Similarly, $w \in D(y)$ is supported by $v \in D(x)$ if $(v, w) \in C_{xy}$.

A *Constraint Satisfaction Problem* (CSP) is a tuple (X, D, C),

[1] This work has received support from Science Foundation Ireland under Grant 00/PI.1/C075.

where X is a set of variables, $D(\cdot)$ is a function mapping each $x \in X$ to its non-empty domain, and C is a set of constraints between variables in subsets of X. We shall only consider CSPs whose constraints are binary. CSP (X, D, C) is called *arc-consistent* if its domains are non-empty and for each $C_{xy} \in C$ it is true that every $v \in D(x)$ is supported by y and that every $w \in D(y)$ is supported by x. A *support-check* (consistency-check) is a test to find out if two values support each other.

The *tightness* of the constraint C_{xy} between x and y is defined as $1 - |\, C_{xy} \,| / |\, D(x) \times D(y) \,|$, where $\cdot \times \cdot$ denotes cartesian product. The *density* of a CSP is defined as $2e/(n^2 - n)$, for $n > 1$.

The *(directed) constraint graph* of CSP (X, D, C) is the directed graph whose nodes are given by X and whose arcs are given by $\cup_{C_{xy} \in C} \{ (x, y), (y, x) \}$. The *degree* of a variable in a CSP is the number of neighbours of that variable in the (directed) constraint graph of that CSP.

MAC is a backtracker that maintains arc-consistency during search. MAC-i uses arc-consistency algorithm AC-i to maintain arc-consistency.

The following notation is not standard but will turn out useful. Let $\delta_o(v)$ be the original degree of v, let $\delta_c(v)$ be the current degree of v, let $k(v) = |\, D(v) \,|$, and let $\#(v)$ be a *unique* number which is associated with v. We will assume that $\#(v) \leq \#(w)$ if and only if v is lexicographically less than or equal to w.

3 Operators for Composing Selection Heuristics

In this section we shall introduce notation to describe and "compose" variable and arc selection heuristics. The reader not interested in the nitty gritty details of such heuristics may wish to skip this section and return to it later. Motivation, a more detailed presentation, and more examples may be found in [van Dongen, 2003b, Chapter 3].

It is recalled that a relation on set T is called a *quasi-order* on T if it is reflexive and transitive. A relation, $\prec$, on T is called *linear* if $v \prec w \vee w \prec v$ for all $v, w \in T$. Linear quasi-orders may allow for "ties," i.e. they may allow for situations where $v \prec w \wedge w \prec v \wedge v \neq w$. A quasi-order $\preceq$ is called a *partial order* if $v \preceq w \wedge w \preceq v \implies v = w$ for all $v, w \in T$. An *order* (also called a *linear order*) is a partial order that is also a linear quasi-order. An order $\preceq$ *prefers* v to w if and only if $v \preceq w$.

The *composition* of order $\preceq_2$ and linear quasi-order $\preceq_1$ is denoted $\preceq_2 \bullet \preceq_1$. It is the unique order on T which is defined as follows:

$$v \preceq_2 \bullet \preceq_1 w \iff (v \preceq_1 w \wedge \neg w \preceq_1 v) \vee (v \preceq_1 w \wedge w \preceq_1 v \wedge v \preceq_2 w) .$$

In words, $\preceq_2 \bullet \preceq_1$ is the selection heuristic that uses $\preceq_1$ and "breaks ties" using $\preceq_2$. Composition associates to the left, i.e. $\preceq_3 \bullet \preceq_2 \bullet \preceq_1$ is equal to $(\preceq_3 \bullet \preceq_2) \bullet \preceq_1$.

Let $\preceq$ be a linear quasi-order on T and let $f :: Y \mapsto T$ be a function. Then $\otimes^f_{\preceq}$ is the unique linear quasi-order on Y which is defined as follows:

$$v \otimes^f_{\preceq} w \iff f(v) \preceq f(w) , \qquad \text{for all } v, w \in Y .$$

Finally, let $\pi_i((\, v_1, \ldots, v_n \,)) = v_i$ for $1 \leq i \leq n$.

We are now in a position where we need no more notation. For example, the minimum domain size heuristic with a lexicographical tie breaker is given by $\otimes^{\#}_{\leq} \bullet \otimes^{k}_{\leq}$, the ordering on the maximum original degree with a lexicographical tie breaker is given by $\otimes^{\#}_{\leq} \bullet \otimes^{\delta_o}_{\geq}$, the *Brelaz heuristic* (cf. [Gent *et al.*, 1996]) with a lexicographical tie breaker is given by $\otimes^{\#}_{\leq} \bullet \otimes^{\delta_c}_{\geq} \bullet \otimes^{k}_{\leq}$, and $\otimes^{\#\circ\pi_2}_{\leq} \bullet \otimes^{\#\circ\pi_1}_{\leq}$ is the lexicographical arc-heuristic. As usual, $\cdot \circ \cdot$ denotes function composition.

4 Related Literature

In 1977, Mackworth presented an arc-consistency algorithm called AC-3 [Mackworth, 1977]. AC-3 has a $\mathcal{O}(ed^3)$ bound for its worst case time-complexity [Mackworth and Freuder, 1985]. AC-3 has a $\mathcal{O}(e + nd)$ space-complexity. AC-3 cannot remember all its support-checks. AC-3 uses *arc-heuristics* to repeatedly select and remove an arc, (x, y), from a data structure called a *queue* (a set, really) and to use the constraint between x and y to *revise* the domain of x. Here, to revise the domain of x using the constraint between x and y means to remove the values from $D(x)$ that are not supported by y. AC-3's arc-heuristics determine the constraint that will be used for the next support-check. Besides these arc-heuristics there are also *domain-heuristics*. These heuristics, if given the constraint that will be used for the next support-check, determine the values that will be used for the next support-check. The interested reader is referred to [Mackworth, 1977; Mackworth and Freuder, 1985] for further information about AC-3.

Wallace and Freuder pointed out that arc-heuristics can influence the efficiency of arc-consistency algorithms [Wallace and Freuder, 1992]. Similar observations were made by Gent *et al.* [Gent *et al.*, 1997]. Despite these findings only few authors describe the heuristics that were used for their experiments. We believe that to facilitate ease of replication all information to repeat experiments should be described in full. This includes information about arc-heuristics.

Bessière and Régin presented AC-2001, which is based on AC-3 [Bessière and Régin, 2001] (see also [Zhang and Yap, 2001] for a similar algorithm). AC-2001 revises one domain at a time. The main difference between AC-3 and AC-2001 is that AC-2001 uses a lexicographical domain-heuristic and that for each variable x, for each $v \in D(x)$ and each constraint between x and another variable y it remembers the last support for $v \in D(x)$ with y so as to avoid repeating checks that were used before to find support for $v \in D(x)$ with y. AC-2001 has an optimal upper bound of $\mathcal{O}(ed^2)$ for its worst case time-complexity and its space-complexity is $\mathcal{O}(ed)$. AC-2001 behaves well on average. It was observed that AC-3 is a good alternative for stand alone arc-consistency if checks are cheap and CSPs are under-constrained but that AC-3 is very slow for over-constrained CSPs and CSPs in the phase transition [Bessière *et al.*, 1999; Bessière and Régin, 2001].

We made similar observations in experimental comparisons between AC-7, AC-2001 and AC-3_d, which is a cross-breed between Mackworth's AC-3 and Gaschnig's DEE [Mackworth, 1977; Gaschnig, 1978; van Dongen, 2002]. We did *not* consider search. The only difference between AC-3 and AC-3_d is that AC-3_d sometimes takes two arcs out of the queue and *simultaneously* revises *two* domains with Algorithm $\mathcal{D}$ from [van Dongen, 2001]. A double-support heuristic is a heuristic that prefers checks between two values each of whose support statuses are unknown. For two-variable CSPs the double-support heuristic is optimal and requires about half the checks that are required by a lexicographical heuristic if the domain sizes of the variables are about equal and sufficiently large [van Dongen, 2003a]. AC-3_d and MAC-3_d have a low $\mathcal{O}(e+nd)$ space-complexity. Our results indicated that AC-3_d was promising for stand alone arc-consistency.

5 Description of Algorithms

In this section we shall describe MAC-3_d, MAC-3_p, MAC-2001, and MAC-2001_p in more detail. The presentation is to provide a good understanding of the basic machinery of the algorithms and to highlight the differences between them. We shall also prove that MAC-2001 has a $\mathcal{O}(ed\min(n,d))$ space-complexity.

5.1 MAC-3_d and MAC-3_p

AC-3_d is a cross-breed between AC-3 and DEE [Mackworth, 1977; Gaschnig, 1978]. The only difference between AC-3 and AC-3_d is that if AC-3_d's arc-heuristic select the arc (x, y) from the queue and if the reverse arc (y, x) is also in the queue then AC-3_d will remove both arcs from the queue and and will simultaneously revise *two* domains with algorithm $\mathcal{D}$ described in [van Dongen, 2001; 2003a]. $\mathcal{D}$ uses a *double-support* domain-heuristic, i.e. a heuristic which prefers double-support checks. AC-3_p is a "poor man's" version of AC-3_d; It is not as efficient but easier to implement. It can be obtained from AC-3_d by replacing its call to $\mathcal{D}$ by two calls to Mackworth's $revise$ to *sequentially* revise two domains with one constraint. The difference between AC-3_p and AC-3_d is AC-3_d's double-support heuristic. AC-3_d and AC-3_p inherit their $\mathcal{O}(ed^3)$ worst case time-complexity and $\mathcal{O}(e+nd)$ space-complexity from AC-3. MAC-3_d (MAC-3_p) is implemented by replacing AC-3 in MAC-3 by AC-3_d (AC-3_p). The space-complexity of MAC-3_d and MAC-3_p is equal to $\mathcal{O}(e+nd)$.

5.2 MAC-2001 and MAC-2001_p

Pseudo-code for an arc-based version of AC-2001 and the $revise$-2001 algorithm upon which it depends is depicted in Figures 1 and 2. The "foreach $s \in S$ do $statement$" construct assigns the members in S to s from small to big and carries out $statement$ after each assignment. For the purpose of the presentation of AC-2001 it is assumed that the values in the domains are ordered from small to big. For each variable x, for each value $v \in D(x)$, and for each neighbour y of x it is assumed that $last[x][v][y]$ is initialised to some value that is smaller than the values in $D(y)$.

AC-2001 finds support for $v \in D(x)$ with y by checking against the values in $D(y)$ from small to large. It uses a counter $last[x][v][y]$ to record the last check that was carried out. This allows it to save checks the next time support for $v \in D(x)$ has to be found with y if $last[x][v][y] \in D(y)$. Furthermore, checks are saved by not looking for support with values that are less than or equal to $last[x][v][y] \in D(y)$.

MAC-2001 requires additional data structures during search. It maintains the counter $last[x][v][y]$ to remember the last support for $v \in D(x)$ with $D(y)$. The space-complexity of *last* is $\mathcal{O}(ed)$ [Bessière and Régin, 2001]. It seems to have gone unnoticed so far that MAC-2001 has a $\mathcal{O}(ed\min(n,d))$ space-complexity. The reason for this space-complexity is that MAC-2001 has to *maintain* the data structure $last$. This only seems to be possible using one of the following two methods (or a combination):

1. Save all relevant counters once before AC-2001. Upon backtracking these counters have to be restored. This requires a $\mathcal{O}(ned)$ space-complexity because $\mathcal{O}(ed)$ data structures may have to be saved n times.
2. Save each counter before the assignment to $last[x][v][y]$ in $revise$-2001 and count the number of changes, c, that were carried out. Upon backtracking, restore the c counters in the reverse order. This comes at the price of a space-complexity of $\mathcal{O}(ed^2)$ because each of the $2ed$ counters may have to be saved $\mathcal{O}(d)$ times.

```
function AC-2001( X ) : Boolean;
begin
   Q := { ( x, y ) ∈ X² : x and y are neighbours };
   while Q ≠ ∅ do begin
      select and remove any arc ( x, y ) from Q;
      if not revise-2001(x, y, change_x) then
         return false;
      else if change_x then
         Q := Q ∪ { ( z, x ) : z ≠ y, z is a neighbour of x };
   end;
   return true;
end;
```

Figure 1. Arc-based version of AC-2001.

```
function revise-2001( x, y, var change ) : Boolean;
begin
   change := false;
   foreach r ∈ D(x) do
      if last[x][r][y] ∉ D(y) then
         if ∃c ∈ D(y) s.t. c > last[x][r][y]
                  and c supports r then
            last[x][r][y] := the first such value c;
         else begin
            D(x) := D(x) \ { r };
            change := true;
         end;
   return D(x) ≠ ∅;
end;
```

Figure 2. $revise$-2001.

Therefore, MAC-2001's space-complexity is $\mathcal{O}(ed\min(n,d))$. Christian Bessière (private communication) implemented MAC-2001 using Method 2.

The consequences of MAC-2001's space requirements can be prohibitive. For example, without loss of generality we may assume the usual lexicographical value ordering. Let $n = d > 1$ and consider the binary CSP where all variables should be pairwise different. Finally, assume that Method 2 is used for MAC-2001 (Method 1 will lead to a similar order of space-complexity). Note that the "first" solution can be found with a backtrack free search. Also note that in the first solution i is assigned to the i-th variable. We shall see that MAC-2001 will require a lot of space to solve the given CSP.

Just before the assignment of i to the i-th variable we have the following. For each variable x, for each variable $y \neq x$, and for each $v \in D(x) = \{ i, \ldots, n \}$ we have $last[x][v][y] = \min(\{ i, \ldots, n \} \setminus \{ v \})$. To make the CSP arc-consistent after the assignment of i to the i-th current variable, (only) the value i has to be removed from the domains of the future variables. Unfortunately, for each of the remaining $n-i$ future variables x, for each of the remaining $n-i$ values $v \in D(x) \setminus \{ i \}$, and for each of the remaining $n-i-1$ future variables $y \neq x$, i was the last known support for $v \in D(x)$ with y. This means that $(n-i)^2 \times (n-i-1)$ counters must be saved and incremented during the AC-2001 call following the assignment of i to the i-th variable. In total, MAC-2001 has to save $\sum_{i=1}^{n}(n-i)^2 \times (n-i-1)$, i.e. $(n-2) \times (n-1) \times n \times (3n-1)/12$ counters. For $n = d = 500$, MAC-2001 will require space for at least $15,521,020,750$ counters and this may not be available on every machine. Sometimes MAC algorithms that do not re-discover *do* require a lot of space, even for deciding relatively small CSPs that allow a backtrack free search.

AC-2001$_p$ is to AC-2001 what AC-3$_p$ is to AC-3. If its arc-heuristic selects (x, y) from the queue and if (y, x) is also in the queue then it will remove both and use (at most) two calls to $revise$-2001 to revise the domains of x and y.

6 Experimental Results

In this section we shall compare MAC-2001, MAC-2001$_p$, MAC-3$_d$, MAC-3$_p$ and MAC-3 for random and real-world problems. For the random problems we implemented support-checks as cheap lookup operations in arrays. For the real world problems we implemented support-checks as (more) expensive function calls.

6.1 Implementation Details

All implementations used the same basic data structures as used by MAC-3$_d$. The implementations of MAC-2001 and MAC-2001$_p$ were arc-based. This allowed us to evaluate the algorithms for different arc-heuristics. Previously, we used Christian Bessière's variable based implementation of MAC-2001 [van Dongen, 2003b]. However, Bessière's implementation came with only one arc-heuristic and it was about 17% slower than our own implementation.

All solvers were real-full-look-ahead solvers and to ensure that they visited the same nodes in the search tree they were equipped with the same dom/deg variable ordering heuristic. Using the notation introduced in Section 3 this heuristic is given by $\otimes_{\leq}^{\#} \bullet \otimes_{\leq}^{f}$, where $f(v) = k(v)/\delta_o(v)$. We considered three different arc-heuristics, called lex, $rlex$, and $comp$. Using the notation introduced in Section 3 these can be defined as:

$$
\begin{aligned}
lex &= \otimes_{\leq}^{\#\circ\pi_2} \bullet \otimes_{\leq}^{\#\circ\pi_1}, \\
rlex &= \otimes_{\leq}^{\#\circ\pi_1} \bullet \otimes_{\leq}^{\#\circ\pi_2}, \quad \text{and} \\
comp &= \otimes_{\leq}^{\#\circ\pi_2} \bullet \otimes_{\geq}^{\delta_c\circ\pi_2} \bullet \otimes_{\leq}^{k\circ\pi_2} \bullet \otimes_{\leq}^{\#\circ\pi_1} \bullet \otimes_{\geq}^{\delta_c\circ\pi_1} \bullet \otimes_{\leq}^{k\circ\pi_1}.
\end{aligned}
$$

At the moment of writing $comp$ is the best known arc-heuristic for MAC-3$_d$. Further in this section we shall see that it is also an excellent heuristic for the remaining algorithms. Profiling revealed that arc-selection for MAC-3$_d$ with $comp$ usually takes between 10% and 20% of the solution time, whereas selection with lex hardly takes any time. However, $comp$ has a far better effect on constraint propagation than both lex and $rlex$ and investing in it is well spent. We intend to cut down on the time for arc-selection with $comp$ by supporting it with a special data type for the queue. It is not quite clear *why* this heuristic has such a good effect on constraint propagation. This is something we intend to investigate further.

6.2 Random Problems

Random problems were generated for $15 \leq n = d \leq 30$. We will refer to the class of problems for a given combination of $n = d$ as the problem class with *size* n. The problems were generated as follows. For each problem size and each combination (C, T) of average density C and uniform tightness T in $\{(i/20, j/20) : 1 \leq i, j \leq 19\}$ we generated 50 random CSPs. Next we computed the average number of checks and the average time that was required for deciding the satisfiability of each problem using MAC search. All problems were run to completion. Frost *et al.*'s model B [Gent *et al.*, 2001] random problem generator was used to generate the problems (`http://www.lirmm.fr/~bessiere/generator.html`).

The test was carried out in parallel on 50 identical machines. All machines were Intel Pentium III machines, running SuSe Linux 8.0, having 125 MB of RAM, having a 256 KB cach size, and running at a clock speed of about 930 MHz. Between pairs of machines there were small (less than 1%) variations in clock speed. Each machine was given a unique identifier in the range from 1 through 50. For each machine random problems were generated for each combination of density and tightness. The CSP generator on a particular machine was started with the seed given by 1000 times the machine's identifier. All problems fitted into memory and no swapping occurred. The total time for our comparison is equivalent to more than 100 days of computation on a single machine.

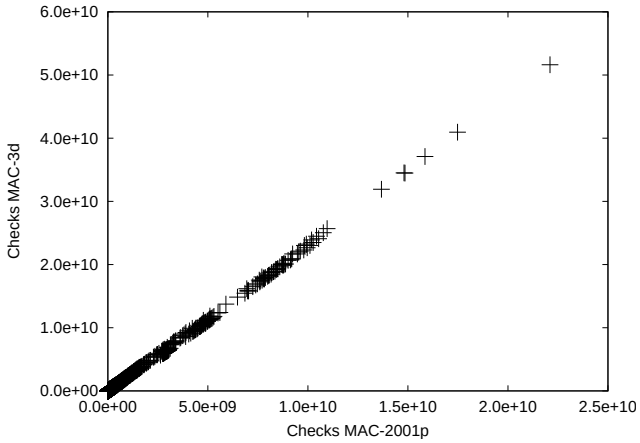


Figure 3. Scatter plot of checks MAC-3$_d$ and $comp$ vs. checks of for MAC-2001$_p$ and $comp$ for search and problem size 30.

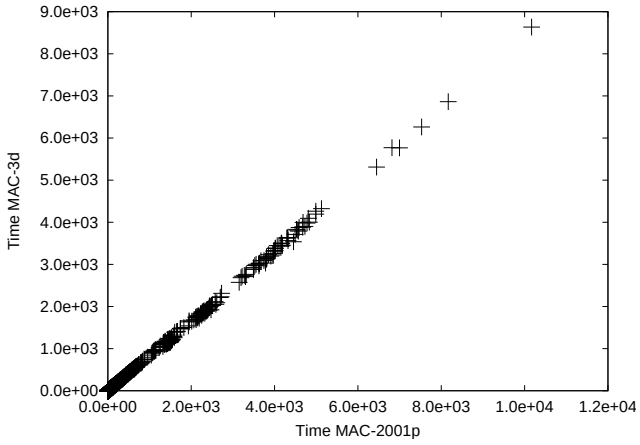


Figure 4. Scatter plot of time MAC-3$_d$ and $comp$ vs. time of of MAC-2001$_p$ and $comp$ for search and problem size 30.

For random problems, the best lightweight algorithm turned out to be MAC-3$_d$ with a $comp$ heuristic. The best algorithm from the MAC-2001 family was MAC-2001$_p$ with a $comp$ heuristic. Figure 3 depicts a scatter plot of the checks required by MAC-3$_d$ with $comp$ versus the number of checks required by MAC-2001$_p$ with $comp$ for problem size 30. Figure 4 depicts a scatter plot of the time required by MAC-3$_d$ with $comp$ versus the time required by MAC-2001$_p$ with $comp$ for problem size 30. Both figures suggest that there is a linear relationship between the number of checks

required by MAC-3_d and MAC-2001_p and between the solution times of MAC-3_d and MAC-2001_p. Similar linear relationships were observed for other combinations of algorithms.

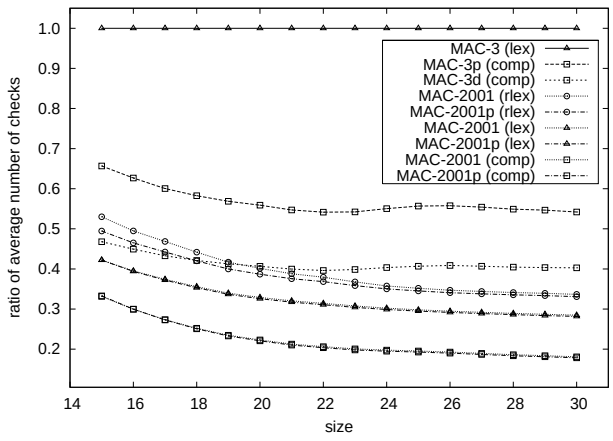


Figure 5. Ratio of average #checks vs. problem size for random problems and search. For each size the average #checks is divided by the average #checks required by MAC-3 with lex.

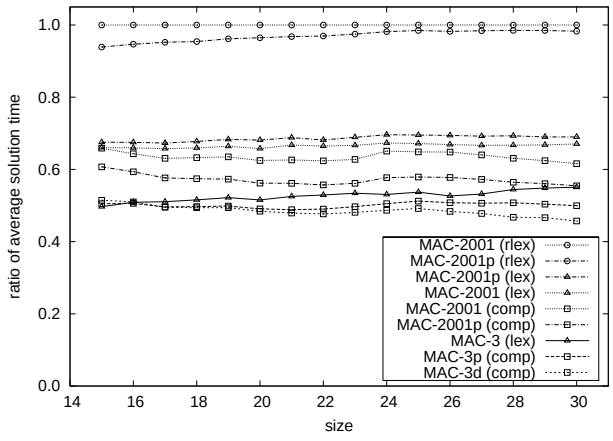


Figure 6. Ratio of average solution time vs. problem size for random problems and search. For each size the average time is divided by the average time required by MAC-2001 with $rlex$.

Figure 5 depicts the ratio between the average number of checks on the one hand and the average number required by MAC-3 with a lex arc-heuristic on the other for problem sizes 15–30 and different combinations of algorithms and arc-heuristics. Similarly, Figure 6 depicts the ratio between the average solution time and the average solution time of MAC-2001 with an $rlex$ arc-heuristic. The order from top to bottom in which the algorithms and heuristics are listed in the legends of the figures corresponds to the height of their graphs for problem size 30. It is difficult to see but what seem to be two lines at the bottom of Figure 5 are two pairs of lines. The pair at the bottom corresponds to MAC-2001 and MAC-2001_p with a $comp$ heuristic. The other pair corresponds to MAC-2001 and MAC-2001_p with a lex heuristic. As the problem size increases the lines for MAC-2001 and MAC-2001_p with an $rlex$ heuristic also seem to converge. MAC-2001_p and MAC-2001 with a $comp$ heuristic are the best when it comes to saving checks.

For problem size 30 the average solution time of MAC-2001_p was about 36.289 seconds, that of MAC-2001 was about 40.294 seconds, and that of MAC-3_d was about 29.910 seconds. On average and over all problems MAC-2001_p required about 21% more time than MAC-3_d, whereas MAC-2001 required about 35% more time.

For any heuristic and for saving time MAC-2001 and MAC-2001_p are never better on average than MAC-3, MAC-3_p and MAC-3_d. Our findings about MAC-3_d are consistent with our previous work [van Dongen, 2002; 2003b]. The results about MAC-2001 and MAC-3 are in contrast with other results from the literature [Bessière and Régin, 2001]. However, this should not be a reason for dismissing these findings; Our testing has been fair and thorough and we cannot recall having seen such comprehensive comparison before. MAC-3 with lex requires about 5 times more checks on average than MAC-2001 and MAC-2001_p with $comp$ but solves more quickly on average. The lack of intelligence in its strategy for propagation does not seem to hinder MAC-3 at all when checks are cheap.

Figures 5 and 6 seem to suggest that as a rule and given one of the algorithms MAC-2001 and MAC-2001_p the heuristic $comp$ was better than lex which, in its turn, was better than $rlex$ both for checks and time. Investigation of the test data revealed that this was true.

For random problems and for clock on the wall time the best algorithm was MAC-3_d with a $comp$ heuristic. MAC-3_p with a $comp$ arc-heuristic was a good second. MAC-3_d's double-support heuristic allows it to improve on MAC-3_p. Overall, the best algorithm from the MAC-2001 family required more than 21% more time on average than MAC-3_d.

6.3 Real-World Problems

The real-world problems came from the CELAR suite [CELAR, 1994]. We did not consider optimisation but only considered satisfiability. The same problems were considered as in [Bessière and Régin, 2001]. These problems have become a sort of a standard benchmark for real-world problems. However, see our comments further on about these problems. For every problem we computed the average solution time over 50 runs. Checks were implemented as function calls and were more expensive than for random problems. For all problems the domain size was equal to 44.

The results for the tests are depicted in Table 1. Due to space-restrictions the results for MAC-2001 have been omitted. MAC-2001 performed about the same as MAC-2001_p but required more checks and time on average. For each problem the least average number of checks and the least average solution time for that problem for all arc-heuristics are printed in bold face. For each of the remaining heuristics the least average number of checks and least average solution time are printed italicised. Again MAC-2001_p is the best algorithm when it comes to saving checks. This time it pays off. MAC-2001_p also seems to be the best algorithm when it comes to solving quickly. MAC-3_d records the least solution time for RLFAP 1 and GRAPH 14. These are the problems for which $ed\min(n,d)$ has the first and third largest size. For GRAPH 9, the problem for which $ed\min(n,d)$ has the second largest size, MAC-3_d also performs well. The larger the problems become, the better MAC-3_d starts to perform relative to the performance of MAC-2001 and MAC-2001_p. It is only for the smaller problems that MAC-2001 and MAC-2001_p are the best. A possible critique is that the density of these problems is rather low—it is always below 2%. It should be interesting to compare the algorithms for larger real-world problems.

For the real-world problems that we considered MAC-2001$_p$ and MAC-2001 are the best algorithms both in time and checks. The sparsity of the constraint graph of these problems suits both MAC-2001 and MAC-2001$_p$ very well. There does not seem to be much between them and MAC-3$_d$ or MAC-3$_p$ with a *comp* heuristic.

					Checks			Time		
Algorithm	Problem	n	e	*lex*	*rlex*	*comp*	*lex*	*rlex*	*comp*
MAC-3	RLFAP 1	916	5548	4.24e+06	4.02e+06	4.17e+06	0.50	0.58	0.59
MAC-3$_p$	RLFAP 1	916	5548	3.89e+06	3.97e+06	3.64e+06	0.47	0.52	0.51
MAC-3$_d$	RLFAP 1	916	5548	2.60e+06	2.67e+06	1.92e+06	**0.380**	0.43	*0.38*
MAC-2001$_p$	RLFAP 1	916	5548	*1.85e+06*	*1.85e+06*	**1.78e+06**	0.38	*0.43*	0.44
MAC-3	RLFAP 11	680	4103	2.90e+08	1.57e+08	5.66e+07	34.12	20.63	7.86
MAC-3$_p$	RLFAP 11	680	4103	2.13e+08	1.42e+08	4.37e+07	25.78	18.60	6.20
MAC-3$_d$	RLFAP 11	680	4103	1.72e+08	1.15e+08	3.09e+07	23.10	16.91	5.35
MAC-2001$_p$	RLFAP 11	680	4103	*3.48e+07*	2.91e+07	**1.04e+07**	*11.74*	*10.25*	**3.87**
MAC-3	GRAPH 9	916	5246	4.43e+06	4.51e+06	3.90e+06	0.54	0.65	0.58
MAC-3$_p$	GRAPH 9	916	5246	4.33e+06	4.48e+06	3.59e+06	0.54	0.60	0.52
MAC-3$_d$	GRAPH 9	916	5246	3.31e+06	3.43e+06	2.18e+06	0.47	0.52	*0.420*
MAC-2001$_p$	GRAPH 9	916	5246	*1.86e+06*	1.87e+06	**1.79e+06**	0.42	*0.46*	0.46
MAC-3	GRAPH 10	680	3907	8.25e+06	8.30e+06	5.68e+06	1.00	1.13	0.85
MAC-3$_p$	GRAPH 10	680	3907	8.08e+06	8.57e+06	5.50e+06	0.98	1.13	0.80
MAC-3$_d$	GRAPH 10	680	3907	7.02e+06	7.49e+06	4.29e+06	0.90	1.05	0.71
MAC-2001$_p$	GRAPH 10	680	3907	*2.67e+06*	2.74e+06	**2.33e+06**	0.60	*0.72*	*0.61*
MAC-3	GRAPH 14	916	4638	3.89e+06	3.95e+06	3.40e+06	0.48	0.56	0.50
MAC-3$_p$	GRAPH 14	916	4638	3.83e+06	3.92e+06	3.09e+06	0.48	0.51	0.45
MAC-3$_d$	GRAPH 14	916	4638	2.87e+06	2.96e+06	1.73e+06	0.41	0.45	**0.34**
MAC-2001$_p$	GRAPH 14	916	4638	*1.65e+06*	*1.65e+06*	**1.59e+06**	0.36	*0.39*	0.39

Table 1. Average results for real-world problems.

7 Conclusions and Recommendations

We compared five algorithms called MAC-2001, MAC-2001$_p$, MAC-3, MAC-3$_p$, and MAC-3$_d$. MAC-2001 and MAC-2001$_p$ have an arc-consistency component with an optimal worst case time-complexity. The remaining algorithms do not. We demonstrated that MAC-2001's space-complexity is $\mathcal{O}(ed\min(n,d))$ and we demonstrated that this size may be prohibitive even for easy CSPs. We compared the algorithms for search and for three different arc-heuristics, called *lex*, *rlex*, and *comp*. We considered random problems where checks are cheap and real-world problems where checks are expensive. For the random problems our findings are that good arc-consistency algorithms do not always need to have an optimal worst case time-complexity. We presented results that suggest quite the opposite. For a given arc-heuristic MAC-2001 and MAC-2001$_p$ always required more solution time than the others. MAC-3$_d$ and a *comp* arc-heuristic, was the most efficient combination when it comes to saving time. MAC-2001$_p$ required about 21% more time on average than MAC-3$_d$ and MAC-2001 required about 34% more. For the real-world problems things were not as clear. Here MAC-2001 and MAC-2001$_p$ were the best in solving quickly but MAC-3$_d$ with a *comp* arc-heuristic was not much worse.

REFERENCES

[Bessière and Régin, 2001] C. Bessière and J.-C. Régin. Refining the basic constraint propagation algorithm. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 309–315, 2001.

[Bessière *et al.*, 1995] C. Bessière, E.C. Freuder, and J.-C. Régin. Using inference to reduce arc consistency computation. In C.S. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, volume 1, pages 592–598, Montréal, Québec, Canada, 1995. Morgan Kaufmann Publishers.

[Bessière *et al.*, 1999] C. Bessière, E.G. Freuder, and J.-C. Régin. Using constraint metaknowledge to reduce arc consistency computation. *Artificial Intelligence*, 107(1):125–148, 1999.

[CELAR, 1994] CELAR. Radio link frequency assignment problem benchmark, `ftp://ftp.cs.city.ac.uk/pub/constraints/csp-benchmarks/celar`, 1994.

[Gaschnig, 1978] J. Gaschnig. Experimental case studies of backtrack vs. Waltz-type vs. new algorithms for satisficing assignment problems. In *Proceeding of the 2nd Biennial Conference, Canadian Society for the Computational Studies of Intelligence*, pages 268–277, 1978.

[Gent *et al.*, 1996] I.P. Gent, MacIntyre E., P. Prosser, B.M. Smith, and T. Walsh. An empirical study of dynamic variable ordering heuristics for the constraint satisfaction problem. In E.C. Freuder, editor, *Principles and Practice of Constraint Programming*, pages 179–193. Springer, 1996.

[Gent *et al.*, 1997] I.P. Gent, E. MacIntyre, P. Prosser, P. Shaw, and T. Walsh. The constrainedness of arc consistency. In *Proceedings of the 3rd International Conference on Principles and Practice of Constraint Programming*, pages 327–340. Springer, 1997.

[Gent *et al.*, 2001] Ian Gent, Ewan MacIntyre, Patrick Prosser, Barbara Smith, and Toby Walsh. Random constraint satisfaction: Flaws and structure. *Journal of Constraints*, 6(4):345–372, 2001.

[Mackworth and Freuder, 1985] A.K. Mackworth and E.C. Freuder. The complexity of some polynomial network consistency algorithms for constraint satisfaction problems. *Artificial Intelligence*, 25(1):65–73, 1985.

[Mackworth, 1977] A.K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8:99–118, 1977.

[Mohr and Henderson, 1986] R. Mohr and T. Henderson. Arc and path consistency revisited. *Artificial Intelligence*, 28:225–233, 1986.

[Sabin and Freuder, 1994] D. Sabin and E.C. Freuder. Contradicting conventional wisdom in constraint satisfaction. In A.G. Cohn, editor, *Proceedings of the 11th European Conference on Artificial Intelligence*, pages 125–129. John Wiley and Sons, 1994.

[van Dongen, 2001] M.R.C. van Dongen. Domain heuristics for arc-consistency algorithms. In *Proceedings of the 12th Irish Conference on Artificial Intelligence and Cognitive Science*, pages 179–188, 2001.

[van Dongen, 2002] M.R.C. van Dongen. AC-3$_d$ an efficient arc-consistency algorithm with a low space-complexity. In P. Van Hentenryck, editor, *Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, volume 2470 of *Lecture notes in Computer Science*, pages 755–760. Springer, 2002.

[van Dongen, 2003a] M.R.C. van Dongen. Domain-heuristics for arc-consistency algorithms. In B. O'Sullivan, editor, *Recent Advances in Constraints*, volume 2627 of *Lecture Notes in Artificial Intelligence*, pages 61–75. Springer, 2003.

[van Dongen, 2003b] M.R.C. van Dongen. Lightweight arc-consistency algorithms. Technical Report TR-01-2003, Cork Constraint Computation Centre, January 2003.

[Wallace and Freuder, 1992] R.J. Wallace and E.C. Freuder. Ordering heuristics for arc consistency algorithms. In *AI/GI/VI '92*, pages 163–169, Vancouver, British Columbia, Canada, 1992.

[Zhang and Yap, 2001] Y. Zhang and R.H.C. Yap. Making AC-3 an optimal algorithm. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 316–321, 2001.