Tree-based Analysis of Mesh Overlays for Peer-to-Peer Streaming

Bartosz Biskupski[1], Marc Schiely[2], Pascal Felber[2], René Meier[1]

[1]Trinity College Dublin, Ireland
[2]University of Neuchâtel, Switzerland

Abstract. Mesh-based P2P streaming approaches have been recently proposed as an interesting alternative to tree-based approaches. However, many properties of mesh overlays remain little understood as they are difficult to study due to the lack of a predefined structure. In this paper we show that when data is streamed through mesh overlays, it follows tree-based diffusion patterns and thereby mesh-based streaming can be studied in a similar manner to tree-based approaches. We identify properties of the diffusion trees that emerge in mesh overlays and compare them to optimal diffusion trees. We show that the emerging diffusion trees exhibit suboptimal height and are unbalanced, which results in increased buffering delay of mesh-based P2P systems, particularly in heterogeneous environments. We present an algorithm that adapts the mesh overlay to shorten diffusion trees and to reduce the buffering delay.[1]

1 Introduction

The use of peer-to-peer (P2P) overlays for multicast media streaming has gained significant attention in recent years as it alleviates scalability problems of centralised client-server architectures and weaknesses that prevent a wide adoption of IP Multicast. Two main approaches for building overlays for P2P multicast media streaming are tree-based [1] and mesh-based [2,3,4]. The former approach explicitly places peers in a single tree or multiple multicast trees, where they receive the stream from their parent(s) and forward it to their children. In the mesh-based approach, the P2P overlay is unstructured, formed by peers connecting to neighbours, which may be randomly selected. The media stream is typically split into small data blocks that are exchanged between neighbouring peers, resulting in their propagation throughout the overlay. The main advantage of mesh overlays compared to tree-based overlays is their much higher robustness to peer churn. In tree-based approaches, a peer can receive data only from its specified parent and when that parent fails or leaves the network, its whole

sub-tree loses that data until the tree is reconstructed. In mesh-based streaming systems, data chunks can be obtained from any neighbour that holds it and thus when one neighbour fails, other neighbours may still provide the data. For that reason, many researches focus on mesh overlays for P2P streaming. However, one problem posed by mesh overlays is that they do not rely on any predefined network structure and thereby are more difficult to study than tree-based overlays. In this paper, we show that when data chunks are streamed over mesh overlays, tree-based diffusion patters dynamically emerge in the overlay. These tree-based patterns of diffusion can be studied in the same manner as tree-based overlay structures. The contribution of this paper is that we identify and analyse properties of the emerging tree structures in mesh overlays and, in order to evaluate their performance, we compare them to optimal diffusion trees in both homogeneous and heterogeneous environments. This provides insights into how mesh overlays can be adapted to reduce buffering delay in mesh-based streaming systems to a theoretical minimum. Based on this analysis we developed an algorithm that reduces diffusion tree heights in a mesh overlay and thus, also reduces buffering delay.

The paper is organised as follows: in Section 2 different approaches to the analysis of mesh-based streaming systems are presented. Section 3 shows how diffusion trees emerge in mesh overlays and analyses these diffusion trees. Finally, the adaptation algorithm is presented and evaluated in Section 4 before the paper concludes in Section 5.

2 Related Work

Many mesh-based P2P streaming systems have been proposed in the last few years [2,3,4], but none of them has been formally analysed due to their complexity.

Chunkyspread [5] is one example of an unstructured approach to media streaming. It uses a multi-tree (multi-description) based structure on top of an unstructured overlay. The structure is very dynamic as each peer periodically searches for new partners in its local environment. Peers exchange information (load, latency, creation of loops) with their neighbours to search for the best parent-child pairs for each tree. The constraints on these relationships are (1) to avoid loops, (2) to satisfy any tit-for-tat constraints, (3) to adapt load (shall be in a per peer defined range) and (4) to reduce latency. The loop-preventing algorithm which is run on the overlay ensures that chunks are distributed following a multi-tree structure. In this paper we argue, that trees do not need to be built explicitly, but that they are inherent to the mesh structure.

In contrast, SplitStream [1] is a tree-based P2P media streaming architecture that focuses on robustness. Different to our model, the stream is split into multiple stripes that can be distributed independently. A distinct tree is constructed for each of these stripes on all the participating peers. The robustness in SplitStream comes from the fact that each node is an inner node in at most one tree and a leaf node in all the other trees. Thus, if a peer fails, only one

distribution tree is affected and has to be rebuilt. In our model a tree structure close to SplitStream is derived from a mesh-based approach. Peers are also inner nodes in only one tree and leaf nodes in all others. Due to the mesh structure, trees are dynamically built and adapted if nodes fail or bandwidth conditions change.

A comparative study of tree- and mesh-based approaches for media streaming is presented in [6]. Authors first propose an organised view of data delivery in mesh overlays, which consists of data diffusion and swarming phases, and later introduce delivery trees, which they discover in mesh overlays in a similar fashion to diffusion trees described in our paper. Our work is different in that we focus on formally analysing properties of diffusion trees rather than evaluating them by simulation. We also propose an overlay adaptation algorithm that improves properties of these trees.

A different approach to analysing P2P media streaming systems are fluid models. In [7] the authors present a stochastic fluid model that takes into account peer churn, heterogeneous peer upload capacities, peer buffering and delays. In this paper we analyse the distribution trees created in a mesh such that known adaptations for tree-based approaches can be applied to meshes.

In [8] tree-based P2P streaming systems are analysed and it is shown that moving high-bandwidth nodes close to the source is advantageous and leads to high performance gains in terms of total download capacity. We show in this paper that the same holds for mesh-based systems and that trees can be shortened by adapting the location of high-bandwidth nodes in diffusion trees.

3 Mesh-based P2P Streaming

The mesh-based approach to data streaming originates from research on gossip and epidemic protocols, where nodes periodically exchange information among each other, which results in the eventual dissemination of all information to all nodes. The BitTorrent [9] file-sharing system popularised this approach for the dissemination of large volumes of data from a transmitter to all receivers. BitTorrent creates an unstructured overlay mesh to distribute a data file. A file is divided into chunks, which are exchanged by nodes in a pull-based fashion until nodes can reconstruct the original file.

In contrast to file-sharing systems, the transmitter in live P2P streaming protocols does not have access to the entire data as it is generated "live", and thus, it cannot split the whole data into chunks for distribution throughout the network. In order to leverage mesh-based delivery, streaming protocols require a delay between the stream creation time at the transmitter and the receiver playback time. The data stream produced within this delay is split into small chunks and distributed throughout the network similar to the way chunks of an entire file are distributed in mesh-based file-sharing protocols. Nodes maintain sliding windows that reflect this delay and capture which chunks have already been received and which are still missing. The buffers move forward with the speed of the original video transmission rate, which is discovered by all nodes

from the video stream. The beginning of the buffer points at the chunk currently being played at the receiving node and the end of the buffer reflects the chunk currently generated at the transmitting node. Chunks that do not arrive in time (outside the sliding window) are lost and cause video playback degradation.

A mesh overlay is created in a random fashion by joining nodes connecting with selected nodes. The selection of neighbours can be based on different strategies, e.g. random or bandwidth-based. Neighbouring nodes maintain local knowledge about data chunks they possess by informing each other whenever they receive a new chunk. The missing chunks are requested from neighbours immediately or periodically, following a chunk selection algorithm. Different strategies such as most-recent-chunk-first, rarest-chunk-first or random can be used to schedule the chunk requests.

3.1 Mesh Overlay Properties

In our previous research on mesh overlay adaptation [10,11], we identified that completely random mesh overlays limit the network throughput by underutilising the available upload bandwidth at peers. Limited network throughput in turn reduces possible video streaming rates and the corresponding video quality. We showed properties of mesh overlays that, when satisfied, optimise the network throughput. This requires that each peer maintains two sets of neighbours - (1) children, which are the neighbours to which data is uploaded and (2) parents, which are the neighbours from which data is downloaded. The network throughput is optimised in such a directed mesh overlay when:

- Each peer has a constant (configurable) number of parents
- Each peer has a number of children proportional to its upload bandwidth

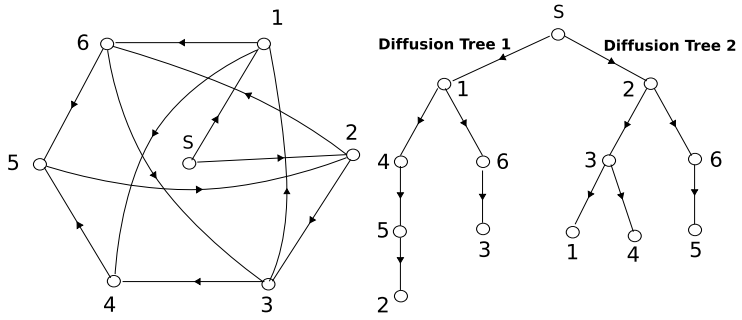
We showed in [10] that a mesh overlay satisfying these two conditions optimises the upload bandwidth utilisation and enables all peers to download at the maximum possible global video streaming rate. We also proposed algorithms for adapting the mesh overlay to satisfy these conditions. In this paper, we conduct our analysis on directed mesh overlays that satisfy these two conditions and thus we can provide a fair comparison to multiple-tree-based overlays that also optimise the network throughput. This paper is novel in that we show how diffusion trees emerge in these adapted directed mesh overlays; we analyse properties of diffusion trees and compare them to those of multiple-tree-based overlays; and finally, propose an algorithm that improves these properties.

3.2 Tree-based View of Mesh Overlays

Mesh overlays are very dynamic and thus are difficult to analyse. In contrast, trees are well understood and it is easier to derive properties of trees. Meshes can be seen as a structure of multiple trees if we assume that bandwidth of all peers remain constant over time and that the chunk selection algorithm is deterministic. We assume that peers request missing chunks from parents immediately

when they are notified of them, following a most-recent-chunk-first strategy, i.e., when a decision is made between two chunks, a chunk with a more recent time-stamp is requested. This chunk request strategy is based on an observation that most recent generated chunks are also the rarest in the overlay and thus need to be given priority for distribution.

We assume that the stream rate is set to the maximum rate supported by the overlay such that all peers can receive it, i.e., equal to $\frac{\sum_i^N upload_i}{N-1}$, where N is the total number of peers including the source node (the source uploads, but does not download data). We also assume that the mesh overlay satisfies conditions discussed in Section 3.1 and that a peer's upload bandwidth is shared equally by all its connections. Under such assumptions, upload of all peers is saturated and the upload rate of each link is the same, equal to $\frac{\sum_i^N upload_i}{(N-1)*K}$, where K is a globally configurable number of parents of each peer. From this follows that each chunk is transferred over a link in time $\frac{s*(N-1)*K}{\sum_i^N upload_i}$, where s is the size of a chunk. The source node generates a new chunk every $\frac{s*(N-1)}{\sum_i^N upload_i}$ time units, so by the time a single chunk is transferred to a child, K new chunks are generated. Since it is desired that the source node sends different chunks to different children (to distribute chunks equally in the overlay), we use a round-robin strategy to push chunks from the source node to its direct children in which the ith child receives chunks with sequence numbers $t_0 + j * K + (i \bmod K)$, for some initial t_0 and $j = 0, 1, 2, 3, ...$ Peers, which are not direct children of the source node, request the most recently generated missing chunks, so they always request a missing chunk that travelled the least number of hops (and time). Effectively, K diffusion trees emerge, where each tree propagates every Kth chunk. This process of diffusion trees emerging in a mesh overlay, which has properties outlined in Section 3.1, is illustrated in Figure 1 for $K = 2$.



(a) Directed mesh overlay where each peer has 2 parents (b) Corresponding diffusion trees

Fig. 1. Mesh overlay and its two diffusion trees.

3.3 Analysis

In this section we show how optimal multiple trees are constructed in both homogeneous and heterogeneous environments and analyse their heights in order to compare them, in the next subsection, to diffusion trees emerging in mesh overlays.

Height of optimal trees in a homogeneous environment. First, we analyse a homogeneous environment, where all peers have the same upload capacity. Optimal K distribution trees can be created by placing each peer as an inner node in exactly one tree and as a leaf node in the other $K-1$ trees. Thus, each peer has K parents, one in each optimal distribution tree. In a homogeneous environment, this means that the out-degree d of each peer is equal to K. Since a peer has children in only one tree, K and d are the number of children of each inner node in each tree. Thus, the height of each of K optimal distribution trees in a homogeneous environment with N nodes is equal to the height $H(d,N)$ of an evenly balanced tree with N nodes and out-degree d, which is calculated using a relation

$$\sum_{i=0}^{H(d,N)} d^i = N$$

based on the fact that there are d^i peers at tree level i. Solving this geometric sequence gives an equation for the height of a balanced homogeneous tree:

$$H(d,N) = log_d\left((d-1)*N+1\right) - 1 \tag{1}$$

Therefore, the height of each of K optimal trees in a homogeneous environment is given by $H(K,N)$. In this paper we also use an equation for the number of leaf nodes $L(d,N)$ in a balanced homogeneous tree with N nodes and out-degree d, given by

$$L(d,N) = d^{H(d,N)} = \frac{(d-1)*N+1}{d} \tag{2}$$

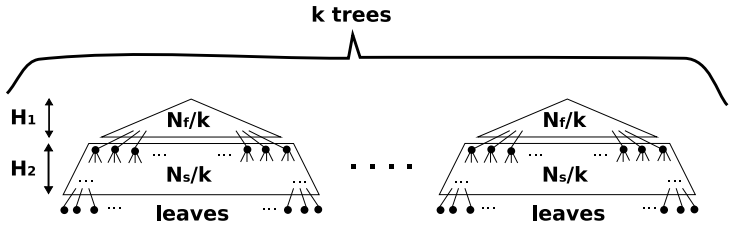


Fig. 2. Optimal construction of K trees consisting of fast and slow nodes.

Height of optimal trees in a heterogeneous environment. We study the construction of optimal trees in a heterogeneous environment by using two types of peers

- N_s slow peers and N_f fast peers, where a fast peer has upload bandwidth i times higher than a slow peer. In such a scenario, the optimal placement of peers that minimises the height of each of the K trees is presented in Figure 2. Similar to homogeneous environments, each peer is an inner node in exactly one tree and a leaf node in $K-1$ trees. Additionally, fast nodes are placed at the top of the trees in order to reduce the height of the trees. Slow nodes have out-degree d, while fast nodes can upload i times faster, so their out-degree is $i*d$. The out-degree of slow and fast nodes is derived from the fact that the total number of outgoing links of all peers must be equal to the total number of incoming links in the P2P overlay, while taking into account that the source node has out-going links, but does not have any incoming links. From this we have $N_s*d+N_f*i*d=K*(N_s+N_f-1)$, which gives

$$d=\frac{K*(N_s+N_f-1)}{i*N_f+N_s} \tag{3}$$

The height H_{het} of each heterogeneous tree constructed as in Figure 2 is calculated as $H_{het}=H_1+H_2+1+1$, which is the sum of the height H_1 of the upper part of the tree composed of inner fast nodes only, the height H_2 of the lower part of the tree composed of slow inner nodes only, plus one level between the two parts of the tree and one level for the peers that are leaves in the tree (and which are inner nodes in other trees). The height H_1 is calculated using Eq. 1 as the height of a homogeneous tree of N_f/K fast nodes with out-degree $i*d$:

$$H_1=H(i*d,\frac{N_f}{K})=log_{i*d}\left((i*d-1)*\frac{N_f}{K}+1\right)-1$$

The height H_2 is calculated as the height of a homogeneous tree of $\frac{N_s/K}{L_1*i*d}$ slow nodes with out-degree d

$$H_2=H(d,\frac{N_s/K}{L_1*i*d})=log_d\left((d-1)*\frac{N_s/K}{L_1*i*d}+1\right)-1$$

where $L_1=L(i*d,\frac{N_f}{K})$ is the number of leaves in the upper part, i.e., H_1. From these equations we derive a formula for the optimal height H_{het} of each optimal heterogeneous diffusion tree

$$H_{het}=log_{i*d}\left((i*d-1)*\frac{N_f}{K}+1\right)+log_d\left((d-1)\,\frac{N_s}{(i*d-1)*N_f+K}+1\right) \tag{4}$$

where d is the out-degree of a slow node given by Eq. 3.

3.4 Evaluation

We compare the optimal tree heights, calculated in Equation 4, to the average height of diffusion trees that emerge in mesh overlays and are calculated by our custom-built simulator of mesh overlays. The simulator relies on the assumptions

outlined in Sections 3.1 and 3.2. We used 50,000 nodes and studied both a homogeneous environment and environments with different levels of heterogeneity. In experiments involving heterogeneity, 10% of all nodes are fast nodes with upload bandwidth 2 and 8 times higher than the remaining slow nodes. The overall upload bandwidth in all overlays is the same. The results are presented in Figure 3. The results show that the average height of diffusion trees in homogeneous mesh overlays is around 2 levels above the optimal height, for all K. The reason for that is that in the optimal tree each peer is an inner node in exactly one diffusion tree, whereas in the trees emerging in mesh overlays a peer is located randomly and can be an inner node in several trees. The results show that when the level of heterogeneity increases, the gap between the height of diffusion trees in the mesh overlay and optimum trees significantly increases. For the case with 10% of peers being 8 times faster than the remaining slow peers, the average height of a diffusion tree in the mesh overlay for $K = 2$ is 3 times higher than the optimum and drops to 2 times over the optimum for $K = 16$. Increased heterogeneity results in higher importance of the location of fast and slow peers in the tree. Worse performance for small K, in turn, is caused by higher variation in the height of diffusion trees - some leaves are much lower or higher than the others. This tree imbalance can be observed in Figure 4 that shows the cumulative distribution function (CDF) of the depth of leaf nodes in diffusion trees that emerge in a mesh overlay for both homogeneous and heterogeneous environments. The highest diffusion tree imbalance is for small K.

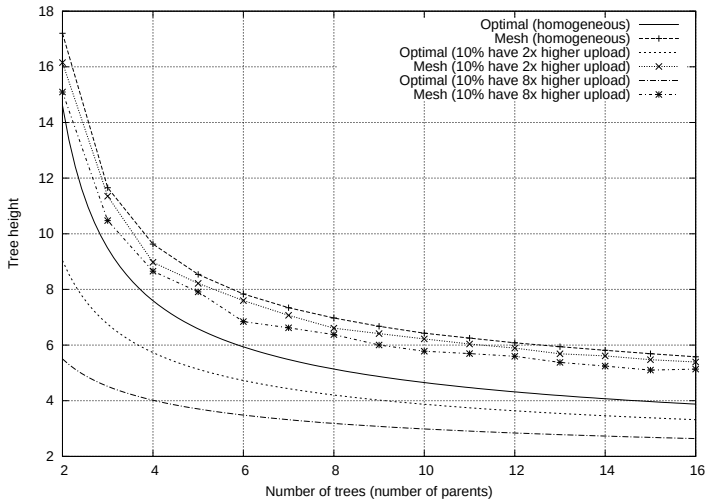


Fig. 3. Average tree height for different number of parents and different heterogeneity levels.

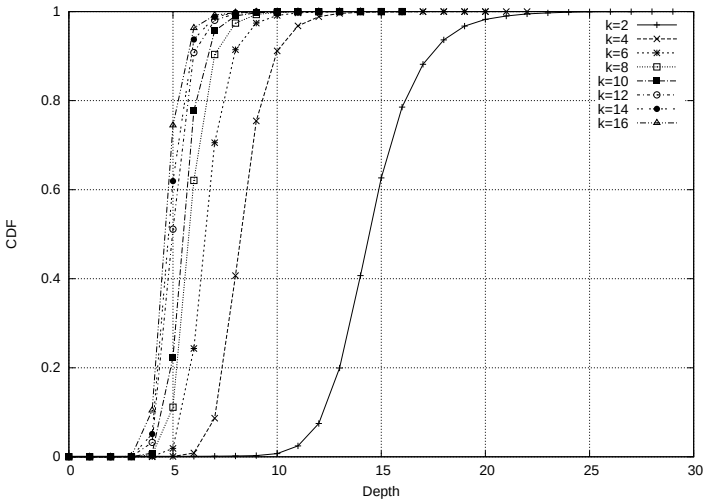


Fig. 4. CDF of the height of diffusion trees in mesh overlays in a heterogeneous (10% peers have 4x upload) environment.

Chunk propagation delay. In order to measure the impact of the tree height on the buffering delay, we analyse the time required to propagate a chunk through the diffusion trees in mesh overlays. Since in a mesh overlay, a peer can be placed anywhere in each diffusion tree, its buffering delay needs to accommodate the maximum difference between chunk arrivals in each distribution tree, which is equal to the chunk propagation delay. The propagation delay can be calculated as

$$delay = \frac{H * s * K * (N-1)}{\sum_i^N upload_i}$$

where H is the height of the tree, s the size of a chunk and the remaining part of the formula derives from the equation for the bandwidth of a link (see Section 3.2). It can be observed that this delay represents a trade-off between the height of a tree and the number K of distribution trees. Larger K produce shorter trees, however, it takes longer for a node to upload a chunk to all its children (since a node has more children). Smaller chunk sizes allow for their faster propagation, but more control messages are required to notify/request chunks. Propagation delay as a function of the number of diffusion trees (peer parents) is shown in Figure 5 (for an average upload bandwidth of 1,000kbps and a chunk size of 4KB). The results show that a small number of diffusion trees result in shorter buffering delays. However, small number of diffusion trees also means that the number of parents of each peer is small and this reduces robustness to peer failures.

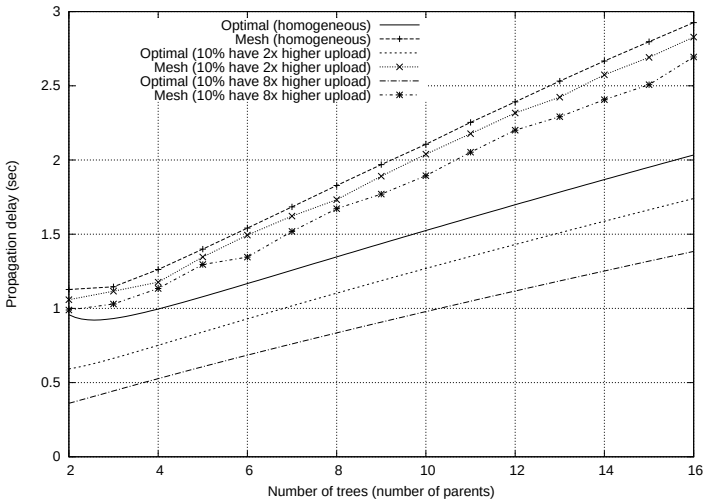


Fig. 5. Propagation delay for varying number of trees for mesh overlays and the optimal case.

4 Mesh Adaptation Algorithm

In the previous sections we showed that the heights of diffusion trees in mesh overlays are much higher than the optimal height. In this section we present an algorithm that adapts the location of high-bandwidth peers dynamically. To shorten tree lengths it is advantageous to place high-bandwidth nodes near the source and low-bandwidth peers near the leaves.

4.1 Algorithm

We assume that peers have accurate information about their bandwidth, either through user input or through passive measurement techniques, such as [12]. Furthermore, the assumption is made that techniques are deployed that prevent peers from cheating about their bandwidth. To do this, peers may for example team up to compare effective bandwidth of neighbours with their indicated bandwidth and drop links to cheaters if the difference is too high. Alternatively, a reputation system like [13] could be implemented.

Each chunk being distributed from the source s to a peer p contains a hop count of the path it travelled. Peers can use this hop count as an estimate of their distance to the source. As explained in previous sections, the goal of each peer is to climb up, respectively to its upload bandwidth, in one diffusion tree and to become a leaf node in all other diffusion trees. In order to achieve this, each peer periodically executes Algorithm 1, which improves a peer's position in one diffusion tree. Since each parent of a peer is responsible for delivering only one tree, the algorithm aims at improving the peer's position by replacing its current

best parent (nearest to the source) with one of its grandparents that is closer to the source, subject to the conditions discussed below, effectively moving higher in one tree. Specifically, a peer p tries to find its parent *parent* and a grandparent *grandparent* (a parent of *parent*) that satisfies the following conditions:

1. distance(*grandparent*) $<$ distance(bestparent(p))
2. upload(p) $>$ upload(*parent*) OR bestparent(*parent*) $\neq$ *grandparent*

The first condition requires that *grandparent* is closer to the source than the current best parent. The second condition requires that the upload bandwidth of peer p is greater than the upload bandwidth of *parent* (child of *grandparent*) or *grandparent* is not the best parent of *parent* (*parent* does not climb up in that tree) and thus, *parent* can give up that *grandparent*. If these two conditions are satisfied, then peer p climbs up one level by: replacing *parent* as a child of *grandparent*, becoming a new parent of *parent* and losing one child, which becomes a child of *parent* (Figure 6 shows the exchange protocol). This way, the number of children and parents of all peers involved (p, *parent* and *grandparent*) remain unchanged and thus, the properties of the overlay required for achieving the optimal network throughput, described in Section 3.1, remain satisfied. The presented adaptation algorithm effectively results in each peer climbing up in one tree as long as its parent in this tree has lower upload bandwidth and climbing down in other trees (by giving up its position in these other trees to its children that climb up in these trees). The algorithm does not affect the network throughput as it does not change the number of children or parents of any peer.

Algorithm 1 Adapting position of peer p in the mesh overlay

```
for all parent ← parent(p) do
  for all grandparent ← parent(parent) do
    if parent ≠ source then
      if distance(grandparent) < distance(bestparent(p)) then
        if upload(p) > upload(parent) or bestparent(parent) ≠ grandparent
        then
          exchangePosition(p, parent, grandparent)
        end if
      end if
    end if
  end for
end for
```

4.2 Evaluation

In this section, we show the results of our evaluation of the adaptation algorithm presented in Section 4.1. The algorithm was implemented in our custom-built

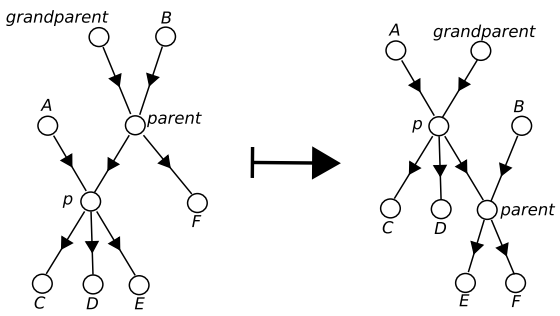


Fig. 6. Peers p and *parent* exchange their positions respectively to *grandparent*.

simulator and executed on 50,000 nodes with different ratios of upload bandwidth of fast and slow nodes. First, an initial mesh was created and tree heights calculated. Then, Algorithm 1 was executed to adapt the positions of all peers until no more adaptations were possible.

In all experiments 10% of all peers had i ($i = \{2, 8\}$) times higher upload bandwidth than the remaining peers. The number of trees K varied from 2 to 16. As can be seen in Figure 7, there is a significant benefit of placing high-bandwidth nodes near the source. The average tree heights decrease by about 35% for two trees ($K = 2$). The same improvement is in the buffering delay, which is proportional to the tree height. Figure 8 shows the cumulative distribution function (CDF) of the depth of leaf nodes in diffusion trees in adapted mesh overlays. This figure, when compared to the analogous Figure 4, shows that diffusion trees in the adapted mesh overlays are significantly more balanced. However, despite of much improvement, some imbalance in the diffusion tree heights remains and, for that reason, the height of diffusion trees (and the corresponding buffering delay) is suboptimal. To achieve optimal diffusion trees, a more system-wide adaptation is required, which is a focus of our future work.

5 Conclusions

In this paper we analysed data diffusion in mesh overlays. We showed that data chunks follow dynamically formed diffusion trees and analysed properties of these trees. The proposed structured view of meshes allows us to apply knowledge about trees directly to mesh-based streaming approaches. Our results show that diffusion trees in mesh overlays are unbalanced with suboptimal height and thereby, buffering delay in mesh overlays is suboptimal. With the increasing heterogeneity in an overlay, the diffusion trees become even more suboptimal due to imperfect placement of fast peers in the diffusion trees. This implies that a mesh adaptation algorithm that places fast nodes closer to the source in exactly one diffusion tree shortens the height and improves the balance of diffusion trees, thereby significantly reducing the data buffering delay. We presented such a mesh adaptation algorithm and showed that it improves tree heights. In future

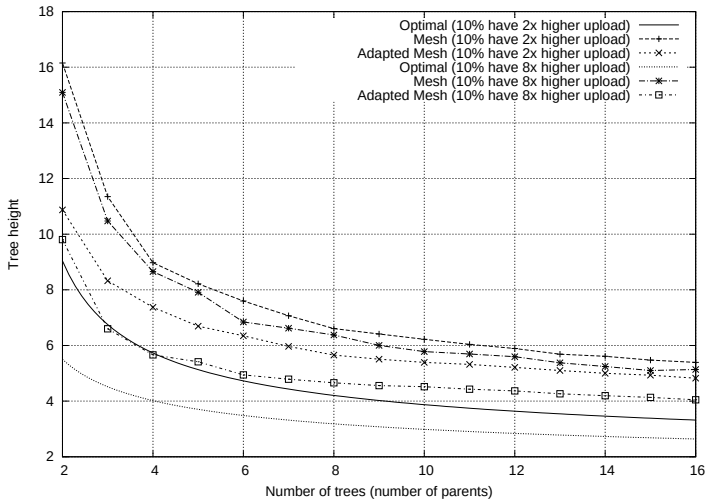


Fig. 7. Average tree heights for different proportions of upload bandwidth and 50,000 peers.

work the algorithm will be enhanced to better balance the height of diffusion trees, implemented in our prototypes and experimentally evaluated to show its effectiveness in real-world scenarios.

Acknowledgements

This work is supported in part by MiNEMA, ESF, Swiss National Foundation Grant 102819 and Enterprise Ireland under the Commercialisation Proof of Concept Programme (MeshTV).

References

1. Castro, M., Druschel, P., Kermarrec, A.M., Nandi, A., Rowstron, A., Singh, A.: SplitStream: High-bandwidth multicast in cooperative environments. In: SOSP'03: Proceedings of the nineteenth ACM Symposium on Operating Systems Principles, New York, NY, USA (2003) 298–313
2. Pai, V.S., Kumar, K., Tamilmani, K., Sambamurthy, V., Mohr, A.E.: Chainsaw: Eliminating trees from overlay multicast. In: IPTPS. (2005) 127–140
3. Magharei, N., Rejaie, R.: PRIME: Peer-to-peer receiver-driven mesh-based streaming. In: 26th Annual IEEE Conference on Computer Communications IEEE INFOCOM 2007. (2007)
4. Pianese, F., Perino, D., Keller, J., Biersack, E.: PULSE: an adaptive, incentive-based, unstructured p2p live streaming system. IEEE Transactions on Multimedia, Special Issue on Content Storage and Delivery in Peer-to-Peer Networks **9**(6) (2007)

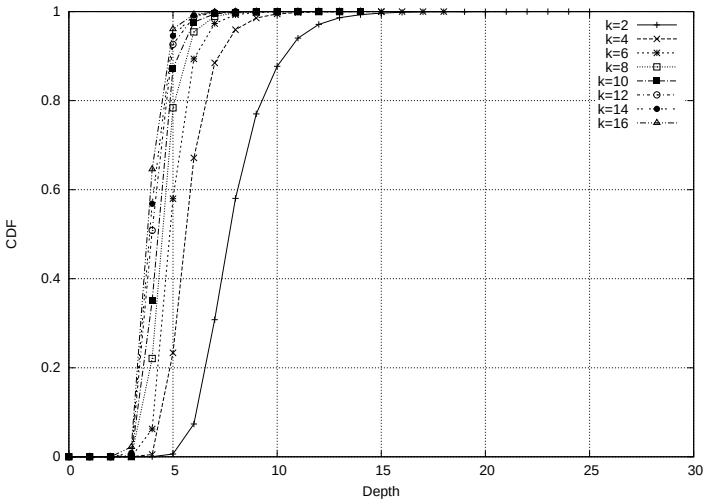


Fig. 8. CDF of the height of diffusion trees in adapted mesh overlays in a heterogeneous (10% peers have 4x upload) environment.

5. Venkatraman, V., Yoshida, K., Francis, P.: Chunkyspread: Heterogeneous unstructured end system multicast. In: Proceedings of 14th IEEE International Conference on Network Protocols. (Nov 2006)
6. Magharei, N., Rejaie, R., Guo, Y.: Mesh or multiple-tree: A comparative study of live p2p streaming approaches. In: Proceedings of 26th IEEE International Conference on Computer Communication (INFOCOM). (May 2007) 1424–1432
7. Kumar, R., Liu, Y., Ross, K.: Stochastic fluid theory for p2p streaming systems. In: Proceedings of 26th IEEE International Conference on Computer Communication (INFOCOM). (May 2007) 919–927
8. Schiely, M., Renfer, L., Felber, P.: Self-organization in cooperative content distribution networks. In: Proceedings of IEEE International Symposium on Network Computing and Applications (NCA). (Jul 2005) 109–116
9. Cohen, B.: Incentives build robustness in BitTorrent. In: the 1st Workshop on Economics of Peer-to-Peer Systems, Berkeley, CA, USA (June 2003)
10. Biskupski, B., Cunningham, R., Dowling, J., Meier, R.: High-bandwidth mesh-based overlay multicast in heterogeneous environments. In: AAA-IDEA '06: Proceedings of the 2nd International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications, ACM Press (2006) 4–11
11. Biskupski, B., Cunningham, R., Meier, R.: Improving throughput and node proximity of p2p live video streaming through overlay adaptation. In: Proceedings of the 9th IEEE International Symposium on Multimedia (ISM 2007), IEEE Computer Society (2007) 245–252
12. Strauss, J., Katabi, D., Kaashoek, F.: A measurement study of available bandwidth estimation tools. In: IMC'03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement, New York, NY, USA (2003) 39–44
13. Nandi, A., Ngan, T.W., Singh, A., Druschel, P., Wallach, D.S.: Scrivener: Providing incentives in cooperative content distribution systems. In: Middleware. (2005) 270–291