Upper Confidence Bounds for Trees on Demand-Side Management:

Charging Electric Vehicles

by

Ruohua Li, BEng (Hons)

Dissertation

Presented to the

University of Dublin, Trinity College

in fulfillment

of the requirements

for the Degree of

Master of Science in Computer Science

University of Dublin, Trinity College

September 2013

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Ruohua Li

August 30, 2013

Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Ruohua Li

August 30, 2013

Acknowledgments

I would like to express my thanks to Professor Stephen Barrett and Dr. Edgar Galván-López for their valuable guidance and encouragement during my research work. I would also like to thank Mr. PingYuan Zhang, who supplied helpful advice on statistical virtualization.

Finally, I would like to give my most sincere thanks to Mrs. Chaohong Tu and Mr. Jianmin Li who sponsored me for my masters degree.

RUOHUA LI

University of Dublin, Trinity College

September 2013

Upper Confidence Bounds for Trees on Demand-Side Management:

Charging Electric Vehicles

Ruohua Li, M.Sc.

University of Dublin, Trinity College, 2013

Supervisor: Stephen Barrett

Co-supervisor: Edgar Galván-López

Smart grid had become a popular topic in recent years. One big area of smart grid which make it different from the traditional grid is the demand-side management. Many research efforts have been made to improve demand-side management performance using arifitial intellengence, while most of those proposed approaches behaves to be non-real-time. A very recent algorithm UCT, appeared not long from now, has achieved a lot of evidence to make it self one of the best performing real-time optimal solution. In this research, UCT will be tested to deal with real-time demand-side management for its capability on Electric Vehicle charging coordination problem.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

While the prospect of Artificial Intelligence (AI) has long captivated the minds of science fiction writers, the latter half of the twentieth century has witnessed its emergence as one of the most influential fields of computer science.

Early implementations were limited in their complexity, but they demonstrated the potential of AI to become a transformative force in the modern world. However, for many, it is not until the legendary defeat of chess Grandmaster Garry Kasparov by the supercomputer *Deep Blue* [1] that AI confirmed its status as the technology of the future.

The history of electricity distribution extends back beyond the emergence of AI[2], and consequently the technology underpinning this area is more mature. However, despite coexisting for nearly 50 years, the use of AI to address the challenges of electricity distribution has been a much more recent development[3]. Their combination has been motivated by two factors: the limitations of traditional approaches with regards to efficiently meeting the increasing electricity demands of developed countries, and the phenomenal progress in computer performance, which has made AI a viable option for addressing this issue.

The growing use of electricity to power transportation results in one such distribution

problem. Electric vehicles (EVs) must be regularly charged and can place heavy demands on the electricity distribution infrastructure. In particular, sharp peaks in power demand can result as a consequence of many owners charging their vehicles concurrently. Any system that must be capable of handling these peaks in demand must also sacrifice efficiency in its supply of electricity. It is therefore essential to develop efficient strategies for co-ordinating this charging process.

The coordinated charging of EVs may be considered as a *demand-side management* (DSM) problem - the system may be optimised by influencing the charging patterns of the vehicle owners. Developments from the field of AI, and in particular the powerful *Upper Confidence Bounds for Trees* (UCT) algorithm represent one approach to addressing this problem. This motivates the research aim considered in this thesis.

1.1 Research Aim

The aim of this research is to investigate whether UCT can be applied to the EVs' charging coordination problem to deliver efficiency improvements in EV charging infrastructure, and furthermore, to consider its influence on household electricity distribution.

1.1.1 Research question

Is UCT applicable to Electric Vehiclesą́ charging coordination problem, in terms of delivering positive impacts, in PAPR decrement and overall PCSOC[1] achievement.

1.2 Research Approach

In order to address the research aim outlined above, the following steps will be taken:

[1] both are described in chapter 2

1. A survey of the *Smart Grid*, game theory and AI literature, together with previous research relating to this problem will be conducted.

2. The UCT algorithm will be applied to a range of simplistic game theory problems, to investigate its applicability for the task.

3. A system implementation of the UCT algorithm for the EV charging problem will be constructed.

4. A range of simulations will be performed to evaluate the performance of the implementation.

5. An evaluation of the research will be conducted, together with suggestions for future work.

1.3 Research Contribution

The major contributions of this thesis are:

1. **Providing a real-time approach for DSM problems**: There are many approaches dealing with DSM problems but very few of handle analysis in real-time. A successful application of UCT to the EVs charging coordination problem contributes to the real-time DSM field.

2. **Establishing a multi-agent system test-bed for DSM methods study**: Rather than simply building algorithms in a grid simulator, this research applies a distributed server-client model, in which the solution algorithm functions as a plug-in component. This solution is therefore highly extensible, and is easily adjustable to further research.

3. **Determining the limitations of UCT for DSM**: Though UCT has been proven to be robust and effective for certain optimization problems, this research describes some of the limitations to the algorithm.

1.4 Dissertation Outline

The structure of this thesis is as follows. *Chapter 2* offers a description of the state-of-the-art of current smart grid research, particularly with respect to DSM technology, as well as a perspective on solutions that utilize game theory. *Chapter 3* provides an in-depth discussion of the UCT algorithm. Details of how UCT is implemented in the EV chargers coordination solution are given in *chapter 4*. *Chapter 5* describes the experiments conducted to evaluate the implementation. Results are presented and analyzed in *chapter 6*. Finally, in *chapter 7*, conclusions will be drawn from the research as a whole.

Chapter 2

State of the Art

In this chapter, an overview of demand-side management (DSM) and game theory is provided. A survey of applications of game theory is then given, followed by a review of implementations that have been used to address issues similar to the research problem considered in this thesis.

2.1 Smart Grid and Demand-Side Management

2.1.1 Micro Grid and Smart Grid

A *micro grid* is *"a localized grouping of electricity generation, energy storage and loads that normally operates connected to a traditional centralized grid (macro grid)"*[4]. As yet, there is no universally accepted definition as to what constitutes a smart grid, but for the purposes of this dissertation the following publication by the U.S. Department of Energy (DOE) [5] explains that a smart grid uses *"digital technology to improve reliability, security, and efficiency of electric system: from large generation, through the delivery systems to electricity consumers and a growing number of distributed-generation and storage resources."*

In [6] Christian gives the following reasons of why smart grid is a necessity to meet today's requirements:

- The average age of a power plant is 35 years. [1].
- Electricity delivery is highly inefficient (35%) [2].
- The electricity grid must be able to handle the transition between peak and off-peak demand. Consequently, to guarantee the supply of electricity, extra energy must be pushed onto the grid, resulting in inefficiency.

In general, a smart grid's construction must consist of three basic components[7]: a robust and flexible topology, a standardized integrated communication system, an advanced metering infrastructure and intelligent demand-side management.

2.1.2 Demand-Side Management

The Electric Power Research Institute (EPRI) first introduced Demand-side management (DSM) in the 1980s [8]. In this report, a DSM program is defined to consist of the *planning*, *implementing*, and *monitoring* activities of electric utilities. It is also applied in cases where the objective is to persuade consumers to modify their level or manner of electricity consumption.

In general, there are three forms of DSM:

- *Energy Efficiency*: Energy efficiency is typically achieved by increasing the efficiency of energy distribution and appliances.
- *Demand Response*: Methods to reduce, smooth or shift peak demand. It includes all intentional modifications to consumption patterns of electricity of end-use customers that are intended to alter the level of instantaneous demand, the time of electricity usage, or the total consumption.[9]
- *Dynamic Demand*: Dynamically shift appliances usage forward or backward. By observing the power plant's supply, appliance consumption can be modified to reduce

[1] Troxell, Wade O. ąřSmart Grid: Transforming the US Power Grid.ąś Powerpoint Presentation. Undated. Accessed from `http://www.nrel.gov/visitors\_center/` in April 2013

[2] IBID

critical power mismatches. The magnitude of this shifting only advances or delays the appliance consumption by a few seconds, to reduce the impact on the users' experience. A (now-lapsed) patent based around this idea was issued by power systems engineer Fred Schweppe in the United States, in 1982.[10]

2.1.3 Metrics for DSM evaluation

Peak-to-Average Power Ratio

The Peak-to-Average Power Ratio (PAPR) is the ratio of the highest power load to the average power load over a time interval. It has been regarded as a key metric in many DSM studies [11], [12], [13], [14], [15]. *Formula 2.1* illustrates how the PAPR is calculated [16], where the power $x(t)$ is a function of time t. M is the number of periods and $\tilde{T}$ represents the time for one period.

$$\text{PAPR} = \frac{\text{peak power of } x(t)}{\text{average power of } x(t)} = \frac{\underset{0 \le t \le M\tilde{T}}{max} |\, x(t)\,|^2}{\frac{1}{M\tilde{T}} \int_0^{M\tilde{T}} |\, x(t)\,|^2 dt} \tag{2.1}$$

Assuming that electricity conforms to a rectangular waveform, the the formula above can be transformed to:

$$\text{PAPR} = \frac{\underset{m=0,1,\ldots,M-1}{max} |\, \tilde{x}_m\,|^2}{\frac{1}{M} \sum_{m=0}^{M-1} |\, \tilde{x}_m\,|^2} \tag{2.2}$$

State-of-charge

State-of-charge (SOC) represents the electricity remaining in a battery (0%: "empty", 100%: "full"). In the case of EV charging it can be used as an indicator for how effectively the charging approach satisfied the user's demand.

Cost

Cost is another popular metric for evaluating the performance of DSM approaches. The concept is simple, but it can be applied in a variety of ways.

2.1.4 Goals of DSM

There are many reasons for implementing DSM in grids [17]. For a utility company, DSM can contribute to a lower PAPR, which decreases the cost of electricity *generation* and *distribution*. For a government DSM can result in increased energy efficiency, representing a more environmentally friendly policy. Consumers can save money when DSM schedules are used to control appliances' usage and adjust these to low-cost periods. Therefore, DSM represents a valuable approach in the implementation of modern day grids.

DSM has the following 4 goals [18]:

- *Minimize maximum power demand*: Limit the maximum power load to achieve PAPR decrement.
- *Maximize profits*: These profits includes utility companies' profits, governments' profits and the cost to consumers.
- *Minimize loss*: The loss refers to the dissipation that occurs during electricity distribution. It typically depends on the voltage strength and the medium for transmission.
- *Minimize discomfort to the consumer*: DSM should not impact the experience of consuming electricity

2.1.5 Challenges in DSM

In [19] and [20] Goran Strbac et al. described some of the major challenges involved in the development of DSM.

Firstly, the current grid infrastructure is limited in its ability to deal with communications and information gathering, both of which are necessary for a DSM system. Thus, implementing a DSM system would require a significant investment into an upgrade of the grid infrastructure.

Secondly, the difficulty of convincing the public of the benefits of DSM should not be underestimated, in terms of both cost and the use of AI. Both infrastructure changes and high-performance computers are major costs in building such a system. There is currently a lack of empirical evidence of the benefits of DSM, that could be used to convince consumers of its value.

Thirdly, a lack of industry standards slows down the development of DSM. The research domain of DSM needs common metrics or guidelines to follow to create a consistent framework, rather than the assortment of benchmarks that currently exist.

2.1.6 Electric Vehicles and DSM

There are a number of differences between Electric Vehicles (EVs) and traditional appliances (e.g. washing machines, refrigerator etc.). EVs have considerably higher electricity consumption, making it a very significant component in the grid environment. If EVs are managed without DSM, they can lead to severe load management difficulties. In [21], the authors explained that EVs can not only act as loads, but also sources(i.e. storage of energy) due to their high electricity capacity, adding further complexities to the problem.

2.2 Approaches in Demand-side Management

In order to achieve the objectives of DSM, several approaches have proposed. Amir-Hamed Mohsenian-Rad et al. in [22] described several such approaches, described below.

2.2.1 CPP

One method, known as *critical-peak pricing* (CPP), involves a strategy in which utility companies set a very high price (usually 3-10 times higher) for critical peak hours, to discourage consumers from using energy in that period.

2.2.2 ToUP

In *Time-to-Use Pricing* (ToUP), the electricity prices are set in advance and maintained over a long period, usually 1 to 2 years. Customers know in advance how they will be billed in relation to their usage.

2.2.3 RTP

Real-time pricing (RTP) also aims to have different prices at different hours. However, it differs from previous approaches in that prices in ToUP are predefined, while in RTP, real time analysis and adjustments can occur.

2.2.4 DLC

In direct load control (DLC), appliances are grouped into two types: shiftable and non-shiftable. Changes are affected by transferring control of the shiftable appliances to the DLC system, which can supply electricity when needed (e.g. lighting, thermal comfort equipment etc.). However, one major concern with regard to this approach is the privacy of the consumer.

2.2.5 Comparison

CPP, DLC and ToUP are comparatively simple to implement but do not perform as well as RTP which can make adjustments in real time. However, each of DLC and RTP have their own drawbacks. DLC directly controls shiftable appliances, which may impact the consumer experience of electricity consumption. On the other hand, RTP avoids this kind of experience, but it requires consumers to actively cooperate in shifting the electricity consumption. Most consumers cannot frequently respond to prices hour by hour. As a result, it cannot achieve the same performance as DLC.

In [22], the authors construct a model quite unlike the traditional DSM strategies that focus their attention on the interaction between the utility company and each user. They divide the appliances into two groups: shiftable (dishwashers, PHEVs etc.) and non-shiftable (air conditioner, refrigerator etc.). Shiftable appliances are controlled by an energy consumption scheduler (ECS). They then formulate two optimization problems: Peak-Average Ratio (PAR) minimization and energy cost minimization. Since in our problem, the EVs are the only type of appliance, and each is shiftable this model cannot be directly applied to the scenario.

2.3 Game Theory

In [23] Walid Saad et al. gave a review of fundamental game theoretic methods that can be applied to the Smart Grid problem. Game theory is a mathematical framework that can be classified into two categories: Noncooperative Game Theory and Cooperative Game Theory.

2.3.1 Noncooperative Game Theory

Noncooperative game theory can be used to analyze decisions to resolve conflicting interests that exist among all players. Each of their decisions leads to an action that will the actions of other players.

Noncooperative games does not necessarily preclude player cooperation. Players may cooperate, but any cooperation that arises must be self-enforced to maximize its benefit.

There are two categories in noncooperative games [24]: *static* games and *dynamic* games. In static games, the choices of other players will not affect the decision made by the current player. In dynamic games, a player may play more than once and the information shared between players at a certain level, through which players can decide whether to change their course of action.

The objective of noncooperative game theory is to provide strategies for optimization problems when players are making choices independently until a Nash equilibrium is reached, in which no player can improve its utility through a strategy change.

2.3.2 Cooperative Game Theory

In noncooperative games, players are unable to communicate with each other. However, in cooperative games [25], communication is allowed, facilitating the transfer of side payments such as shared utilities. In this respect, cooperative game theory provides a framework that describes how players act when they decide to cooperate with communication. Cooperative games enable analysis of how a single player can contribute to the entire group of players, to improve his own position in the game.

Cooperative game theory may divided into two parts: Nash bargaining and coalitional

game. Nash bargaining considers those scenarios in which a number of players need to come to an agreement on the terms under which they cooperate. Coalitional game theory deals with the formation of cooperative groups or coalitions. Cooperative game theory in both these two branches provides tools that allow the players to decide which other players are best to cooperate with, and associates several cooperation incentives and fairness rules for this task.

2.3.3 Learning in Games

Learning algorithms play an important role in enabling players to learn during the playing of the game, and to achieve the desired game outcome. A pre-defined strategy can be prepared for players to reach a certain equilibrium. Typically, a learning procedure is an iterative process including[26]:

1. Observing the current state of the environment.
2. Using simulation, evaluation functions or environmental trials to estimate the utility distribution.
3. Updating the strategy based on the utility distribution estimated in step 2.

Many learning algorithms have been proposed in [27],[28],[26], [29],[30],[31]. The simplest algorithm, *best response dynamics*, uses iterations of making a selection of the max utility strategy (i.e. best response) . It has the benefit of a simple implementation, but it suffers a number of drawbacks. Firstly, the best response process can only maximize some utility functions [29]. Secondly, the best response decision is very sensitive to the initial state of the environment, thus different initial state will lead to different equilibriums. Thirdly, since different equilibrium exist for the best response process, it cannot guarantee

the equilibrium it reaches is the most efficient one[29].

As a consequence of this, a number of more advanced learning algorithms have been developed for learning the equilibria of game theoretic models. Generally they can be divided into the following categories[23]:

- *Fiction play*: Fiction play is a class of iterative learning algorithms in which each player observes the action of all other players and chooses a certain action by computing the empirical frequency. Researchers have shown that in some special class of games, such as zero-sum games, a Nash equilibrium can always be reached[26]. More recent research has also proposed a number of enhancements to fictitious play algorithms[26], [29],[30].
- *Regret matching*: Best response and fictitious play algorithms aim to allow a player to achieve the maximum utility. Regret matching algorithms are algorithms that minimize the regret of choosing a certain action. Some such algorithms have described in detail in[26], [29], [31].
- Others: Many other classes of learning algorithm are also used in game-theoretic scenarios, such as reinforcement learning or stochastic learning [26].

2.3.4 Game theory in DSM

Traditional demand-side management approaches, such as those mentioned previously (CPP, ToUP, RTP and DLC) are candidates for game theory application. A number of pieces of research have been conducted on the application of game theory in DSM.

In [18] ,[32] and [33] Mohsenian-Rad et al. models the DSM problem as a consumption scheduling game. All consumers are players and they schedule their electricity usage to reduce the peak-to-average power ratio (PAPR), and the total cost (according to CPP or ToUP). In contrast to many of the demand-side management programs, it distributes the

computation into different agents and the environment is regarded as a non-cooperated game.

2.4 Agent-based approaches in DSM

Agent-based models are widely use in studying DSM problems, [34] and [35] are two examples of this approach. In [35] M. Pipattanasomporn et al. consider how multi-agent systems should be designed and implemented.

2.5 Monte Carlo methods in DSM

Monte Carlo methods have been seldom used in DSM problem, but a few examples exist. Ming Zhou et al.[36] used a sequential Monte Carlo simulation to improve the Available Transfer Capability (ATC) of a system.[3]

Another research [38] used component failures and load curve models to allow Monte Carlo simulation to do the evaluation on each state change.

[3] a measure of the transfer capacity remaining in the physical transmission network for further commercial activity over and above already committed uses [37]

Chapter 3

Upper Confidence Bounds for Trees

On June 5, 2013, computer program Zen, applying the UCT algorithm, defeated Takuto Ooomote (9 dan) in a 19×19 Human-Computer Go Challenge [1]. [39] This success was not only a milestone in Human-Computer Go Challenge history, but also a strong prove that UCT is robust enough for problems with high complexities.

Upper Confidence Bounds for Trees (UCT) is a Monte Carlo Tree Search (MCTS) approach for decision optimization in artificial intelligence problems.

This chapter will focus on discussing the foundational basis for the UCT algorithm and the algorithm itself in detail. In *section 3.1*, some of the primary concepts of UCT will be introduced. Then a detailed explanation of UCT will be provided in *section 3.2*. *Section 3.3* will give an overview as to the types of problems for which UCT is most suited. Finally, *section 3.4* will justify the use of UCT in addressing DSM problems, and in particular, the EV charging coordination problem.

[1] Go is a highly strategic board game. It has received extensive interest from the research community, due to the challenge of creating a program that can handle its complexity. Zen defeated Ooomote with a 3 stone handicap. The 9 dan is the highlest level attainable by a Go player.

3.1 Foundations

3.1.1 Markov Decision Processes

Markov Decision Processes (MDPs) provide a mathematical framework for modeling multiple stage decision making in scenarios in which state shift occurs and uncertainty exists.

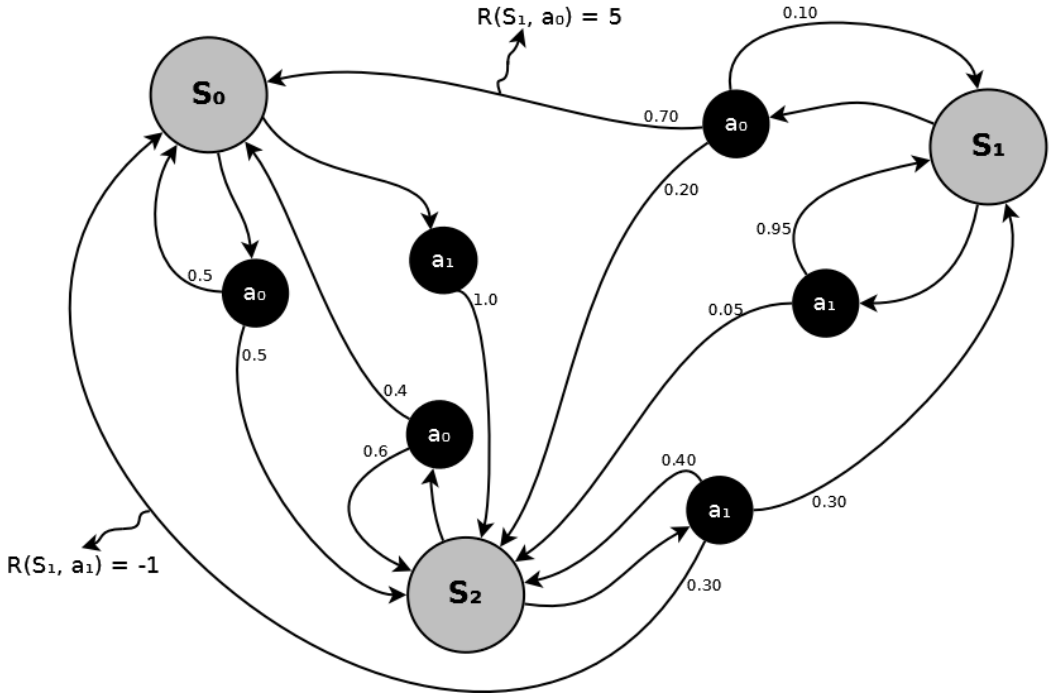


Figure 3.1: A Simple Markov Decision Process.

More precisely, a MDP is a discrete-time stochastic control process and consists of a state set S including an initial state s_0, a finite set of possible actions A, a finite set of possibilities $P_a(s,s')$, $a \in A$ and each $p \in P$ represents the possibility to take the action a that will result in a state change from s to s' and an immediate reward function $R(s,a)$ which returns a real-valued reward by a given state s and a valid action a with respect to that state.

3.1.2 Reinforcement Learning

Reinforcement Learning, an extension of MDPs, is a tool that has seen a great deal of interest in the field of machine learning. *Figure 3.2* demonstrates a typical Reinforcement

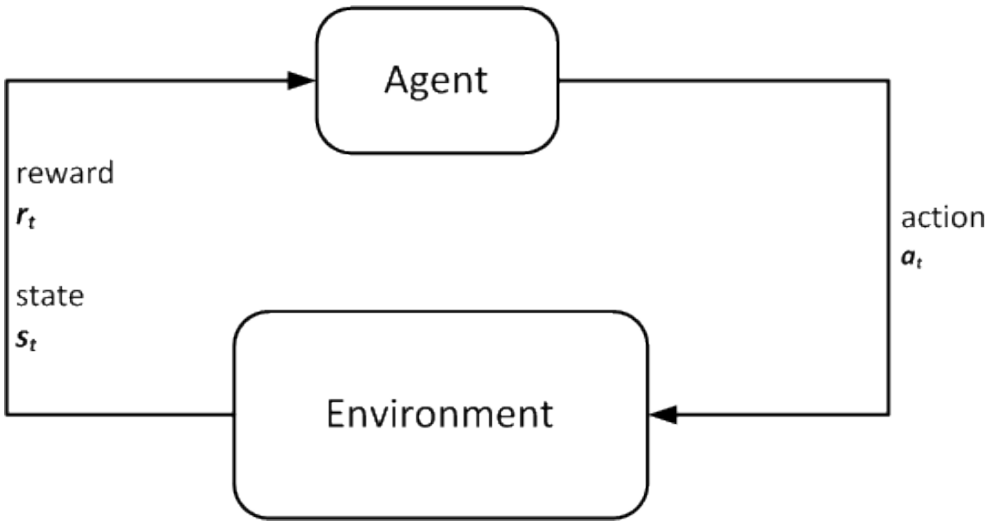


Figure 3.2: Reinforcement Learning.

Learning model. The agent attempts each action a_t at time t in the environment. The environment returns a reward r_t and a state s_t in response to a_t. As time progresses, the agent gains experience.

Exploration

An exploration mechanism plays a very important role in the reinforcement learning agent. It provides the system with a strategy to estimate the solution distribution, leading to better performance than random selection.

3.1.3 Monte Carlo method

Monte Carlo methods are a category of methods that use random sampling to estimate numerical results. They are regularly used in physical and mathematical problems where a deterministic evaluation method is impossible or infeasible to apply. Though Monte Carlo methods vary, most of them follow a distinct pattern:

(i) Define a domain of inputs.

(ii) Generate inputs randomly from the domain.

(iii) Perform a computation on the inputs.

(iv) Collect statistical results.

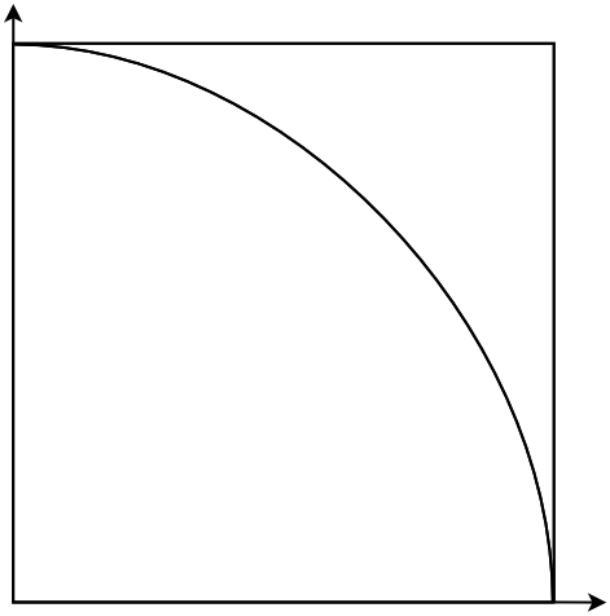


Figure 3.3: Count π problem.

An example application of Monte Carlo methods can be seen in the *Count* π problem. In this problem, the domain consists of the set of points (x, y), for which $x \in (0, r)$ and $y \in (0, r)$. A set of randomly generated points over this domain are used as inputs. For each input, we calculate if the point is located inside the circle quadrant (with distance to the origin less than r, or outside the circle quadrant (with distance to the origin greater than or equal to r). Finally, by computing the ratio of points inside the circle quadrant to outside, we can make an estimate for the value of π with the following equation:

$$\frac{\text{number of points inside}}{\text{number of points outside}} \approx \frac{\text{area inside}}{\text{area outside}} = \frac{\frac{1}{4}\pi \times r^2}{r^2} = \frac{\pi}{4} \tag{3.1}$$

As number of points increase, this formula approaches the true value of $\frac{\pi}{4}$, facilitating a calculation of the value of π.

3.1.4 Bandit-based Method

Bandit problems are those in which a sequential series of decisions must be made, each of which is chosen amongst K actions (e.g. K-armed bandit slot machine, the namesake of the method) in order to maximise the payoff by continuously taking optimal actions.

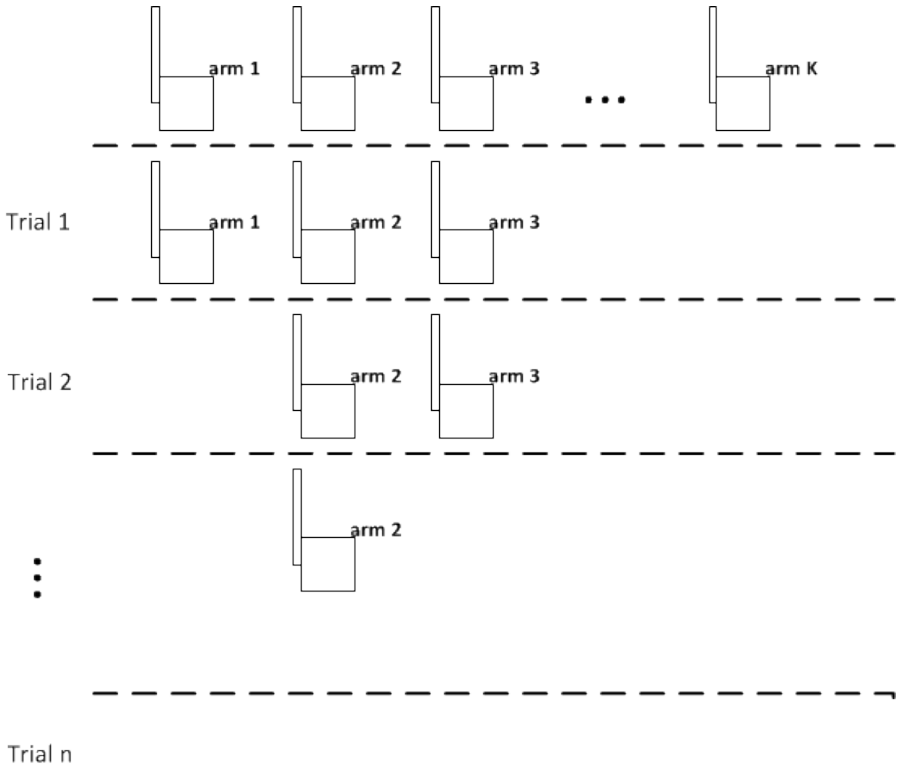


Figure 3.4: Bandit-based problem.

The reward distributions are not known in advance for bandit problems. Consequently, the potential rewards for each action have to be estimated on the basis of previous observations.

Figure 3.4 illustrates a K-armed bandit. Each of the K arms has been played a different numbers of times and there is a reward associated with playing that arm. At each attempt, a *policy* is applied to choose which arm to play, based on the current rewards and number of times each arm has been played.

Exploration vs. Exploitation

A natural tension exists between the exploitation & exploration approaches to estimating potential rewards. Exploitation favours actions with currently good rewards, while exploration favours actions whose rewards could be lower at present, but may overtake other actions in the longer term.

Policy

In [40] Browne, C.B et al. used the following formula to define a policy, which aims to minimise the player's *regret*:

$$R_N = \mu^\star n - \mu_j \sum_{K}^{j=1} \mathbb{E}[T_j(n)] \tag{3.2}$$

where $\mu^\star n$ is the maximum possible reward in the first n trials and $\mu_j \sum_K^{j=1} \mathbb{E}[T_j(n)]$ is the total number of plays for all arms in the first n trials. Thus R_N describes the loss caused by not playing the best arm. To avoid playing a sub-optimal arm due to temporarily promising rewards, it is important to apply a upper confidence bound (UCB) on the observed rewards.

Lai and Robbins in [41] showed that no policy can grow at a rate lower than $O(\ln n)$. The simplest UCB policy for deciding which arm to play is proposed by Auer et al.[42], which produces a logarithmic growth of regret over n:

$$UCB1 = \overline{X}_j + \sqrt{\frac{2lnn}{n_j}} \tag{3.3}$$

where $\overline{X}_j$ is the average reward that arm j has achieved, n_j is the number of times arm j has been played and n is the total number of times all arms have been played. The first term $\overline{X}_j$ drives the policy towards the higher-reward arms which represent *exploitation*, while the second term $\sqrt{\frac{2lnn}{n_j}}$ drives the policy towards the higher-potential arms that have been played fewer times, which represent *exploration*.

3.1.5 Monte Carlo Tree Search

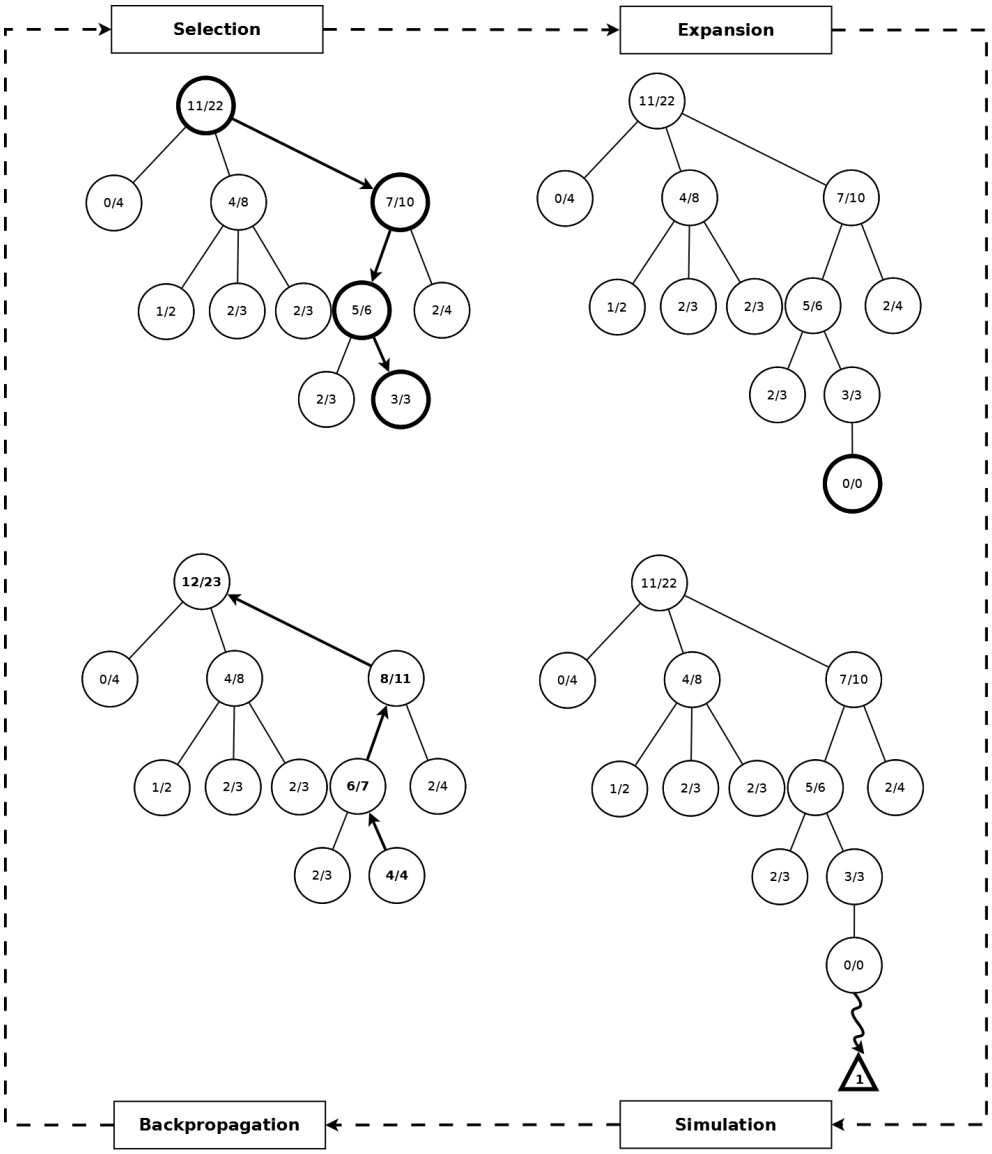


Figure 3.5: Monte Carlo Tree Search.

Monte Carlo Tree Search (MCTS) is a class of algorithms that apply Monte Carlo methods in search trees. Two key mechanisms are necessary in a MCTS algorithm: A *tree policy* to pick the best move according to the previous observations and a *default policy* for random simulation to estimate the value of taking that move. The algorithm makes progress by building a partial game tree using these two mechanisms. As the tree contains more

Algorithm 1 A General MCTS approach

```
1: function MctsSearch(s_0, N)
2:     create root node n_0 with initial state s_0
3:     for i=1 to N do
4:         n_i ←TreePolicy(n_0)
5:         Δ ←DefaultPolicy(s(n_i))
6:         Backup((n_i, Δ))
7:     end for
8:     return a(BestChild(n_0))
9: end function
```

experience, the value distribution of all moves get closer to the real reward distribution.

The idea of MCTS is to operate on an assigned budget (typically in time or memory) to build a search tree in which each node in the tree represents a state of the problem domain. The child nodes represent the possible actions that can be chosen leading to subsequent state changes. When this budget has been used up, the action yielding the greatest utility at the root is returned.

In general, there are four steps in each of the MCTS iterations [43], shown in *figure 3.5*:

1) *Selection*: Starting from the root node, a tree policy is applied to decide which child node to choose recursively until a node that is not fully expanded is reached.

2) *Expansion*: Depending on the strategy, one or several nodes will be created as the child nodes of the node selected in *selection* step.

3) *Simulation*: On each expanded node, a simulation is performed using the *default policy* to reach a predefined terminal state and achieve an outcome.

4) *Backpropagation*: The outcome of the simulation will be added to every node in the path which was chosen by the *tree policy*.

Algorithm 1 summarises these four steps in pseudocode. The root node n_0 corresponds to the initial state s_0, v_i is the final node selected by the *tree policy*, and Δ is the reward

achieved by playing the default policy from state s_i of node n_i. The BACKUP function implements the backpropagation, updating information of nodes in the *tree policy* path. After playing N times, the result function $a(\text{BESTCHILD}(n_0))$ will be used to return the best action directly from root node n_0.

3.2 Upper Confidence Bounds for Trees (UCT)

In this section the best known member of the MCTS family, the UCT algorithm, will be discussed in detail.

3.2.1 The algorithm

As discussed in the previous section, MCTS iteratively uses *Selection*, *Expansion*, *Simulation* and *Backpropagation* to build the search tree, supplying the player with statistical observations of the reward distribution. To select an action, the best action for the root node n_0 will be chosen. One of the advantages of MCTS over other algorithm groups is the use of the *tree policy* for action selection, offering performance gains over *Brute force* and *Greedy* algorithms.

Kocsis et al. in [44] proposed the use of UCB1 (3.3) as an MCTS *tree policy*, due to its well-balanced formulation between exploration and exploitation. By introducing a constant factor C_p, UCT, the value to be maximized, can be defined as follows:

$$UCT = \overline{X}_j + 2C_p\sqrt{\frac{2lnn}{n_j}} \tag{3.4}$$

where C_p adjusts the *tree policy* behaviour in favour of a high reward node (exploitation) or a high potential node (exploration). It is worth noting that an initial value of $n_j = 0$ results in a UCT value of ∞, which ensures that all children of a node will be played at least once before any child has been expanded.

Algorithm 2 The UCT algorithm

function UCTSEARCH(s_0, N)
    create root node n_0 with initial state s_0
    for i=1 to N **do**
        $n_i \leftarrow$TREEPOLICY(n_0)
        $\Delta \leftarrow$DEFAULTPOLICY($s(n_i)$)
        BACKPROP((n_i, Δ))
    end for
    return a(BESTCHILD(n_0))
end function

function TREEPOLICY(n)
    while n has possible action **do**
        if n not fully expanded **then**
            return Expand(n)
        else
            $n \leftarrow$BESTCHILD(n, C_p)
        end if
    end while
    return v
end function

function EXPAND(n)
    choose action a from actions set $A(s(n))$
    create a child node n' to n ▷ $s(n') = f(s(n), a)$ and $a(n') = a$
    return n'
end function

function BESTCHILD(v,n)
    return $\underset{n' \epsilon \text{ children of } n}{\arg max} \left(\frac{Q(n')}{N(n')} + C\sqrt{\frac{2lnN(n)}{N(n')}} \right)$
end function

function DEFAULTPOLICY(s)
    while s is not a terminal state **do**
        choose action a from actions set A(s) randomly
        $s \leftarrow f(s, a)$
    end while
    return reward of s
end function

function BACKPROP(n, Δ)
    while n is not null **do**
        $Visit(n) \leftarrow Visit(n) + 1$
        $Score(n) \leftarrow Score(n) + \Delta(n)$
    end while
end function

The UCT algorithm (borrowed from [40]) is shown in pseudocode in *Algorithm 2*. The state $s(n)$, the total reward $Score(n)$, the number of visits $Visit(n)$ and the actions leading to the state transition $a(n)$ are four attributes of every node n. The UCTSEARCH function applies the approach of the general MCTS algorithm (defined in *Algorithm 1*).

3.2.2 Uses of UCT

Section 3.1 and Section 3.2 suggest that in theory, UCT can be applied generally in problems that can be described as MDPs, using Reinforcement Learning. Such problems can be modeled using MCTS search trees with nodes containing the four attributes outlined above. Importantly, UCT's adjustable feature between exploitation and exploration benefits problems of high complexity (since then for each UCT node, the set of actions is vast).

3.3 The Potential of UCT in DSM

3.3.1 Advantages

Real-time

As is described in *Chapter 2*, Direct Load Control (DLC) requires the agent to be able to directly shift the consumption of managed appliances in real-time. It should be noticed that a real-time control doesn't require the solution to do the major analysis in real-time. There exist many non-real-time approaches which can react immediately or shortly after a minor computation by having a pre-calculated profile generated before run-time.

However, in certain scenarios, pre-calculation of such a profile is infeasible, typically when a change in environment occurs, for example, when there is a change in the grid structure or the arrival of new types of appliance. In such cases, an algorithm which can make changes in real-time should be introduced.

UCT is a perfect fit for these scenarios. It needs only the current state of the environment, a model for how the environment could change, and an objective, in order to make a decision.

Adjustability

By altering the value of C_p, the characteristics of the UCT algorithm can be adjusted to favour *exploitation* or *exploration* in the actions chosen. The *default policy* can be changed to fit the computational budget (the complexity of the *default policy* will determine both the speed of simulation and the precision with which the reward is calculated). The reward function $f(s,a)$ represents the objective of the problem (i.e. a win in a game).

Chapter 4

Implementation

In this chapter, the model for the EV charging scenario is introduced. The system architecture and business logic implementation details of UCT are then described in detail.

4.1 Scenario

The scenario modeled for the first experiment is shown in *figure 4.1*, in which a set of residences receive electricity from the macro grid via the substation. There are 7 houses, each of which has a smart meter which gathers information about consumption (e.g. electricity usage, electricity demand, appliances' schedules etc.) as well as controlling the load for *shiftable appliances* in the house. The model assumes that every house has 1 EV for daily usage, and that each house is connected to the substation through triplex electricity lines.

4.2 GridLAB-D

4.2.1 Why GridLAB-D?

GridLAB-D is an open-source power distribution system simulator[**?**], together with an analysis tool, developed by the U.S. Department of Energy (DOE). It provides a flexible simulation environment, facilitating the integration of a number of pre-built modules, useful

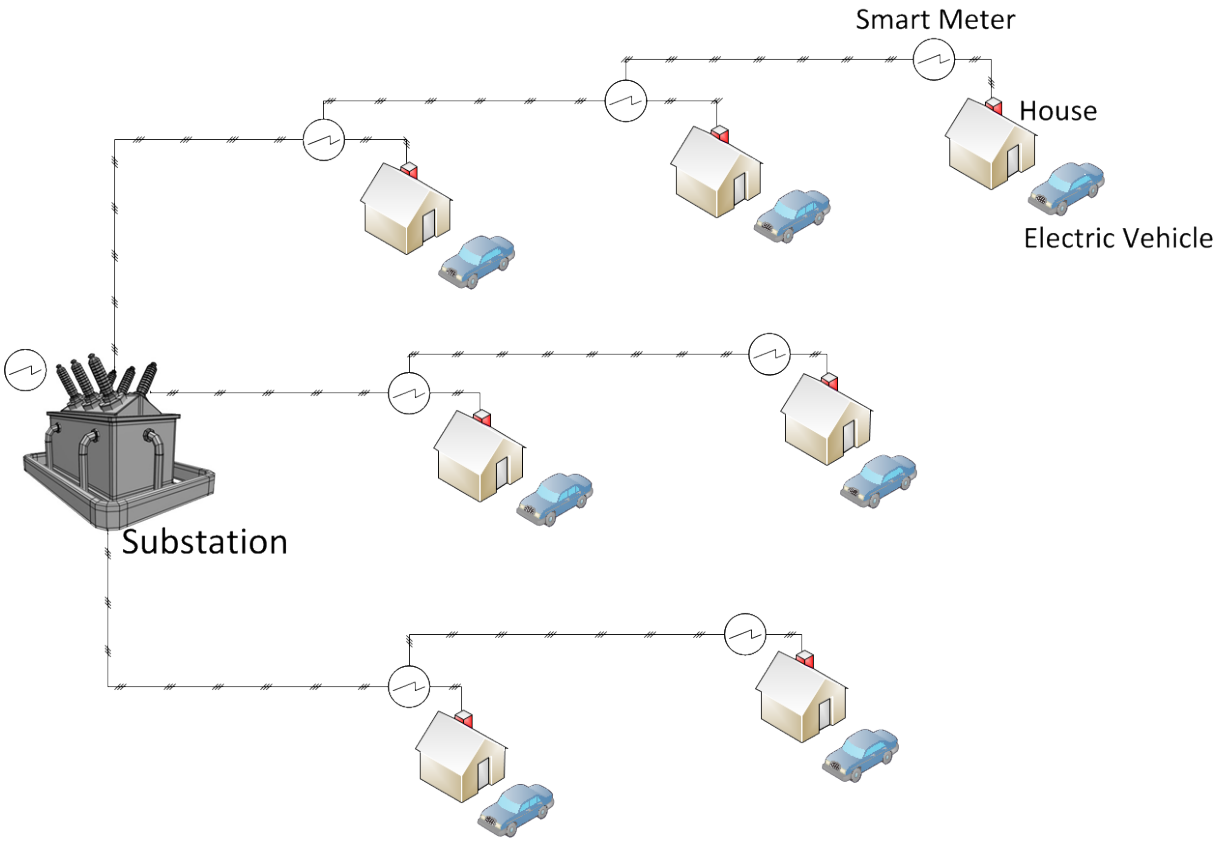


Figure 4.1: The Scenario to be Studied.

for this study. GridLAB-D has been widely used in DSM research [45], [46], [47], [48], [49].

The key benefits of using GridLAB-D as the environment simulator can be summarized as follows:

1. GridLAB-D provides the user with fine-grained control over different aspects of the model. In particular, the schedule, maximum charging power and vehicle efficiency can be defined.

2. Extensive data collection tools are made available to the user, for example, the *recorder*, which can be used to collect selected model information in a predefined time interval.

3. *Player objects* in GridLAB-D allow enable environment parameters to be set.

4.2.2 The model

The specific details of the GridLAB-D simulation model are attached in *appendix A*. Some of the key features of the model are:

1. Every EV follows a fixed schedule, leaving the residence at 08:00 and returning at 17:00, travelling a distance of 20 miles each day. Each EV has a 8.58034kWh battery with a mileage efficiency of 3.846 mile/kWh. It is assumed that the vehicle cannot recharge away from the residence.

2. Each EV charger has a charging efficiency of 0.9 and a maximum charging power of 1,700W.

3. A *recorder* logs the properties of the substation's transformer (e.g. the power, voltage etc.) every 30 minutes.

4. Each house has an associated recorder logging the details of electricity consumption every 30 minutes.

5. Each EV charger similarly has an associated recorder, logging the consumption details every 30 minutes.

4.3 Architectures

GridLAB-D can be extended in a number of different ways. Previous researchers have, largely, implemented algorithmic logic in GridLAB-D's source files. In this research, a client-server architecture has been introduced, facilitating greater flexibility in the project and resulting in a more easily extensible code base.

4.3.1 Technical architecture

The system's technical architecture is illustrated in *figure 4.2*. GridLAB-D runs the simulation using the model outlined above. The simulation uses HTTP connections to commu-

nicate with the web server in a continuous manner. The web server transfers requests from GridLAB-D to the UCT Decision Maker data structure.

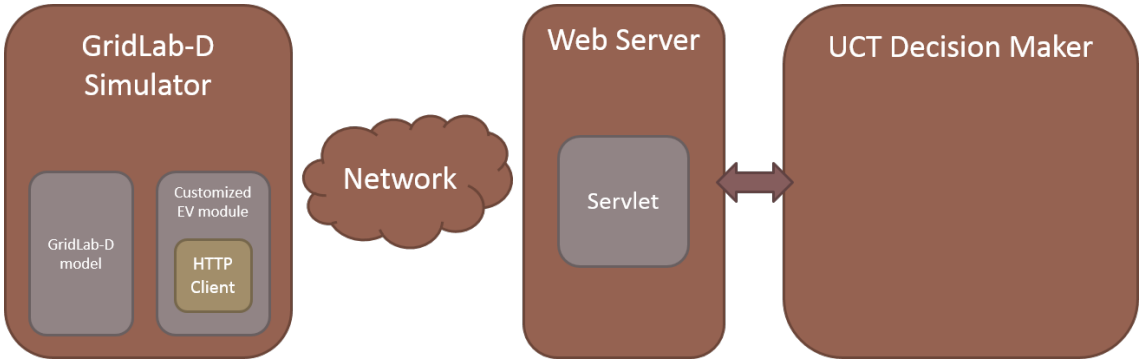


Figure 4.2: Technical architecture.

4.3.2 The simulator

The GridLAB-D simulator is written in C++. For the purposes of this research, the EV module class has been modified. Some of the most significant changes are outlined below:

1. An action table has been added, to be read by the EV charger, leading to the undertaking of the corresponding action.

Listing 4.1: EVs action table

```
//Evcharger_det.h
struct VEHICLEACTION {
        string name;
        int action;
        TIMESTAMP timestamp;
};
static VEHICLEACTION vehicle_actions[7];

//Evcharger_det.cpp
VEHICLEACTION evcharger_det::vehicle_actions[7]={
        {"EV1",0,-1},
        {"EV2",0,-1},
        {"EV3",0,-1},
        {"EV4",0,-1},
        {"EV5",0,-1},
        {"EV6",0,-1},
        {"EV7",0,-1}
};
```

2. Similarly to the basic model containing seven EVs discussed above, four other models were constructed, each containing a different number of vehicles.

3. Logic was added to the EV module's synchronization method to allow each EV to check its state every 30 minutes and update its charging power.

4. POCO HTTP Client has been embedded into the EV module's synchronization method thus each EV is able to send its state as a request to the web server and receive in response a decision as to when the EV requires synchronizing.

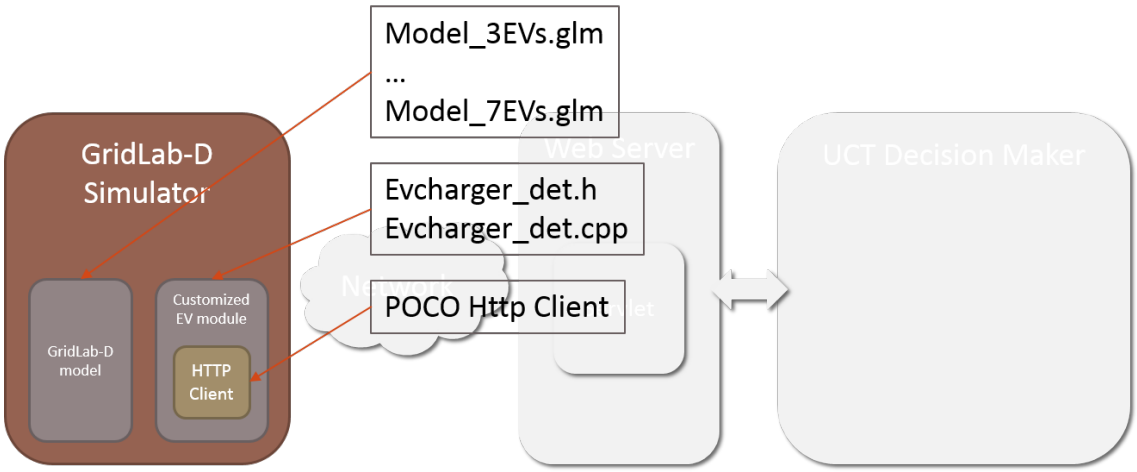


Figure 4.3: The simulator.

4.3.3 Web server

The web server is built using the *Grails* framework. The server handles each incoming request sent by the simulator, and the following actions are performed:

1. The server parses the message sent from an EV charger, using the parameters to initialize an EVCharger instance using the EV charger's ID.

2. It gathers the EV chargers' information. This information is sent to the UCT core for the creation of the root node (described in section 4.4 in detail).

3. The UCT core performs analysis and reaches a decision based on the EV chargers' states.

4. This decision is returned to the simulator.

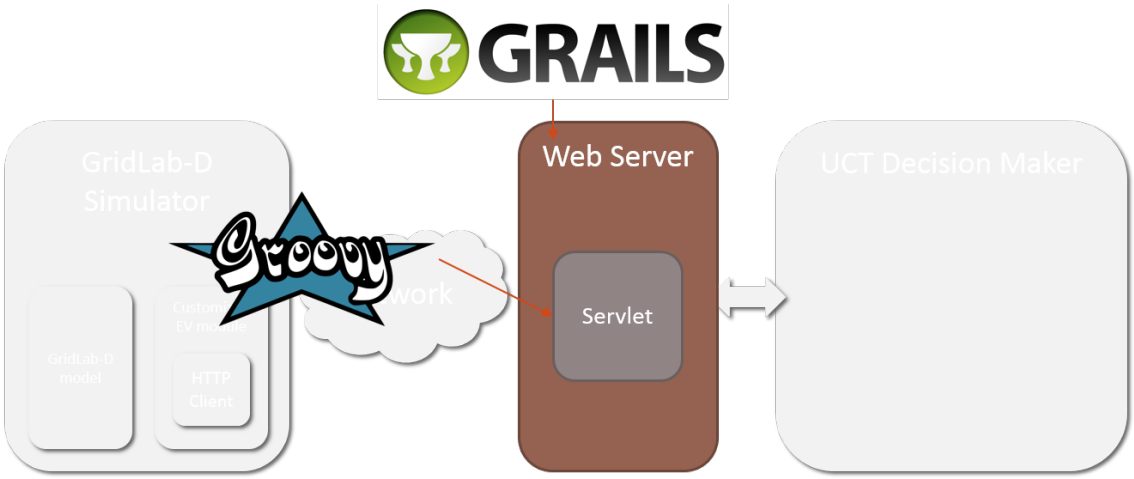


Figure 4.4: Web server.

4.4 UCT decision maker

The UCT decision maker is implemented in Java, for two primary reasons. The first is that building a search tree consumes a great deal of memory. Java's garbage collection eases the difficulty associated with manual collection of unused memory space. Secondly, Java supports a number of efficient data operation functionalities which is critical for this simulation.

4.4.1 Class diagram

The UCT decision maker class diagram is shown in appendix C. The class diagram includes the following parts:

GameState

GameState defines an interface requiring that all subclasses implement the equal() method.

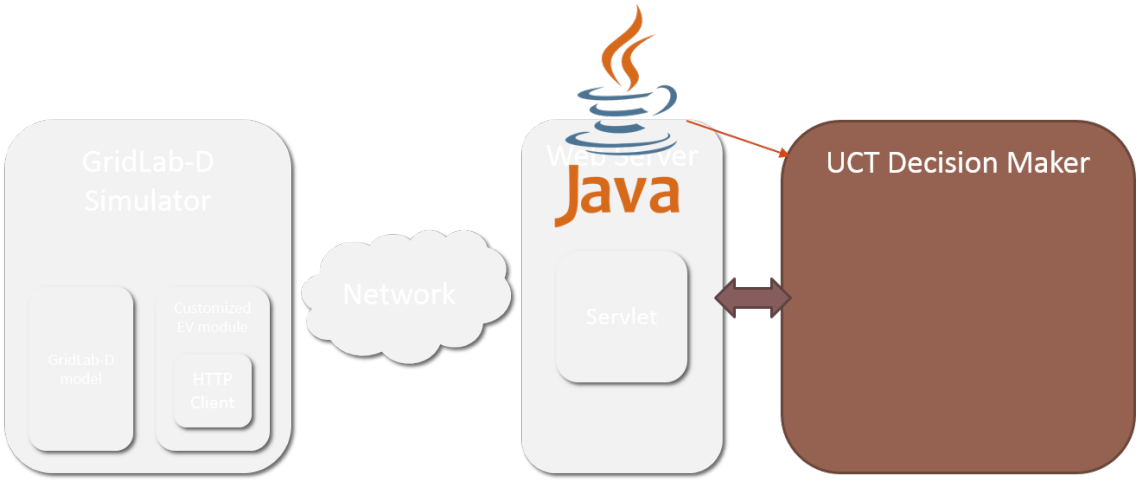


Figure 4.5: UCT decision maker.

EVChargerGridGameState

EVChargerGridGameState implements the GameState interface and stores a list of EVCharger objects, which are used to represent all EV chargers' state at a specific time and unique actionString. The string variable actionString represents the actions of all EV chargers (e.g. 0121342 means charge level "0" for EV1, "1" for EV2, "2" for EV3...).

EVCharger

EVCharger provides a copy of the EVCharger_det module of GridLAB-D to the UCT core, enabling it to simulate the charging procedure and store state information. *chargeType* (used previously in the actionString variable of EVChargerGridGameState) represents the power level of the EV charger ("0", "1", "2", and "3" represent 0/3, 1/3 ,2/3 and 3/3 respectively, of the maximum charging power).

Player

Player defines an interface requiring its subclasses to implement the MakeMove() method for move selection and the getCurState() method to get the current game status (i.e. the environment status of the residence).

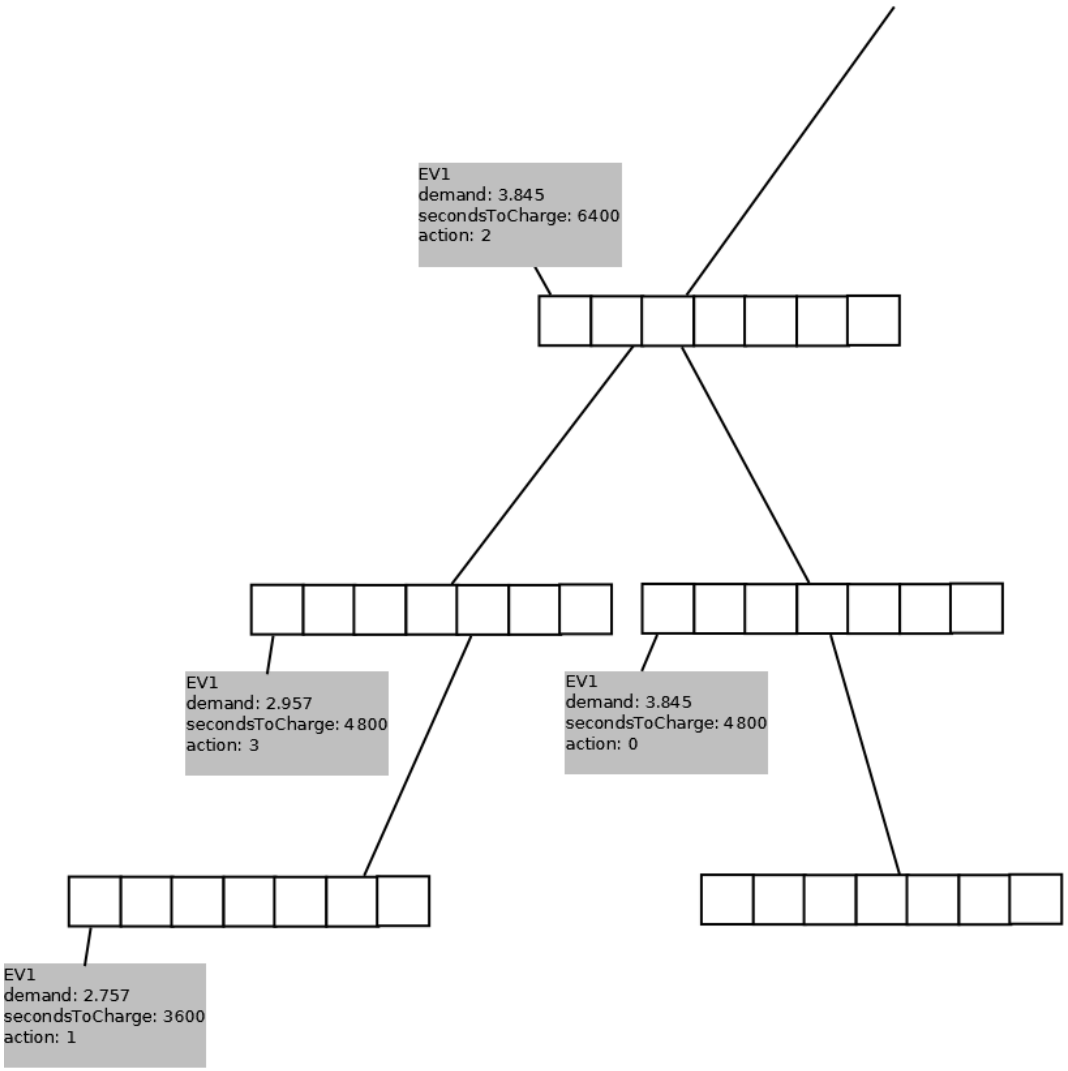


Figure 4.6: A snapshot of one part of UCT tree.

UCTPlayer

UCTPlayer is an abstract class which implements the main business logic of the UCT decision maker. The MakeMove() method supplies a high level call to enable decision making. *runTrial()* applies the four step iterations of UCT (discussed in chapter 3). Win(), Loss(), Level2(), Level3(), Level4() check the status of the state, the status can be only one of them. simulateFrom is a abstract method for the subclasses to overwrite.

Game

Interface Game describes all necessary methods for a problem modeled by game theory.

EVChargerGrid

EVChargerGrid implements the Game interface, modelling the EV charging coordination problem with the use of game theory.

UCTNode

Class UCTNode defines each node in UCT search tree, bestSelection() implements the selection behaviour of UCT, expand() implements the expansion step, getRandomChild() returns a random child node of the node to be expanded.

Chapter 5

Eperiments

5.1 Experiments

To examine the applicability of UCT to the EVs charging coordination problem, a series of experiments have been undertaken.

5.1.1 Experimental procedure

Simulation with UCT decision maker

The procedure for running a simulation with a specific number of EVs is as follows:

1) Set the NUMBER_OF_EVS value to 7 (i.e. the same as the number of EVs in the simulator's model in each experiment) in the UCT decision maker.

2) Launch the Grails server, which contains the UCT decision maker, on an HTTP port.

3) Run GridLAB-D with the relevant model file.

4) After finishing the simulation, gather and store the files generated by the recorder(the substation's record file, every EV charger' record file and every house meter's record file).

After concluding the experiment with 7 EVs, the steps outlined above are repeated with 6, 5, 4 and 3 EVs. All recorder files are collected for statistical analysis.

Simulation without UCT decision maker

1) Run GridLAB-D with the model for 7 EVs.

2) After finishing the simulation, collect the recorder generated files (substation's record file, every EV charger' record file and every house meter's record file).

As before, after concluding the experiment with 7 EVs, the steps outlined above are repeated with 6, 5, 4 and 3 EVs. All recorder files are collected for statistical analysis.

Simulate without EV

To collect the record data which is not contributed by EVs, run a special model without EVs, and collect the substation's record file.

5.1.2 Record Analysis

The simulations described above produce 11 sets of recorder files (5 from the simulation with UCT decision maker, 5 from simulation without UCT decision maker and 1 from the simulation with 0 EVs) ready for analysis.

The two key performance indicators: PAPR and PCSOC are then investigated. A method for providing an assessment of the impact of UCT on the performance of DSM on *PAPR*, as well as *PCSOC* maximization, is given as follows:

PAPR

1) For each type of record, extract the record's power column (referred to as *power_UCT*, *power_non-UCT*, *power_no-EV*).

2) Subtract *power_UCT* from *power_no-EV*, to obtain a *total_EVChargers_power_UCT* to *date-time* relation.

3) Subtract *power_non-UCT* from *power_no-EV*, to get a *total_EVChargers_power_non-UCT* to *date-time* relation.

4) Plot the two relations returned by step 2 and 3.

5) Extract the 7 EV chargers' recorder files from the simulations with UCT. For each file, separate 7 days' *power* to *date-time* relation. Combine the 7 EV chargers' *power* to *date-time* relation with each day's *power* to *date-time* to *EV_ID* relation.

6) Add a column to the relation returned by step 5, the *total_EVChargers_power_UCT* with associated *date-time*.

7) Repeat step 5 of simulation without UCT and step 6 with *total_EVChargers_power_non-UCT*.

8) Plot the relation returned by step 6 and 7.

9) Using the relation resulting from step 2, calculate, for each day, the charging period's PAPR with the UCT decision maker.

10) Using the relation returned by step 3, calculate, for each day, the charging period's PAPR without the UCT decision maker.

11) Calculate average daily PAPR using the PAPRs returned by step 9, and repeat with the PAPRs resulting from step 10.

12) Calculate the variance of both groups of PAPRs.

After completing these 12 steps, repeat with the 6 EVs, 5 EVs, 4 EVs and 3 EVs scenarios.

PCSOC

1) Using each EV's recorder file from the simulation with UCT, extract every day's 4 AM *battery_SOC* value.

2) Calculate the percentage of each level SOC value (e.g. 80% of PCSOCs achieve 100% value).

Chapter 6

Results & Evaluation

The purpose of this chapter is to present the experimental evidence produced by the simulations outlined in Chapter 5. An evaluation of the results is provided and discussed.

6.1 Experimental Results

6.1.1 Single week simulation

The following figures show the comparisons between *non-UCT charging* and *UCT charging* (charging with the coordination of UCT decision maker) in two scenarios. In the first scenario only the EV chargers are considered. The second scenario represents a broader perspective in which the total power load of the substation is taken into account. There are 5 groups of experimental results (3 EVs, 4 EVs, 5 EVs, 6 EVs and 7 EVs). All experimental results are available in digital format.

3EVs

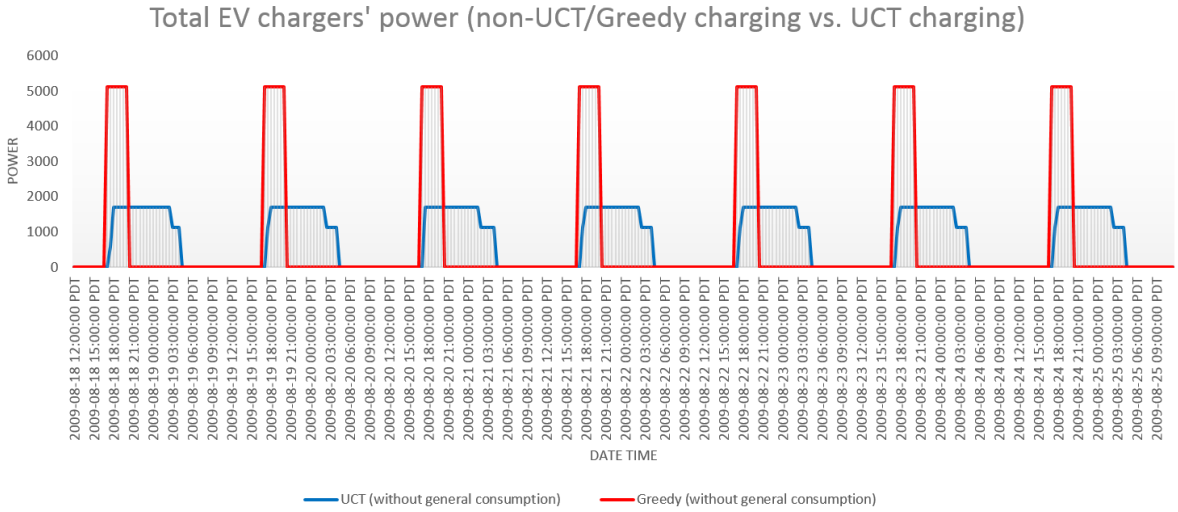


Figure 6.1: 3 EVs total power comparison.

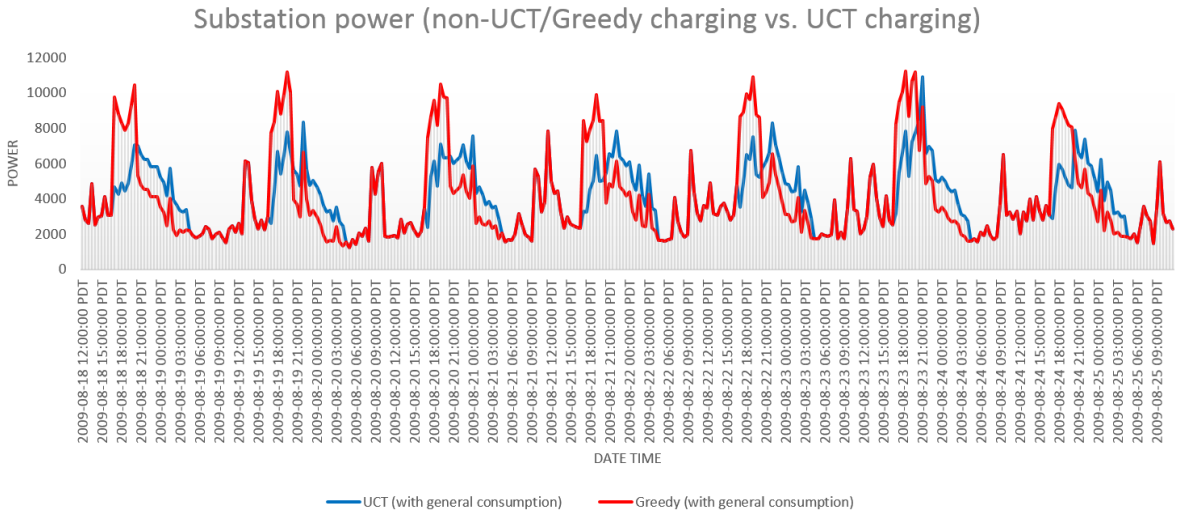


Figure 6.2: Substation power comparison.

Figure 6.1 shows 3 EV chargers' total power to date-time relationship over the 7 day simulation. Note the non-UCT approach charges EVs early at a very high power for a short period. In contrast, when the UCT decision maker is used, the behaviour favours a smoother power consumption over the whole charging period (from 17:00 to 4:30 the next day).

From *figure 6.2* we see that by coordinating EV chargers with the use of the UCT decision

maker, there is a reduction in the total load's peak points, as indicated by the substation's power load.

5EVs

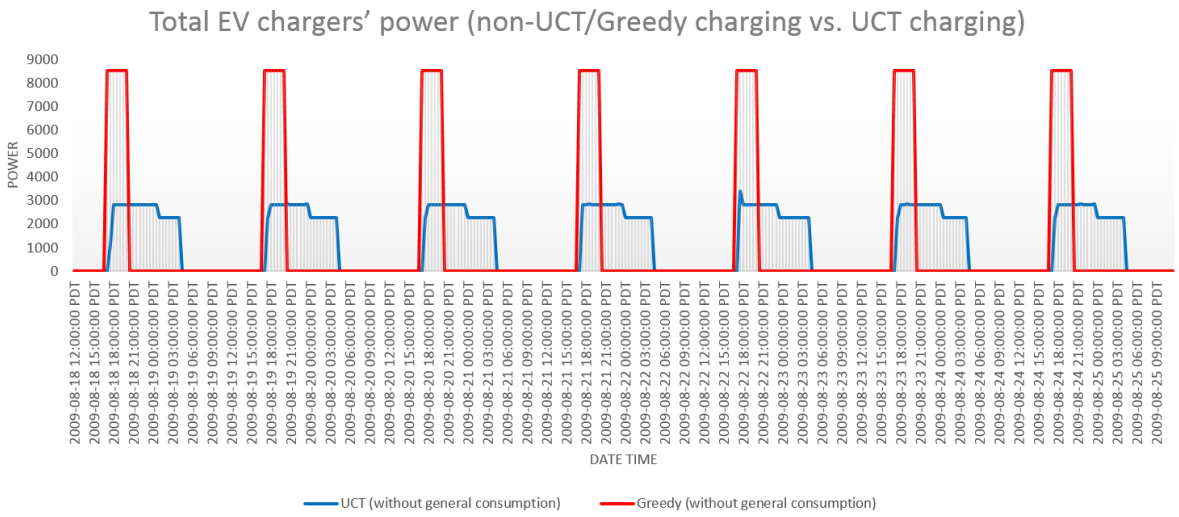


Figure 6.3: 5 EVs total power comparison.

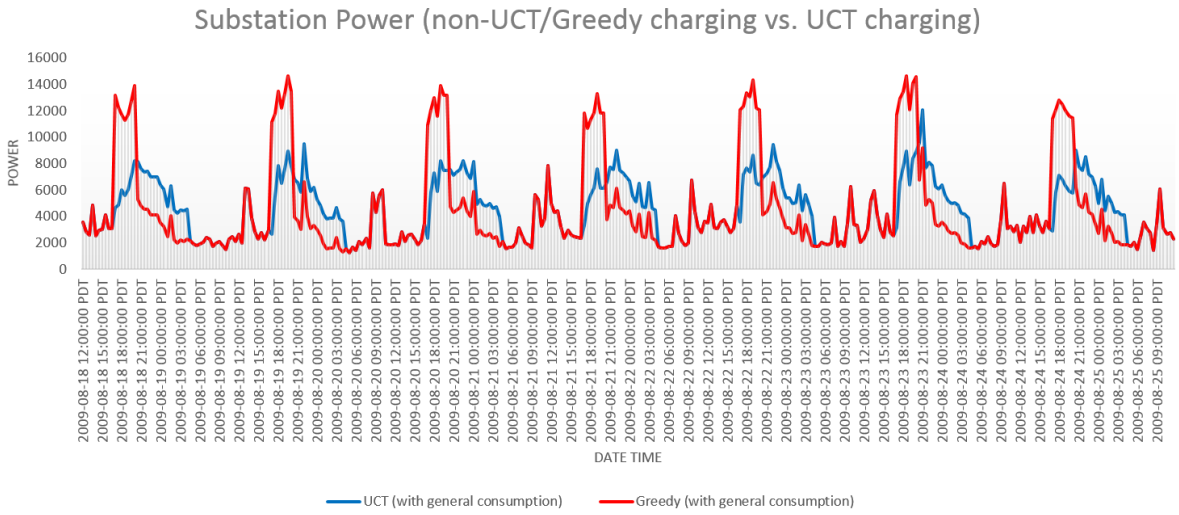


Figure 6.4: Substation power comparison.

Figure 6.3 illustrates 5 EV chargers' total power to date-time relationship over the 7 days' simulation. The behaviour is very similar to the 3 EVs simulation (*figure 6.1*). This indicates

that the UCT decision maker is very stable in terms of EV chargers' coordination. A careful

examination of the substation's power load reveals that the UCT decision maker has fewer peak power points in *figure 6.4* than in *figure 6.2*.

7EVs

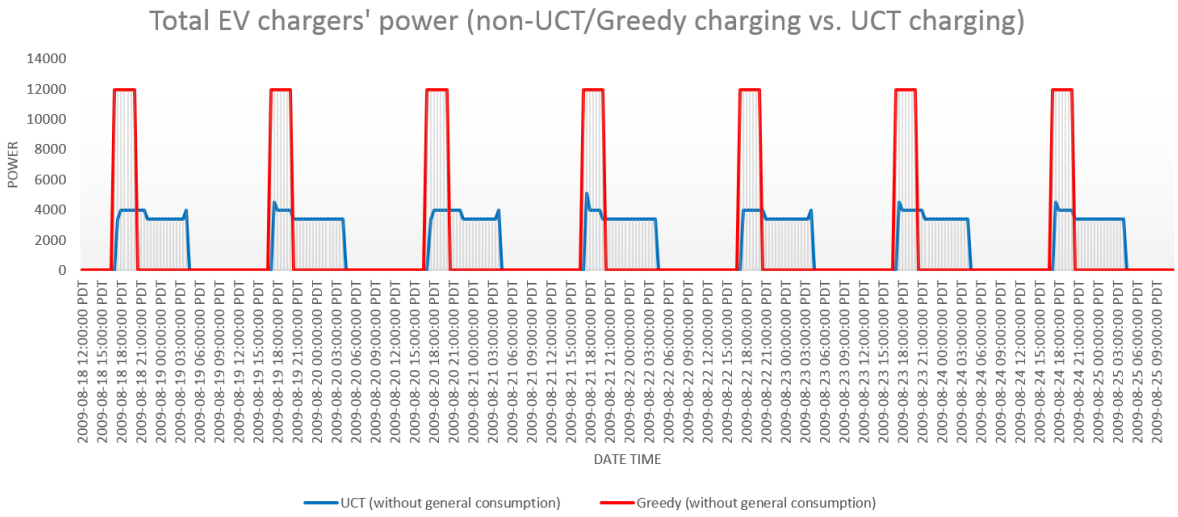


Figure 6.5: 7 EVs total power comparison.

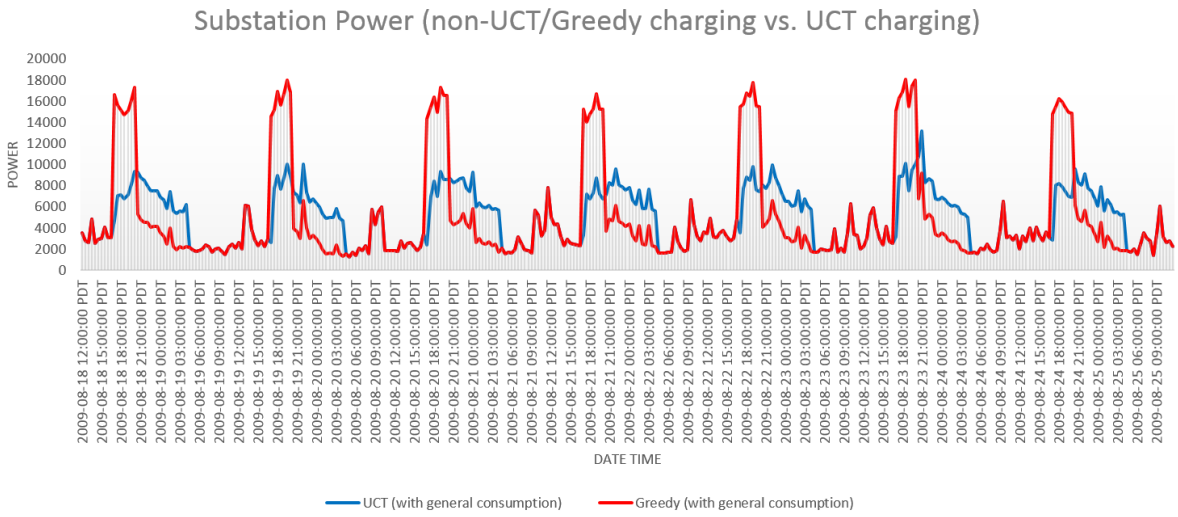


Figure 6.6: Substation power comparison.

Figure 6.5 demonstrates that the UCT decision maker performs well in the 7 EVs scenario. However, the real change can be found in *figure 6.6*, which sees a significant improvement in the balancing of the substation's power.

6.1.2 Daily battery-SOC to time relation

The daily *battery-SOC* to time relation illustrates how all EVs' battery-SOC changes with time over a single day. The curve shows the total EV charging infrastructure's power on the other axis.

The figures for these relations are given in *appendix B*. For the sake of clarity, only the records for day 1 are illustrated for non-UCT EV charging vs. UCT EV charging.

From *table 6.1, 6.2* and *6.3* we see that the charging process coordinated by UCT achieves very smooth curves in both the SOC rate and the total power values' variance.

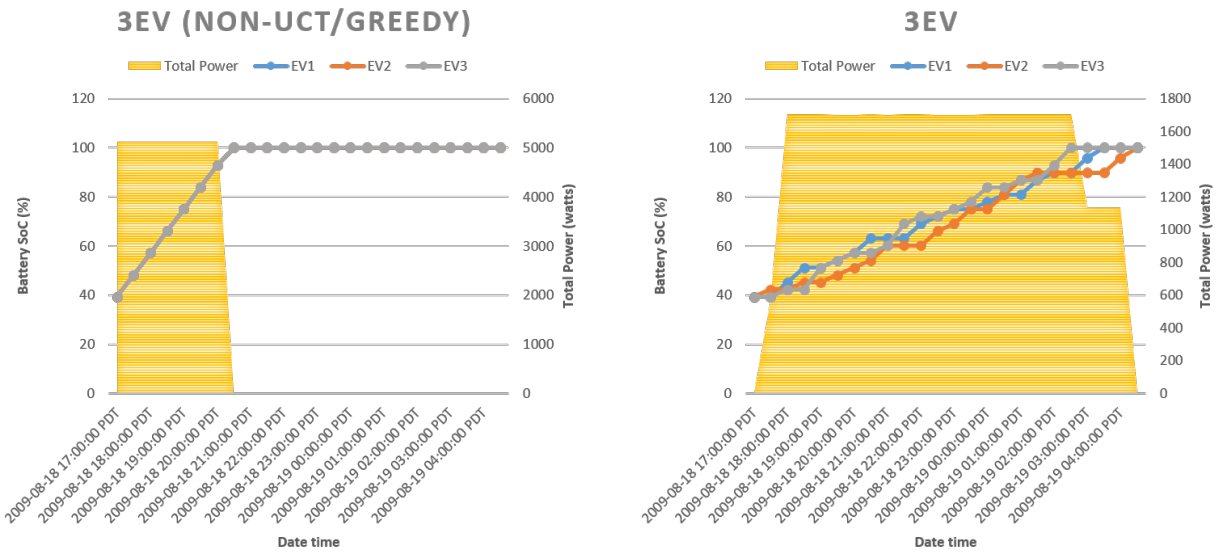
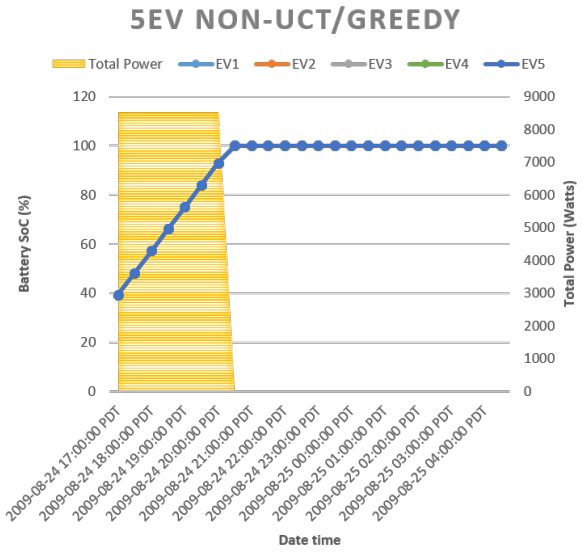


Table 6.1: 3EVs Day 1's SOC & total power to time relation (non-UCT vs. UCT each day).



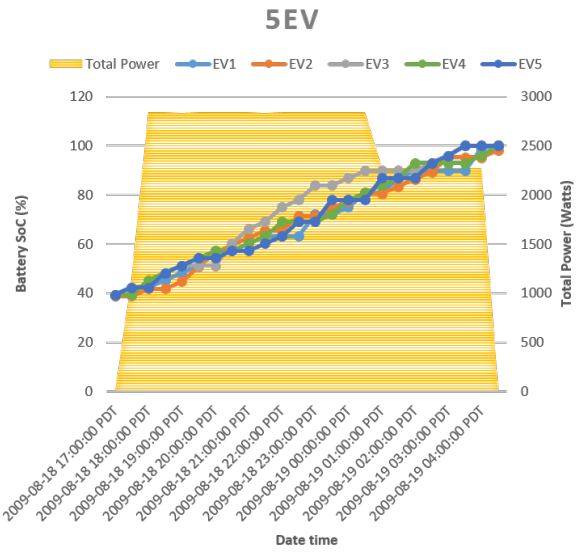
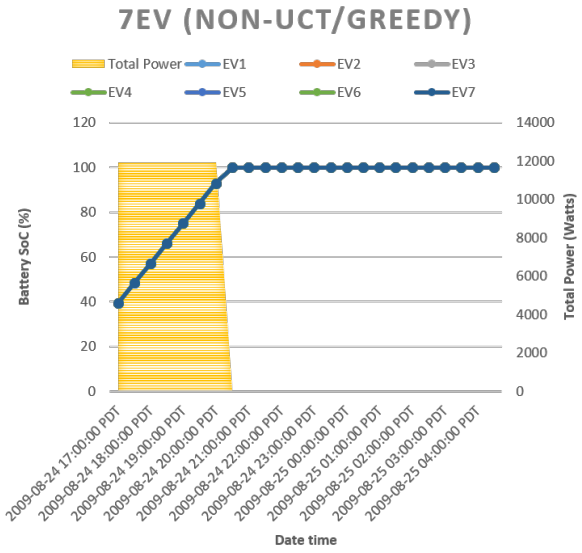


Table 6.2: 5EVs Day 1's SOC & total power to time relation (non-UCT vs. UCT each day).



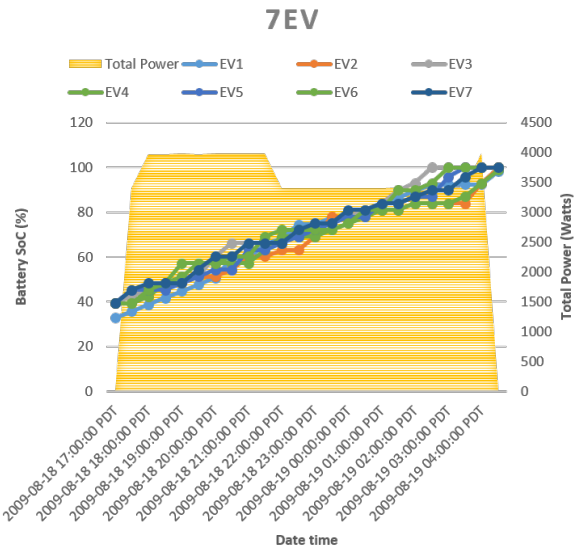


Table 6.3: 7EVs Day 1's SOC & total power to time relation (non-UCT vs. UCT each day).

6.2 Statistical Evidence and Evaluation

Although the figures provided earlier in this chapter give an indication of the positive impact of UCT, in this section we provide more concrete evidence with the use of statistical

analysis. The tables given below depict the statistics for each day's PAPR for both the UCT and non-UCT approaches in terms of the EV chargers only scenario, and overall scenario that takes account of the substation.

6.2.1 PAPR comparison

3EVs

	EV Chargers only			Residential overall		
	PAPR (UCT)	PAPR (Uncontrolled)	PAPR Decrement	PAPR (UCT)	PAPR (Uncontrolled)	PAPR Decrement
Day 1	1.1356	3.1438	63.88%	1.8084	3.1090	41.83%
Day 2	1.1487	3.1437	63.46%	2.6328	3.5095	24.98%
Day 3	1.1471	3.1438	63.51%	1.8772	2.9745	36.89%
Day 4	1.1489	3.1439	63.45%	2.1376	2.8887	26.00%
Day 5	1.1484	3.1440	63.47%	2.1720	3.0643	29.12%
Day 6	1.1499	3.1439	63.42%	3.2444	2.8102	-15.45%
Day 7	1.1485	3.1436	63.47%	2.2170	2.6163	15.26%
Average	**1.15E+00**	**3.14E+00**	**63.52%**	**2.2985**	**2.9961**	**23.28%**
Variance	**2.48E-05**	**2.14E-08**	**2.50E-06**	**2.45E-01**	**7.86E-02**	**3.56E-02**

Table 6.4: 3 EVs PAPR statistics.

Table 6.4 shows that when only considering EV chargers, UCT decision maker achieves a significant improvement on the PAPR decrement (e.g. in Day 1, 1.1356 UCT vs. 3.1438 non-UCT) at an average of 73.52%. In the overall environment, an average of 23.28% improvement is achieved by UCT. It is worth noting that the performance improvement offered by UCT on the overall environment is not very stable. A decrease in performance is detected in Day 5, and the variance (which indicates the stability) further supports this conclusion.

5EVs

	EV Chargers only			Residential overall		
	PAPR (UCT)	PAPR (Uncontrolled)	PAPR Decrement	PAPR (Uncontrolled)	PAPR Decrement	
Day 1	1.1831	3.1434	62.36%	1.7166	3.2972	47.94%
Day 2	1.1983	3.1440	61.89%	2.3864	3.5598	32.96%
Day 3	1.1973	3.1443	61.92%	1.5986	3.1959	49.98%
Day 4	1.1755	3.1436	62.61%	1.9837	3.1718	37.46%
Day 5	1.6835	3.1439	46.45%	2.0076	3.2652	38.52%
Day 6	1.1757	3.1438	62.60%	2.9074	3.0567	4.88%
Day 7	1.1984	3.1436	61.88%	2.0599	2.9271	29.63%
Average	**1.2588**	**3.1438**	**59.96%**	**2.0943**	**3.2105**	**34.77%**
Variance	**3.52E-02**	**9.19E-08**	**3.56E-03**	**1.93E-01**	**3.97E-02**	**2.25E-02**

Table 6.5: 5 EVs PAPR statistics.

Table 6.5 shows that for EV chargers' total power PAPR, a strong improvment of 59.96% was achieved by the UCT decision maker. Compared to 3 EVs overall PAPR decrement, it shows remarkable improvement from 23.23% to 34.77%. The reduction in variance (from 3.56E-02 to 2.52E-2) also indicates an increase in stability.

7EVs

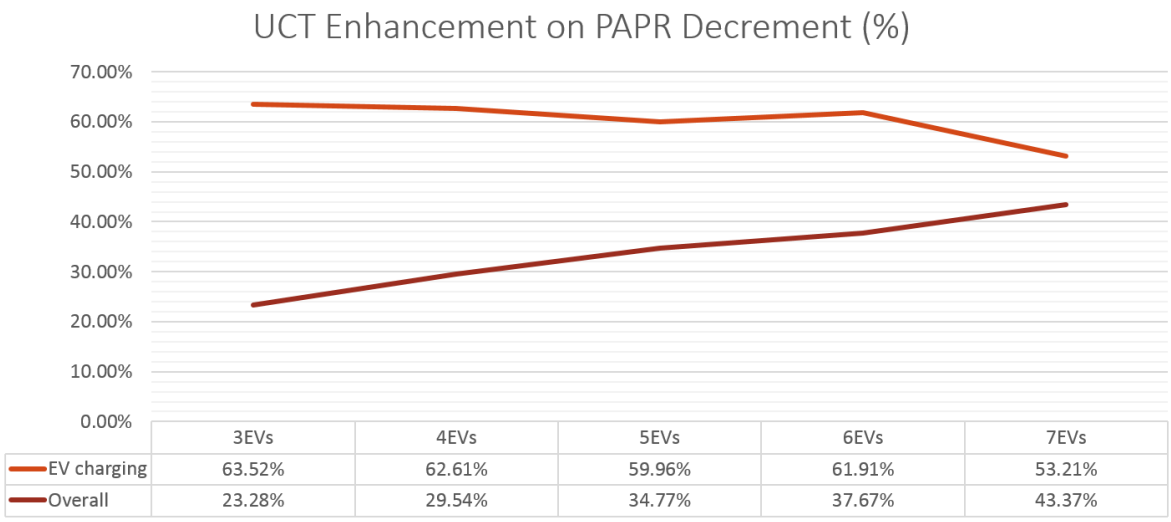
	EV Chargers only			Residential overall		
	PAPR (UCT)	PAPR (Uncontrolled)	PAPR Decrement	PAPR (UCT)	PAPR (Uncontrolled)	PAPR Decrement
Day 1	1.1877	3.1435	62.22%	1.6452	3.3583	51.01%
Day 2	1.5910	3.1440	49.39%	2.0137	3.5464	43.22%
Day 3	1.1705	3.1439	62.77%	1.5355	3.2818	53.21%
Day 4	2.0049	3.1436	36.22%	1.7044	3.2815	48.06%
Day 5	1.2053	3.1435	61.66%	1.7246	3.3392	48.35%
Day 6	1.5456	3.1438	50.84%	2.6736	3.1702	15.67%
Day 7	1.5908	3.1430	49.39%	1.7521	3.0674	42.88%
Average	**1.4708**	**3.1436**	**53.21%**	**1.8642**	**3.2921**	**43.37%**
Variance	**9.34E-02**	**1.07E-07**	**9.46E-03**	**1.49E-01**	**2.28E-02**	**1.62E-02**

Table 6.6: 7 EVs PAPR statistics.

Table 6.6 illustrates the outcome of the final EVs' experiment' PAPR statistics. As ex-

pected, the average PAPR decrement has been increased in comparison to the 5 EVs simulation, from 34.77% to 43.37%. It may be observed that the PAPR decrement diminishes when considering EV chargers only, as the number of EVs is increased.

PAPR change rate



	3EVs	4EVs	5EVs	6EVs	7EVs
EV charging	63.52%	62.61%	59.96%	61.91%	53.21%
Overall	23.28%	29.54%	34.77%	37.67%	43.37%

Figure 6.7: PAPR change rate.

By considering also the results of the 4 EV and 6 EV experiments' statistical analysis, the UCT performance enhancement with regards to the reduction of PAPR, against the number of EVs, can be seen in *figure 6.7*. Note that the PAPR decrement on EV chargers remains at approximately 60% but slowly decreases as the number of EVs increases. This may be caused the computational limitations. The overall PAPR decrement level increases in a stable manner when more EVs are added. Consequently, the greater the number of EVs in the grid, the greater the contribution of the UCT decision maker to the PAPR decrement.

6.2.2 PCSOC distribution

Table 6.7, Table 6.8 and Table 6.9 illustrate the PCSOC records for the EVs, each day in the 3 EV, 5 EV and 7 EV experiment scenarios. From these, we see that the UCT decision maker

has the biggest impact on the 3 EVs scenario. When increasing the number of EVs, the performance on maximizing the PCSOC value decreases. However, the results remain strong, even for the other cases.

	EV1	EV2	EV3
Day 1	100.00%	100.00%	100.00%
Day 2	100.00%	100.00%	100.00%
Day 3	100.00%	100.00%	99.17%
Day 4	100.00%	100.00%	100.00%
Day 5	100.00%	100.00%	100.00%
Day 6	100.00%	100.00%	100.00%
Day 7	100.00%	100.00%	100.00%
Average	100.00%	100.00%	99.88%

Table 6.7: 3EVs daily PCSOC record.

	EV1	EV2	EV3	EV4	EV5
Day 1	100.00%	98.25%	100.00%	100.00%	100.00%
Day 2	98.83%	100.00%	100.00%	100.00%	100.00%
Day 3	100.00%	100.00%	95.86%	100.00%	100.00%
Day 4	100.00%	95.86%	100.00%	100.00%	100.00%
Day 5	100.00%	100.00%	100.00%	100.00%	95.86%
Day 6	98.83%	100.00%	100.00%	100.00%	100.00%
Day 7	100.00%	100.00%	100.00%	100.00%	98.83%
Average	99.67%	99.16%	99.41%	100.00%	99.24%

Table 6.8: 5EVs daily PCSOC record.

	EV1	EV2	EV3	EV4	EV5	EV6	EV7
Day 1	98.25%	100.00%	100.00%	98.83%	100.00%	100.00%	100.00%
Day 2	94.11%	98.83%	100.00%	97.66%	100.00%	100.00%	100.00%
Day 3	98.88%	100.00%	100.00%	99.47%	100.00%	100.00%	100.00%
Day 4	98.83%	100.00%	100.00%	100.00%	98.83%	100.00%	98.83%
Day 5	100.00%	100.00%	100.00%	100.00%	91.72%	100.00%	100.00%
Day 6	98.83%	98.83%	100.00%	100.00%	100.00%	100.00%	98.83%
Day 7	100.00%	100.00%	92.89%	100.00%	98.83%	95.86%	100.00%
Average	98.42%	99.67%	98.98%	99.42%	98.48%	99.41%	99.67%

Table 6.9: 7EVs daily PCSOC record.

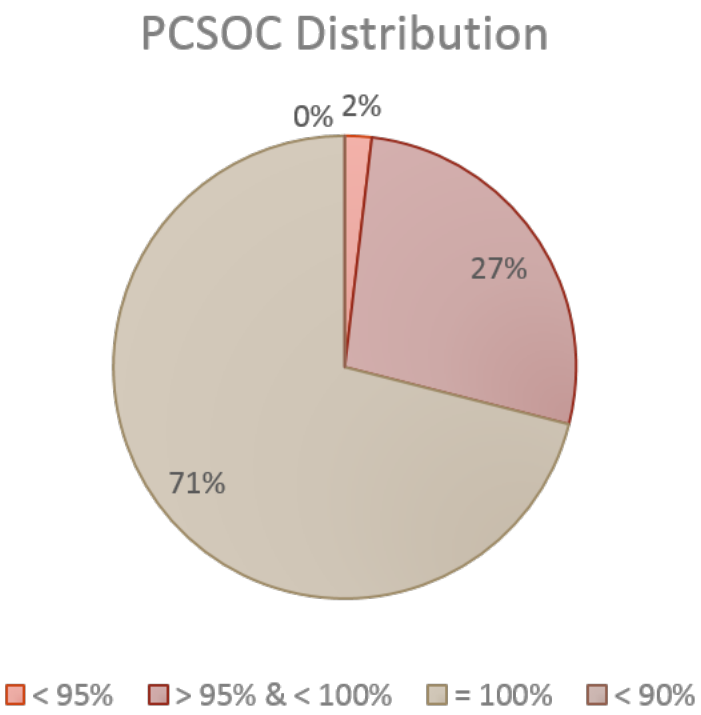


Figure 6.8: Distribution of PCSOC.

Chapter 7

Conclusions

The purpose of this chapter is to draw conclusions from the research conducted, assess how effectively the objectives set out in the research approach introduced in chapter 1 have been met and consider whether or not the research aim has been successfully addressed. Finally, the limitations of the research will be discussed, together with a suggestion of potential future work.

7.1 Research Assessment

7.1.1 Survey of literature

A literature review was conducted, giving this study a strong foundational and theoretical support in terms of the background of smart grids, demand-side management, AI algorithms and Monte Carlo methods.

7.1.2 UCT - Game Theory Problems

A general UCT decision maker was built to solve some simplistic game theory problems such as tic-tac-toe[1] and connect-four[2]. The UCT performed well against a random computer player in 5000 games achieving a degree of success[3]. This work provided the basis for the more sophisticated prototype implementation used to address the research question.

7.1.3 Implementation

GridLAB-D was used to simulate the research problem. The simple UCT decision maker developed for the simplistic problems was extended to function as part of this tool. The latter scenario exhibited far greater complexity than the simple problems, and consequently, one of the significant challenges for the development process was the optimization of the data structures, memory consumption, methods' efficiency etc...to meet the limitations of running the simulation on a single computer. However, this optimization resulted in higher quality, more extensible code.

7.1.4 Experiments

After overcoming some initial issues with memory consumption, the experiments were able to run properly, even in the computationally expensive case of the 7 EV scenario. A number of different EV scenarios were simulated, facilitating a comparison of the UCT approach and a non-UCT approach.

[1] a board game for two players, X and O, who take turns selecting spaces in a 3×3 grid. The player who succeeds in placing three respective marks in a horizontal, vertical, or diagonal row wins the game. (from Wikipedia)

[2] a two-player game in which the players first choose a color and then take turns dropping colored discs from the top into a seven-column, six-row vertically suspended grid. The pieces fall straight down, occupying the next available space within the column. The object of the game is to connect four of one's own discs of the same color next to each other vertically, horizontally, or diagonally before your opponent. (from Wikipedia)

[3] UCT achieved an average win/loss ratio of 5000:0 (as the first player) and 4698:302 (second player) in tic-tac-toe, 4780:220 (first player) and 4456: 504 (second player) in connect-four

7.1.5 Evaluation

The results of the experiments demonstrated that UCT can deliver significant improvements on PAPR as well as meeting consumer demand, as indicated by PCSOC. The impact was far greater than imagined when the project was proposed.

The evaluation also uncovered some of the limitations of UCT (discussed shortly).

7.2 Research Question

The results suggest that UCT is perfectly applicable to the EVs' charging coordination problem, particularly in cases in which we wish to reach the best decision for the EV charging environment in real-time.

7.2.1 Limitations

UCT Computational Performance

If the objective is to achieve the optimal decision, UCT is without doubt one of the most effective approaches available. However its computational requirement grows by $O(n^2)$ as the problem complexity grows by $O(n)$. Therefore, when the computational budget is a potential issue, this characteristic of the algorithm should be taken into consideration.

Implementation Fault Tolerance

While the test-bed that has been developed has many strengths in terms of scalability, extensibility and configurability, it is vulnerable to failure when an error occurs in the GridLAB-D, the UCT decision maker or the communication path between to the server, each representing a single point of failure. For the purposes of this research, this did not constitute a significant limitation, primarily due to the short period of time required to run the simulation. However, for more complex problems, simulations could take considerably longer and in such cases, this limitation must be taken into account.

7.3 Future Work

7.3.1 Modeling background usage in the simulation

Figure 6.2,figure 6.4 and figure 6.6 illustrate that the substation's load is still suboptimal, due to the background electricity usage (not attributed to EVs'). If this usage can be modeled in the simulation environment (both in GridLAB-D and the UCT decision maker), a much smoother curve in the substation's power to date time relation could be achieved.

7.3.2 Further optimization

To address the performance limitations mentioned above, further optimization can be conducted to enhance the performance of the system, rather than investment in additional computational infrastructure. A better heuristic function can be design to reduce the child nodes generated by the expansion step in every UCT's iteration.

7.3.3 Extending the usage

Since UCT performed well in this research, other extensions can be considered. For example other types of co-existing appliances could be investigated, or other DSM problems in which decision optimization is necessary.

Appendix A

GridLAB-D model

```
//Radial (3 branches, 7 households) simulation environement, poletop transformer used for each household
clock {
        timezone PST + 8PDT;
        starttime '2009-08-17 12:00:00';
        stoptime '2009-08-25 12:00:00';
}

 # define stylesheet = http : //gridlab-d.svn.sourceforge.net/viewvc/gridlab-d/trunk/core/gridlabd-2_0
         # set minimum_timestep = 60;
 # set profiler = 1;
 # set relax_naming_rules = 1;

module tape;

module residential {
        implicit_enduses NONE;
}
module powerflow {
        solver_method FBS;
        NR_iteration_limit 100;
}

//Player definition
class player {
        double value;
}

//Configurations

// Triplex Line
object triplex_line_configuration {
        name trip_line_config;
        conductor_1 object triplex_line_conductor {
                resistance 0.97;
                geometric_mean_radius 0.01111;
        };
        conductor_2 object triplex_line_conductor {
                resistance 0.97;
                geometric_mean_radius 0.01111;
        };
        conductor_N object triplex_line_conductor {
                resistance 0.97;
                geometric_mean_radius 0.01111;
        };
        insulation_thickness 0.08;
        diameter 0.368;
}

//Main transformer
object transformer_configuration {
        name transformer_configuration_TTD;
        connect_type WYE_WYE;
        install_type PADMOUNT;
        power_rating 15;
        primary_voltage 12470;
        secondary_voltage 2400;
        resistance 0.011;
        reactance 0.02;
}

//Poletop transformers
object transformer_configuration {
```

```
        name transformer_configuration_poletopA;
        connect_type SINGLE_PHASE_CENTER_TAPPED;
        install_type POLETOP;
        shunt_impedance 10000 + 10000j;
        primary_voltage 2400;
        secondary_voltage 120;
        powerA_rating 14.4; //240V 60 A
        impedance 0.00033 + 0.0022j;
}

object transformer_configuration {
        name transformer_configuration_poletopB;
        connect_type SINGLE_PHASE_CENTER_TAPPED;
        install_type POLETOP;
        shunt_impedance 10000 + 10000j;
        primary_voltage 2400;
        secondary_voltage 120;
        powerB_rating 14.4; //240V 60 A
        impedance 0.00033 + 0.0022j;
}

object transformer_configuration {
        name transformer_configuration_poletopC;
        connect_type SINGLE_PHASE_CENTER_TAPPED;
        install_type POLETOP;
        shunt_impedance 10000 + 10000j;
        primary_voltage 2400;
        secondary_voltage 120;
        powerC_rating 14.4; //240V 60 A
        impedance 0.00033 + 0.0022j;
}

//overhead line configuration

object line_configuration {
        name line_config_A;
        conductor_A overhead_line_conductor_100;
        conductor_B overhead_line_conductor_100;
        conductor_C overhead_line_conductor_100;
        conductor_N overhead_line_conductor_101;
        spacing line_spacing_200;
}

object line_spacing { //0 distance because single cable(residential area)
        name line_spacing_200;
        distance_AB 0;
        distance_BC 0;
        distance_AC 0;
        distance_AN 0;
        distance_BN 0;
        distance_CN 0;
}

object overhead_line_conductor {
        name overhead_line_conductor_100;
        geometric_mean_radius.00446;
        resistance 1.12;
}

object overhead_line_conductor {
        name overhead_line_conductor_101;
        geometric_mean_radius.00446;
        resistance 1.12;
}

//Structure


object node {
        name NT;
        phases ABCN;
        //phaseA_V +7199.558+0.000j;
        //phaseB_V -3599.779-6235.000j;
        //phaseC_V -3599.779+6235.000j;
        nominal_voltage 12470;
        bustype SWING;
}

object transformer {
        name TTD;
        phases ABCN;
        from NT;
        to ND;
        configuration transformer_configuration_TTD;
        groupid Transmission_Trans;
}

//Player for predicted load (read by meter)
object player {
        name TLoadPRED;
        file LoadSchedule.csv;
        loop 28;
}
```

```
object meter {
        name ND;
        phases ABCN;

        //proj_power TLoadPRED.value*1;

        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

//Overhead lines and house nodes (all on one branch distance 1000 fett from main transformer and then 100 feet
    between every household)

object overhead_line {
        name NDtoNH1;
        phases ABCN;
        from ND;
        to NH1;
        length 1000; //feet
        configuration line_config_A;
}

object node {
        name NH1;
        phases ABCN;
        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

object overhead_line {
        name NH1toNH2;
        phases ABCN;
        from NH1;
        to NH2;
        length 100; //feet
        configuration line_config_A;
}

object node {
        name NH2;
        phases ABCN;
        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

object overhead_line {
        name NH2toNH3;
        phases ABCN;
        from NH2;
        to NH3;
        length 100; //feet
        configuration line_config_A;
}

object node {
        name NH3;
        phases ABCN;
        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

object overhead_line {
        name NH3toNH4;
        phases ABCN;
        from NH3;
        to NH4;
        length 100; //feet
        configuration line_config_A;
}

object node {
        name NH4;
        phases ABCN;
        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

object overhead_line {
        name NH4toNH5;
        phases ABCN;
        from NH4;
        to NH5;
        length 100; //feet
        configuration line_config_A;
```

```
}

object node {
        name NH5;
        phases ABCN;
        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

object overhead_line {
        name NH5toNH6;
        phases ABCN;
        from NH5;
        to NH6;
        length 100; //feet
        configuration line_config_A;
}

object node {
        name NH6;
        phases ABCN;
        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

object overhead_line {
        name NH6toNH7;
        phases ABCN;
        from NH6;
        to NH7;
        length 100; //feet
        configuration line_config_A;
}

object node {
        name NH7;
        phases ABCN;
        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

object overhead_line {
        name NH7toNH8;
        phases ABCN;
        from NH7;
        to NH8;
        length 100; //feet
        configuration line_config_A;
}

object node {
        name NH8;
        phases ABCN;
        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

object overhead_line {
        name NH8toNH9;
        phases ABCN;
        from NH8;
        to NH9;
        length 100; //feet
        configuration line_config_A;
}

object node {
        name NH9;
        phases ABCN;
        //voltage_A 2401.7771;
        //voltage_B -1200.8886-2080.000j;
        //voltage_C -1200.8886+2080.000j;
        nominal_voltage 2400;
}

//poletop transformers (one for each household)

object transformer {
        name TH1;
        phases AS;
        from NH1;
        to TRNH1;
        configuration transformer_configuration_poletopA;
        groupid Distribution_Trans;
}
```

```
object transformer {
        name TH2;
        phases BS;
        from NH2;
        to TRNH2;
        configuration transformer_configuration_poletopB;
        groupid Distribution_Trans;
}

object transformer {
        name TH3;
        phases CS;
        from NH3;
        to TRNH3;
        configuration transformer_configuration_poletopC;
        groupid Distribution_Trans;
}

object transformer {
        name TH4;
        phases AS;
        from NH4;
        to TRNH4;
        configuration transformer_configuration_poletopA;
        groupid Distribution_Trans;
}

object transformer {
        name TH5;
        phases BS;
        from NH5;
        to TRNH5;
        configuration transformer_configuration_poletopB;
        groupid Distribution_Trans;
}

object transformer {
        name TH6;
        phases CS;
        from NH6;
        to TRNH6;
        configuration transformer_configuration_poletopC;
        groupid Distribution_Trans;
}

object transformer {
        name TH7;
        phases AS;
        from NH7;
        to TRNH7;
        configuration transformer_configuration_poletopA;
        groupid Distribution_Trans;
}

object transformer {
        name TH8;
        phases BS;
        from NH8;
        to TRNH8;
        configuration transformer_configuration_poletopB;
        groupid Distribution_Trans;
}

object transformer {
        name TH9;
        phases CS;
        from NH9;
        to TRNH9;
        configuration transformer_configuration_poletopC;
        groupid Distribution_Trans;
}

// Household triplex nodes

object triplex_node {
        name TRNH1;
        phases AS;
        nominal_voltage 120;
}

object triplex_node {
        name TRNH2;
        phases BS;
        nominal_voltage 120;
}

object triplex_node {
        name TRNH3;
        phases CS;
        nominal_voltage 120;
}

object triplex_node {
        name TRNH4;
```

```
        phases AS;
        nominal_voltage 120;
}

object triplex_node {
        name TRNH5;
        phases BS;
        nominal_voltage 120;
}

object triplex_node {
        name TRNH6;
        phases CS;
        nominal_voltage 120;
}

object triplex_node {
        name TRNH7;
        phases AS;
        nominal_voltage 120;
}

object triplex_node {
        name TRNH8;
        phases BS;
        nominal_voltage 120;
}

object triplex_node {
        name TRNH9;
        phases CS;
        nominal_voltage 120;
}

//Triplex lines from transformer to house (triplex node to triplex meter)
object triplex_line {
        name TRLH1;
        phases AS;
        from TRNH1;
        to TRMH1;
        length 100;
        configuration trip_line_config;
}

object triplex_line {
        name TRLH2;
        phases BS;
        from TRNH2;
        to TRMH2;
        length 100;
        configuration trip_line_config;
}

object triplex_line {
        name TRLH3;
        phases CS;
        from TRNH3;
        to TRMH3;
        length 100;
        configuration trip_line_config;
}

object triplex_line {
        name TRLH4;
        phases AS;
        from TRNH4;
        to TRMH4;
        length 100;
        configuration trip_line_config;
}

object triplex_line {
        name TRLH5;
        phases BS;
        from TRNH5;
        to TRMH5;
        length 100;
        configuration trip_line_config;
}

object triplex_line {
        name TRLH6;
        phases CS;
        from TRNH6;
        to TRMH6;
        length 100;
        configuration trip_line_config;
}

object triplex_line {
        name TRLH7;
        phases AS;
        from TRNH7;
        to TRMH7;
```

```
        length 100;
        configuration trip_line_config;
}

object triplex_line {
        name TRLH8;
        phases BS;
        from TRNH8;
        to TRMH8;
        length 100;
        configuration trip_line_config;
}

object triplex_line {
        name TRLH9;
        phases CS;
        from TRNH9;
        to TRMH9;
        length 100;
        configuration trip_line_config;
}

//Triple meters (inside every household)s
object triplex_meter {
        name TRMH1;
        phases AS;
        nominal_voltage 120;
}

object triplex_meter {
        name TRMH2;
        phases BS;
        nominal_voltage 120;
}

object triplex_meter {
        name TRMH3;
        phases CS;
        nominal_voltage 120;
}

object triplex_meter {
        name TRMH4;
        phases AS;
        nominal_voltage 120;
}

object triplex_meter {
        name TRMH5;
        phases BS;
        nominal_voltage 120;
}

object triplex_meter {
        name TRMH6;
        phases CS;
        nominal_voltage 120;
}

object triplex_meter {
        name TRMH7;
        phases AS;
        nominal_voltage 120;
}

object triplex_meter {
        name TRMH8;
        phases BS;
        nominal_voltage 120;
}

object triplex_meter {
        name TRMH9;
        phases CS;
        nominal_voltage 120;
}

//Players for loads
object player {
        name p1388;
        file Wr1388.csv;
        //loop 0;
}

object player {
        name p1425;
        file Wr1425.csv;
        //loop 0;
}

object player {
        name p1428;
        file Wr1428.csv;
        //loop 0;
```

```
}

object player {
        name p1435;
        file Wr1435.csv;
        //loop 0;
}

object player {
        name p1440;
        file Wr1440.csv;
        //loop 0;
}

//Houses (with loads)
object house {
        parent TRMH1;
        name H1;

        //Turn off everything
        heating_system_type NONE;
        cooling_system_type NONE;
        fan_type NONE;
        auxiliary_system_type NONE;
        auxiliary_strategy NONE;

        // Plug load (convenience loads)
        object ZIPload {
                groupid householdLoad;
                base_power p1388.value;
                power_fraction 1.00000;
                impedance_fraction 0.000000;
                current_fraction 0.000000;
                power_pf 1.00000;
                current_pf 0.950000;
                impedance_pf 0.950000;
        };

        // EV Car
        object evcharger_det {
                name EV1;

                arrival_at_work 500;
                duration_at_work 11.5 h;
                arrival_at_home 1700;
                duration_at_home 11.5 h;
                mileage_efficiency 3.846;
                maximum_charge_rate 1700;
                charging_efficiency 0.90;
                battery_SOC 100;

                work_charging_available FALSE;
                travel_distance 20;
        };

}

object house {
        parent TRMH2;
        name H2;

        //Turn off everything
        heating_system_type NONE;
        cooling_system_type NONE;
        fan_type NONE;
        auxiliary_system_type NONE;
        auxiliary_strategy NONE;

        // Plug load (convenience loads)
        object ZIPload {
                groupid householdLoad;
                base_power p1425.value;
                power_fraction 1.00000;
                impedance_fraction 0.000000;
                current_fraction 0.000000;
                power_pf 1.00000;
                current_pf 0.950000;
                impedance_pf 0.950000;
        };

        // EV Car
        object evcharger_det {
                name EV2;
                arrival_at_work 500;
                duration_at_work 11.5 h;
                arrival_at_home 1700;
                duration_at_home 11.5 h;
                mileage_efficiency 3.846;
                maximum_charge_rate 1700;
                charging_efficiency 0.90;
                battery_SOC 100;

                work_charging_available FALSE;
                travel_distance 20;
```

```
        };

}

object house {
        parent TRMH3;
        name H3;

        //Turn off everything
        heating_system_type NONE;
        cooling_system_type NONE;
        fan_type NONE;
        auxiliary_system_type NONE;
        auxiliary_strategy NONE;

        // Plug load (convenience loads)
        object ZIPload {
                groupid householdLoad;
                base_power p1428.value;
                power_fraction 1.00000;
                impedance_fraction 0.000000;
                current_fraction 0.000000;
                power_pf 1.00000;
                current_pf 0.950000;
                impedance_pf 0.950000;
        };

        // EV Car
        object evcharger_det {
                name EV3;
                arrival_at_work 500;
                duration_at_work 11.5 h;
                arrival_at_home 1700;
                duration_at_home 11.5 h;
                mileage_efficiency 3.846;
                maximum_charge_rate 1700;
                charging_efficiency 0.90;
                battery_SOC 100;

                work_charging_available FALSE;
                travel_distance 20;
        };

}

object house {
        parent TRMH4;
        name H4;

        //Turn off everything
        heating_system_type NONE;
        cooling_system_type NONE;
        fan_type NONE;
        auxiliary_system_type NONE;
        auxiliary_strategy NONE;

        // Plug load (convenience loads)
        object ZIPload {
                groupid householdLoad;
                base_power p1428.value;
                power_fraction 1.00000;
                impedance_fraction 0.000000;
                current_fraction 0.000000;
                power_pf 1.00000;
                current_pf 0.950000;
                impedance_pf 0.950000;
        };

        // EV Car
        object evcharger_det {
                name EV4;
                arrival_at_work 500;
                duration_at_work 11.5 h;
                arrival_at_home 1700;
                duration_at_home 11.5 h;
                mileage_efficiency 3.846;
                maximum_charge_rate 1700;
                charging_efficiency 0.90;
                battery_SOC 100;

                work_charging_available FALSE;
                travel_distance 20;
        };

}

object house {
        parent TRMH5;
        name H5;

        //Turn off everything
        heating_system_type NONE;
        cooling_system_type NONE;
        fan_type NONE;
```

```
        auxiliary_system_type NONE;
        auxiliary_strategy NONE;

        // Plug load (convenience loads)
        object ZIPload {
                groupid householdLoad;
                base_power p1435.value;
                power_fraction 1.00000;
                impedance_fraction 0.000000;
                current_fraction 0.000000;
                power_pf 1.00000;
                current_pf 0.950000;
                impedance_pf 0.950000;
        };

        // EV Car
        object evcharger_det {
                name EV5;
                arrival_at_work 500;
                duration_at_work 11.5 h;
                arrival_at_home 1700;
                duration_at_home 11.5 h;
                mileage_efficiency 3.846;
                maximum_charge_rate 1700;
                charging_efficiency 0.90;
                battery_SOC 100;

                work_charging_available FALSE;
                travel_distance 20;
        };

}

object house {
        parent TRMH6;
        name H6;

        //Turn off everything
        heating_system_type NONE;
        cooling_system_type NONE;
        fan_type NONE;
        auxiliary_system_type NONE;
        auxiliary_strategy NONE;

        // Plug load (convenience loads)
        object ZIPload {
                groupid householdLoad;
                base_power p1440.value;
                power_fraction 1.00000;
                impedance_fraction 0.000000;
                current_fraction 0.000000;
                power_pf 1.00000;
                current_pf 0.950000;
                impedance_pf 0.950000;
        };

        // EV Car
        object evcharger_det {
                name EV6;
                arrival_at_work 500;
                duration_at_work 11.5 h;
                arrival_at_home 1700;
                duration_at_home 11.5 h;
                mileage_efficiency 3.846;
                maximum_charge_rate 1700;
                charging_efficiency 0.90;
                battery_SOC 100;

                work_charging_available FALSE;
                travel_distance 20;
        };

}

object house {
        parent TRMH7;
        name H7;

        //Turn off everything
        heating_system_type NONE;
        cooling_system_type NONE;
        fan_type NONE;
        auxiliary_system_type NONE;
        auxiliary_strategy NONE;

        // Plug load (convenience loads)
        object ZIPload {
                groupid householdLoad;
                base_power p1388.value;
                power_fraction 1.00000;
                impedance_fraction 0.000000;
                current_fraction 0.000000;
                power_pf 1.00000;
                current_pf 0.950000;
```

```
                impedance_pf 0.950000;
        };

        // EV Car
        object evcharger_det {
                name EV7;
                arrival_at_work 500;
                duration_at_work 11.5 h;
                arrival_at_home 1700;
                duration_at_home 11.5 h;
                mileage_efficiency 3.846;
                maximum_charge_rate 1700;
                charging_efficiency 0.90;
                battery_SOC 100;

                work_charging_available FALSE;
                travel_distance 20;
        };

}

object house {
        parent TRMH8;
        name H8;

        //Turn off everything
        heating_system_type NONE;
        cooling_system_type NONE;
        fan_type NONE;
        auxiliary_system_type NONE;
        auxiliary_strategy NONE;

        // Plug load (convenience loads)
        object ZIPload {
                groupid householdLoad;
                base_power p1440.value;
                power_fraction 1.00000;
                impedance_fraction 0.000000;
                current_fraction 0.000000;
                power_pf 1.00000;
                current_pf 0.950000;
                impedance_pf 0.950000;
        };

}

object house {
        parent TRMH9;
        name H9;

        //Turn off everything
        heating_system_type NONE;
        cooling_system_type NONE;
        fan_type NONE;
        auxiliary_system_type NONE;
        auxiliary_strategy NONE;

        // Plug load (convenience loads)
        object ZIPload {
                groupid householdLoad;
                base_power p1425.value;
                power_fraction 1.00000;
                impedance_fraction 0.000000;
                current_fraction 0.000000;
                power_pf 1.00000;
                current_pf 0.950000;
                impedance_pf 0.950000;
        };

}

//Recorders
//Transformer
object recorder {
        parent ND;
        file V_Data_transformer.csv;
        interval 1800;
        limit 1000000;
        property measured_real_power,
        measured_power.real,
        measured_power.imag,
        voltage_A.real,
        voltage_A.imag,
        voltage_B.real,
        voltage_B.imag,
        voltage_C.real,
        voltage_C.imag;
};

//Main transformer temperature
//object recorder {
//     parent TTD;
//     file T_transformer.csv;
//     interval 1800;
```

```
//     limit 1000000;
//     property top_oil_hot_spot_temperature, winding_hot_spot_temperature;
//};


//branch1
object recorder {
        parent TRMH1;
        file V_tm_1.csv;
        interval 1800;
        limit 1000000;
        property measured_real_power,
        measured_power.real,
        measured_power.imag,
        voltage_1.real,
        voltage_1.imag,
        voltage_2.real,
        voltage_2.imag,
        voltage_12.real,
        voltage_12.imag;
};

object recorder {
        parent EV1;
        file EV1.csv;
        interval 1800;
        limit 1000000;
        property "charge_rate",
        "battery_SOC",
        "vehicle_location",
        "battery_capacity";
};

object recorder {
        parent TRMH2;
        file V_tm_2.csv;
        interval 1800;
        limit 1000000;
        property measured_real_power,
        measured_power.real,
        measured_power.imag,
        voltage_1.real,
        voltage_1.imag,
        voltage_2.real,
        voltage_2.imag,
        voltage_12.real,
        voltage_12.imag;
};

object recorder {
        parent EV2;
        file EV2.csv;
        interval 1800;
        limit 1000000;
        property "charge_rate",
        "battery_SOC",
        "vehicle_location",
        "battery_capacity";
};

object recorder {
        parent TRMH3;
        file V_tm_3.csv;
        interval 1800;
        limit 1000000;
        property measured_real_power,
        measured_power.real,
        measured_power.imag,
        voltage_1.real,
        voltage_1.imag,
        voltage_2.real,
        voltage_2.imag,
        voltage_12.real,
        voltage_12.imag;
};

object recorder {
        parent EV3;
        file EV3.csv;
        interval 1800;
        limit 1000000;
        property "charge_rate",
        "battery_SOC",
        "vehicle_location",
        "battery_capacity";
};

object recorder {
        parent TRMH4;
        file V_tm_4.csv;
        interval 1800;
        limit 1000000;
        property measured_real_power,
        measured_power.real,
```

```
        measured_power.imag,
        voltage_1.real,
        voltage_1.imag,
        voltage_2.real,
        voltage_2.imag,
        voltage_12.real,
        voltage_12.imag;
};

object recorder {
        parent EV4;
        file EV4.csv;
        interval 1800;
        limit 1000000;
        property "charge_rate",
        "battery_SOC",
        "vehicle_location",
        "battery_capacity";
};

object recorder {
        parent TRMH5;
        file V_tm_5.csv;
        interval 1800;
        limit 1000000;
        property measured_real_power,
        measured_power.real,
        measured_power.imag,
        voltage_1.real,
        voltage_1.imag,
        voltage_2.real,
        voltage_2.imag,
        voltage_12.real,
        voltage_12.imag;
};

object recorder {
        parent EV5;
        file EV5.csv;
        interval 1800;
        limit 1000000;
        property "charge_rate",
        "battery_SOC",
        "vehicle_location",
        "battery_capacity";
};

object recorder {
        parent TRMH6;
        file V_tm_6.csv;
        interval 1800;
        limit 1000000;
        property measured_real_power,
        measured_power.real,
        measured_power.imag,
        voltage_1.real,
        voltage_1.imag,
        voltage_2.real,
        voltage_2.imag,
        voltage_12.real,
        voltage_12.imag;
};

object recorder {
        parent EV6;
        file EV6.csv;
        interval 1800;
        limit 1000000;
        property "charge_rate",
        "battery_SOC",
        "vehicle_location",
        "battery_capacity";
};

object recorder {
        parent TRMH7;
        file V_tm_7.csv;
        interval 1800;
        limit 1000000;
        property measured_real_power,
        measured_power.real,
        measured_power.imag,
        voltage_1.real,
        voltage_1.imag,
        voltage_2.real,
        voltage_2.imag,
        voltage_12.real,
        voltage_12.imag;
};

object recorder {
        parent EV7;
        file EV7.csv;
        interval 1800;
```

```
        limit 1000000;
        property "charge_rate",
        "battery_SOC",
        "vehicle_location",
        "battery_capacity";
};
```

Appendix B

Daily battery-SOC to time relationships

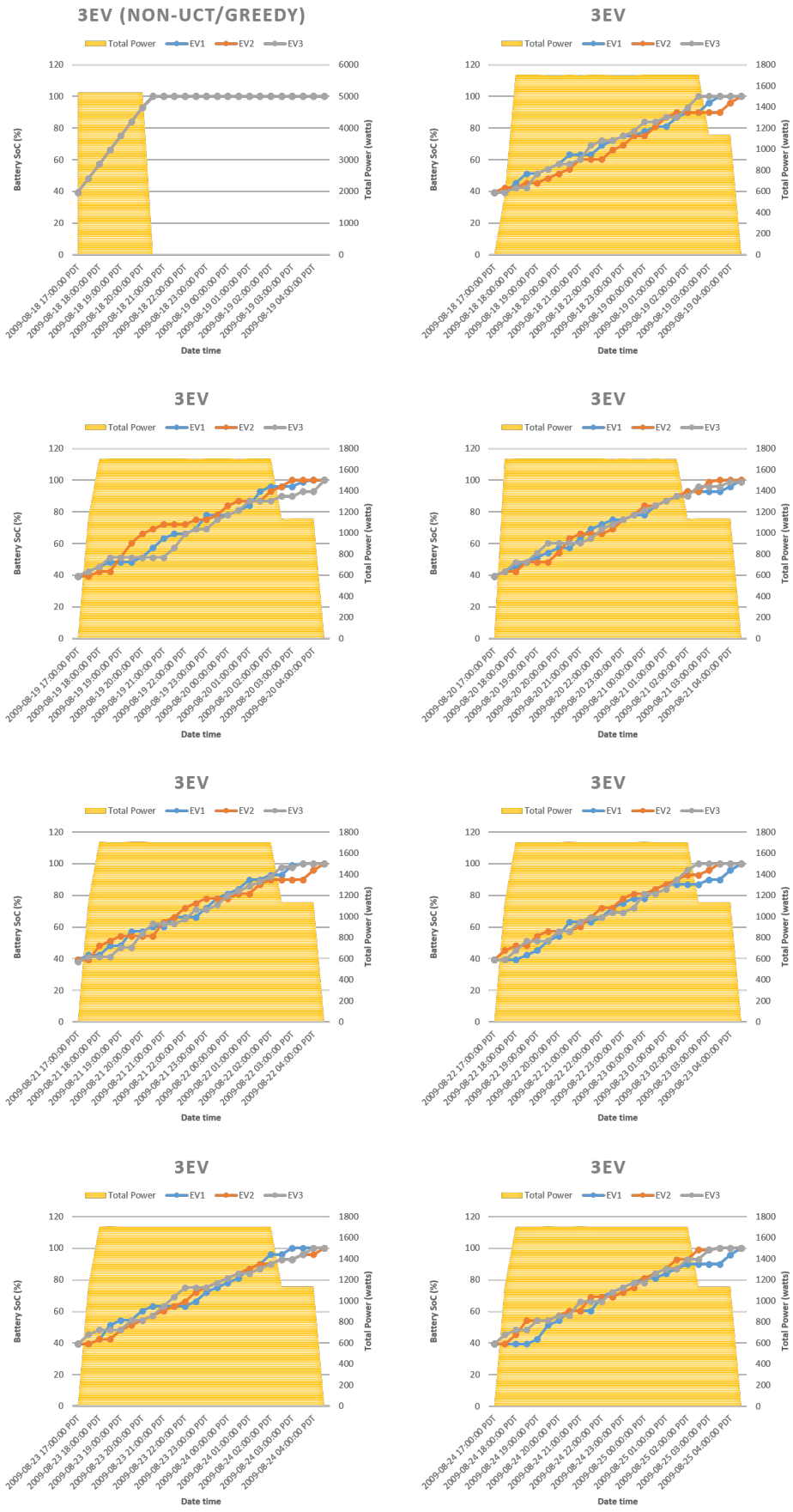


Table B.1: 3EVs daily SOC & total power to time relation (non-UCT vs. UCT each day).

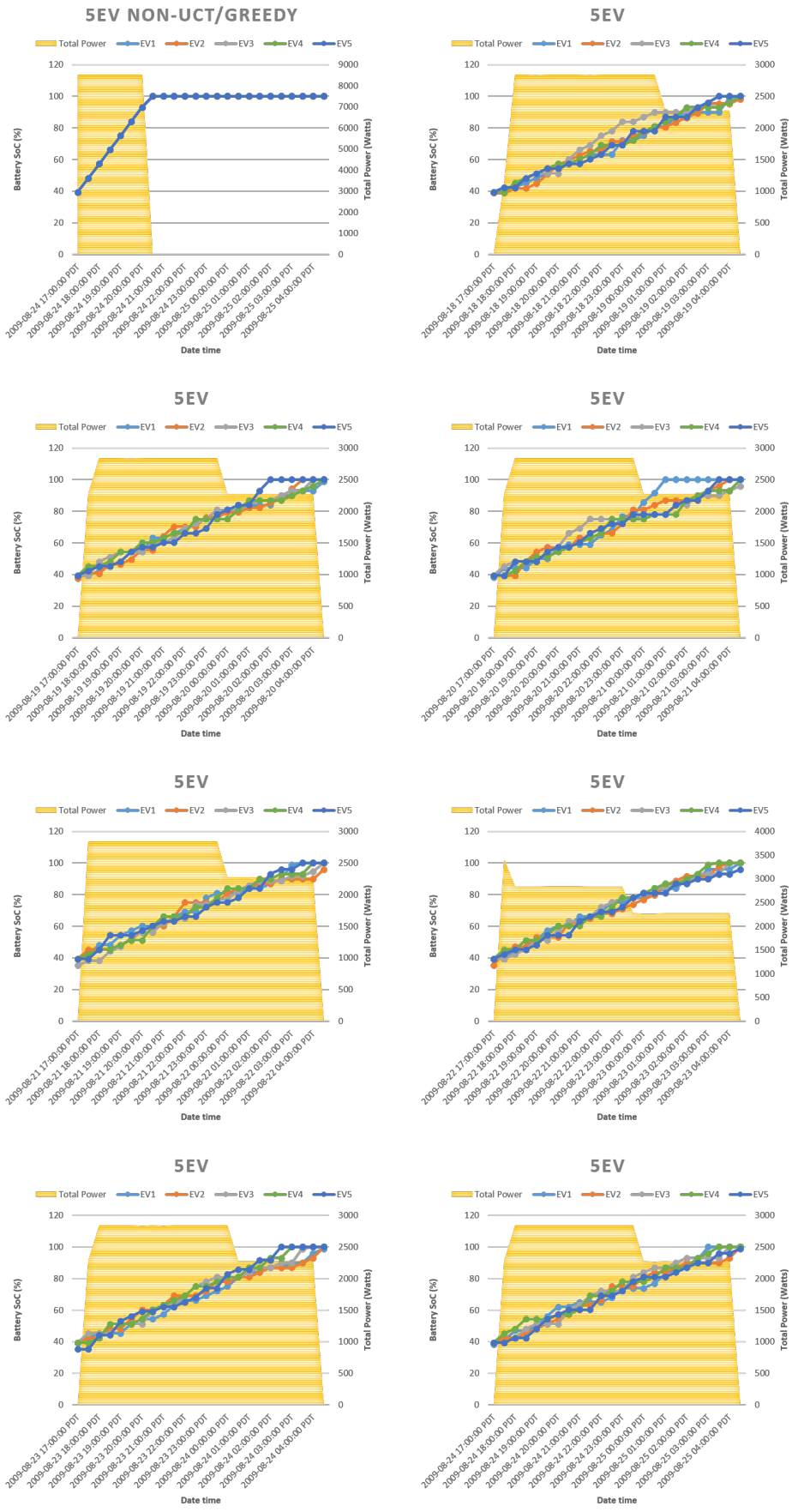


Table B.2: 5EVs daily SOC & total power to time relation (non-UCT vs. UCT each day).

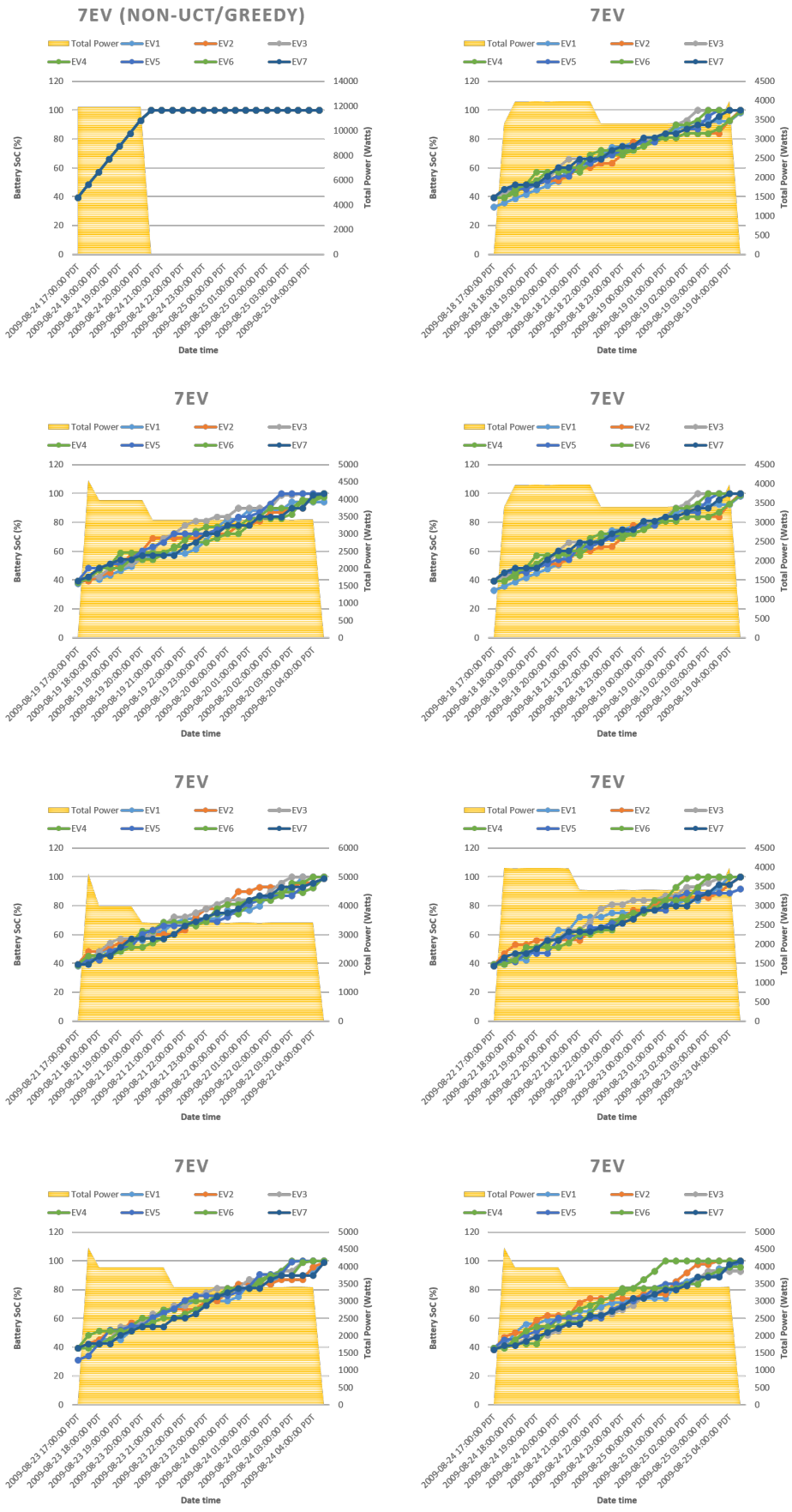


Table B.3: 7EVs daily SOC & total power to time relation (non-UCT vs. UCT each day).

Appendix C

UCT decision maker class diagram

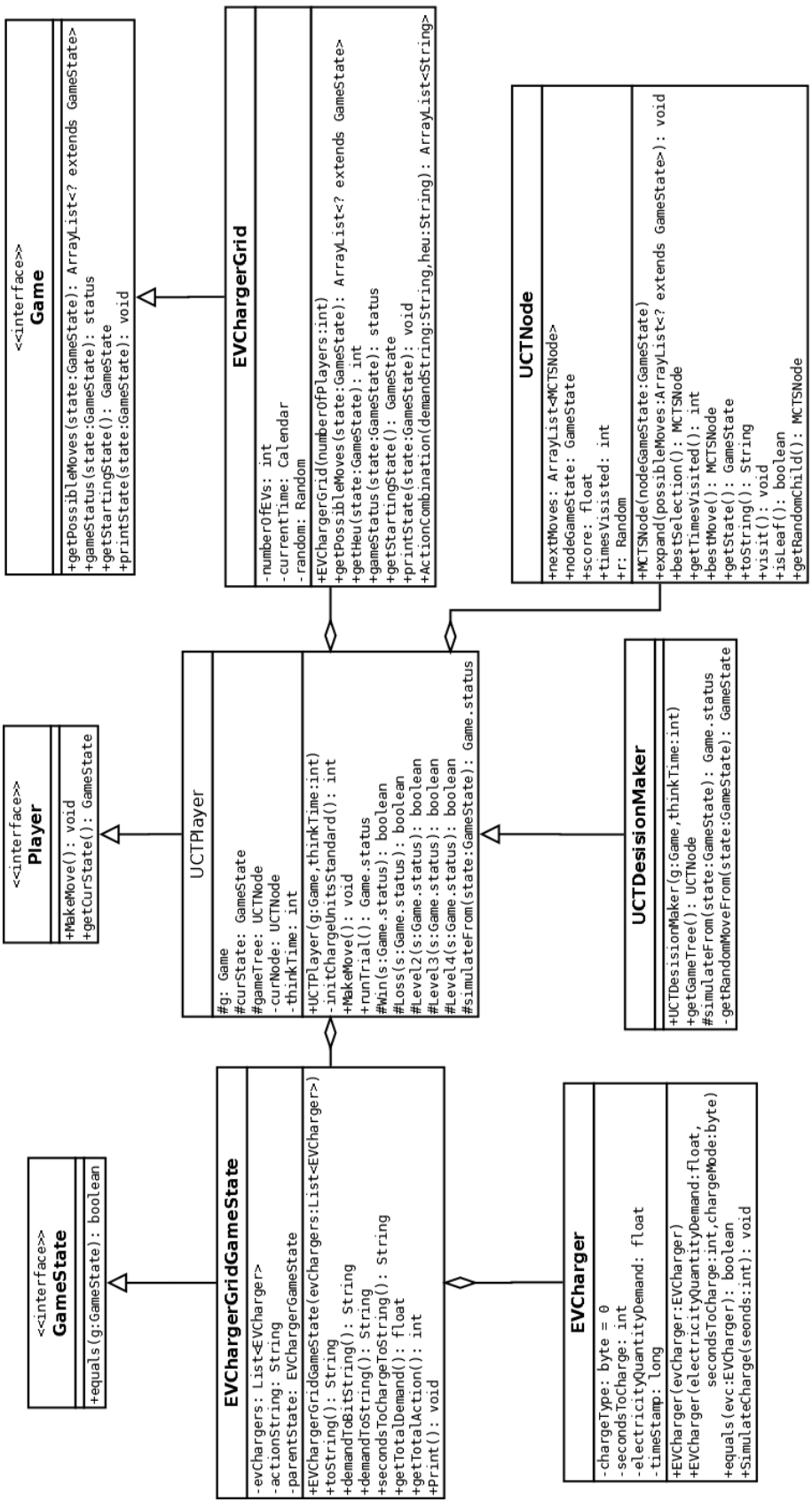


Figure C.1: UCT decision maker class diagram.

Bibliography

[1] M. Newborn and M. Newborn, *Kasparov vs. Deep Blue: Computer chess comes of age*. Springer-Verlag New York, Inc., 1997.

[2] T. P. Hughes, *Networks of power: electrification in Western society, 1880-1930*. JHU Press, 1993.

[3] F. E. R. Commission *et al.*, "Assessment of demand response and advanced metering," *Department of Energy, Washington, DC*, 2006.

[4] S. M. Kaplan, F. Sissine, and T. Net, *Smart Grid: Modernizing electric power transmission and distribution; Energy independence, Storage and security; Energy independence and security act of 2007 (EISA); Improving electrical grid efficiency, communication, reliability, and resiliency; integrating new and renewable energy sources*. The Capitol Net Inc, 2009.

[5] U. D. of Energy (DOE), "The smart grid: An introduction," 2008.

[6] C. Hicks, "The smart grid - where we are today and what the future holds," tech. rep., University of Michigan ERB Institute, 2012.

[7] J. Zhong, R. Zheng, W. Yang, and W. Felix, "Construction of smart grid at information age," *Power System Technology*, vol. 33, no. 13, pp. 12–18, 2009.

[8] V. Balijepalli, V. Pradhan, S. Khaparde, and R. M. Shereef, "Review of demand response under smart grid paradigm," in *Innovative Smart Grid Technologies - India (ISGT India), 2011 IEEE PES*, pp. 236–243, 2011.

[9] M. H. Albadi and E. El-Saadany, "Demand response in electricity markets: An overview," in *Power Engineering Society General Meeting, 2007. IEEE*, pp. 1–5, 2007.

[10] "Frequency adaptive, power-energy re-scheduler."

[11] A.-H. Mohsenian-Rad, V. Wong, J. Jatskevich, and R. Schober, "Optimal and autonomous incentive-based energy consumption scheduling algorithm for smart grid," in *Innovative Smart Grid Technologies (ISGT), 2010*, pp. 1–6, 2010.

[12] A.-H. Mohsenian-Rad, V. Wong, J. Jatskevich, R. Schober, and A. Leon-Garcia, "Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid," *Smart Grid, IEEE Transactions on*, vol. 1, no. 3, pp. 320–331, 2010.

[13] S. Caron and G. Kesidis, "Incentive-based energy consumption scheduling algorithms for the smart grid," in *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, pp. 391–396, 2010.

[14] H. K. Nguyen, J. Song, and Z. Han, "Demand side management to reduce peak-to-average ratio using game theory in smart grid," in *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pp. 91–96, 2012.

[15] A.-H. Mohsenian-Rad and A. Leon-Garcia, "Optimal residential load control with price prediction in real-time electricity pricing environments," *Smart Grid, IEEE Transactions on*, vol. 1, no. 2, pp. 120–133, 2010.

[16] D.-W. Lim, S.-J. Heo, and J.-S. No, "An overview of peak-to-average power ratio reduction schemes for ofdm signals," *Communications and Networks, Journal of*, vol. 11, no. 3, pp. 229–239, 2009.

[17] M. Newborough and P. Augood, "Demand-side management opportunities for the uk domestic sector," in *Generation, Transmission and Distribution, IEE Proceedings-*, vol. 146, pp. 283–293, IET, 1999.

[18] A. Gomes, C. Antunes, and A. Martins, "A multiple objective approach to direct load control using an interactive evolutionary algorithm," *Power Systems, IEEE Transactions on*, vol. 22, no. 3, pp. 1004–1011, 2007.

[19] G. Strbac, "Demand side management: Benefits and challenges," *Energy Policy*, vol. 36, no. 12, pp. 4419 – 4426, 2008. <ce:title>Foresight Sustainable Energy Management and the Built Environment Project</ce:title>.

[20] M. Engwall and A. Jerbrant, "The resource allocation syndrome: the prime challenge of multi-project management?," *International journal of project management*, vol. 21, no. 6, pp. 403–409, 2003.

[21] T. Morgan, "Smart grids and electric vehicles: Made for each other?," tech. rep., Organisation for Economic Cooperation and Development, 2012.

[22] A. Mohsenian-Rad, V. W. Wong, J. Jatskevich, R. Schober, and A. Leon-Garcia, "Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid," *Smart Grid, IEEE Transactions on*, vol. 1, no. 3, pp. 320–331, 2010.

[23] W. Saad, Z. Han, H. V. Poor, and T. Basar, "Game theoretic methods for the smart grid," *CoRR*, vol. abs/1202.0452, 2012.

[24] T. Basar, G. J. Olsder, G. Clsder, T. Basar, T. Baser, and G. J. Olsder, *Dynamic noncooperative game theory*, vol. 200. SIAM, 1995.

[25] Z. Han, D. Niyato, W. Saad, T. Basar, and A. Hjorungnes, *Game theory in wireless and communication networks*. Cambridge University Press, 2012.

[26] H. P. Young, *Strategic learning and its limits*, vol. 2002. Oxford University Press on Demand, 2004.

[27] T. Basar, G. J. Olsder, G. Clsder, T. Basar, T. Baser, and G. J. Olsder, *Dynamic noncooperative game theory*, vol. 200. SIAM, 1995.

[28] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic game theory*. Cambridge University Press, 2007.

[29] D. Fudenberg and D. K. Levine, *The theory of learning in games*, vol. 2. MIT press, 1998.

[30] J. S. Shamma and G. Arslan, "Unified convergence proofs of continuous-time fictitious play," *Automatic Control, IEEE Transactions on*, vol. 49, no. 7, pp. 1137–1141, 2004.

[31] L. Rose, S. Lasaulce, S. M. Perlaza, and M. Debbah, "Learning equilibria with partial information in decentralized wireless networks," *Communications Magazine, IEEE*, vol. 49, no. 8, pp. 136–142, 2011.

[32] A.-H. Mohsenian-Rad and A. Leon-Garcia, "Optimal residential load control with price prediction in real-time electricity pricing environments," *Smart Grid, IEEE Transactions on*, vol. 1, no. 2, pp. 120–133, 2010.

[33] A.-H. Mohsenian-Rad, V. Wong, J. Jatskevich, and R. Schober, "Optimal and autonomous incentive-based energy consumption scheduling algorithm for smart grid," in *Innovative Smart Grid Technologies (ISGT), 2010*, pp. 1–6, 2010.

[34] J. Lagorse, D. Paire, and A. Miraoui, "A multi-agent system for energy management of distributed power sources," *Renewable Energy*, vol. 35, no. 1, pp. 174 – 182, 2010.

[35] M. Pipattanasomporn, H. Feroze, and S. Rahman, "Multi-agent systems in a distributed smart grid: Design and implementation," in *Power Systems Conference and Exposition, 2009. PSCE '09. IEEE/PES*, pp. 1–8, 2009.

[36] M. Zhou, Y. Gao, and G. Li, "Study on improvement of available transfer capability by demand side management," in *Electric Utility Deregulation and Restructuring and Power Technologies, 2008. DRPT 2008. Third International Conference on*, pp. 545–550, 2008.

[37] Y. Ou and C. Singh, "Assessment of available transfer capability and margins," *Power Systems, IEEE Transactions on*, vol. 17, no. 2, pp. 463–468, 2002.

[38] M. Zhou, G. Li, and P. Zhang, "Impact of demand side management on composite generation and transmission system reliability," in *Power Systems Conference and Exposition, 2006. PSCE '06. 2006 IEEE PES*, pp. 819–824, 2006.

[39] computer go.info, "Human-computer go challenges." `http://http://www.computer-go.info/h-c/index.html#2013`, August 2013.

[40] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *Computational Intelligence and AI in Games, IEEE Transactions on*, vol. 4, no. 1, pp. 1–43, 2012.

[41] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[42] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, pp. 235–256, May 2002.

[43] G. M. J.-B. Chaslot, S. Bakkes, I. Szita, and P. Spronck, "Monte-Carlo Tree Search: A New Framework for Game AI," in *Proc. Artif. Intell. Interact. Digital Entert. Conf.*, (Stanford Univ., California), pp. 216–217, 2008.

[44] L. Kocsis, C. Szepesvári, and J. Willemson, "Improved monte-carlo search," *Univ. Tartu, Estonia, Tech. Rep*, vol. 1, 2006.

[45] D. Chassin and S. Widergren, "Simulating demand participation in market operations," in *Power Energy Society General Meeting, 2009. PES '09. IEEE*, pp. 1–5, 2009.

[46] D. Wang, B. de Wit, S. Parkinson, J. Fuller, D. Chassin, C. Crawford, and N. Djilali, "A test bed for self-regulating distribution systems: Modeling integrated renewable energy and demand response in the gridlab-d/matlab environment," in *Innovative Smart Grid Technologies (ISGT), 2012 IEEE PES*, pp. 1–7, 2012.

[47] Z. Zhou, F. Zhao, and J. Wang, "Agent-based electricity market simulation with demand response from commercial buildings," *Smart Grid, IEEE Transactions on*, vol. 2, no. 4, pp. 580–588, 2011.

[48] S. Widergren, K. Subbarao, D. Chassin, J. Fuller, and R. Pratt, "Residential real-time price response simulation," in *Power and Energy Society General Meeting, 2011 IEEE*, pp. 1–5, 2011.

[49] J. Fuller, K. Schneider, and D. Chassin, "Analysis of residential demand response and double-auction markets," in *Power and Energy Society General Meeting, 2011 IEEE*, pp. 1–7, 2011.