Improving Throughput and Node Proximity of P2P Live Video Streaming through Overlay Adaptation*

Bartosz Biskupski, Raymond Cunningham, and René Meier
Trinity College Dublin
{biskupski,racunnin,rmeier}@cs.tcd.ie

Abstract

Due to the heterogeneity of the environment, in which hosts may have different bandwidth capacities and network distances between hosts vary, current mesh-based multicast protocols for video streaming over the Internet tend to inefficiently utilise the available bandwidth and often transfer large amounts of data between distant hosts. This limits system throughput, which results in reduced video quality, and imposes significant costs on Internet Service Providers (ISPs) caused by network traffic outside a provider's own network. This paper presents MeshTV, a mesh-based peer-to-peer (p2p) multicast protocol for streaming live video from a transmitter to numerous viewers. MeshTV proposes an algorithm for adapting the mesh overlay in which nodes explore their possible neighbour nodes and select neighbours so that data throughput is optimised and data is transmitted between nearby (low-latency) nodes, typically within the same ISP thus reducing the costs to ISPs. Our evaluation demonstrates that the adaptation algorithm used in MeshTV can improve video streaming throughput by over 100% and typically reduces the distances (network latencies) between interacting nodes by 50% compared to unoptimised mesh overlays.

*Presented at International Symposium on Multimedia (ISM 2007). ©IEEE 2007, Proceedings available at http://doi.ieeecomputersociety.org/10.1109/ISM.2007.4412380
This work was partly funded by the "Information Society Technology" Programme of the Commission of the European Union under research contract IST-507953 (DBE) and by Enterprise Ireland under the Commercialisation Proof of Concept Programme (MeshTV).

1 Introduction

The high bandwidth requirements of live video streaming greatly limit the number of viewers that can be served by a centralised server (which we call a transmitter) using unicast transmissions. An efficient infrastructural solution to this problem is IP Multicast, which allows users that are topologically close to each other to share the data transmitted from the server. However, this technique, despite being available for several years, suffers from a number of weaknesses that has slowed its adoption by Internet Service Providers (ISPs). Some of the weaknesses of IP Multicast include: a lack of scalability as routers are required to maintain state for each multicast session, resulting in an unacceptably high overhead at Internet backbone routers that were designed to be stateless, security vulnerabilities that may lead to networks being flooded by malicious users, changes at the infrastructure level that typically require significant investment from ISPs and backbone carriers/providers.

As a result, application-level peer-to-peer (p2p) approaches have been proposed that address some of the issues associated with IP Multicast. Peer-to-peer overlays offer a promising approach to streaming live video over the Internet without any specific support from the network and where the total bandwidth available to stream content scales with demand. The feasibility and appeal of peer-to-peer video streaming has been proven by systems such as PPLive [2] or CoolStreaming [17] that already attract over half a million simultaneous viewers for certain events [2].

A common approach to p2p live streaming is to organise nodes into a tree-structured overlay with the root at the data transmitter [6]. The tree structure defines the routing decisions – a node receives data from its parent and forwards it to all its children. This approach, however, has a number of drawbacks. Firstly, it does not utilise the outgoing bandwidth of a large fraction of nodes, the leaves in the tree. Secondly, the received bandwidth at a viewer is limited by the minimum bandwidth on the path from the transmitter to that viewer and any loss in the upper level of the tree reduces

bandwidth available to nodes lower in the tree. Finally, tree structures offer poor resilience to churn as the departure of an internal node in the tree can result in the data stream being lost at all its descendants until the tree is fixed. Multiple tree approaches [5] aim at solving the first two problems, but do not address the third problem as they require more tree structures to be maintained.

Recent research has started to focus on building multicast protocols using overlay meshes to overcome the aforementioned problems (e.g., [7, 11, 14, 13, 17] outlined in Section 5). The approach of these systems is that nodes organise into a random unstructured p2p overlay, which is called a mesh overlay, by selecting multiple random nodes as neighbours to exchange data with. In order to enable data exchange between neighbours, the disseminated data is split into smaller data chunks. Each time a node receives a data chunk, it informs its neighbours and they can request this missing chunk. This improves resilience to churn and allows the upload bandwidth of all nodes to be utilised. With a sufficient number of neighbours, failure of a node does not affect throughput of neighbouring nodes because these nodes can choose to download data from other neighbours.

We show that in heterogeneous environments, where nodes have various bandwidth capacities and network distances between communicating nodes may be different, a random mesh overlay is inefficient. Firstly, video streaming over a random mesh underutilises the available bandwidth, resulting in reduced throughput and suboptimal quality of transmitted video. Secondly, p2p systems in which nodes send large quantities of data to distant nodes, outside of their own ISP network, generate significant costs to ISPs and may result in congestion in the core of the Internet. For this reason, some ISPs consider limiting the usage of p2p applications by either blocking them completely or throttling them. Thus, it is important that p2p systems consider network proximity between communicating nodes and, where possible, transfer most of the data between nearby nodes.

This paper presents MeshTV, a mesh-based peer-to-peer (p2p) multicast protocol for streaming live video from a transmitter to numerous viewers. MeshTV proposes an algorithm for adapting the mesh overlay in which nodes explore their possible neighbour nodes and select neighbours so that data throughput is optimised and data is transmitted between nearby (low-latency) nodes, typically within the same ISP thus reducing the costs to ISPs. The contributions of this paper are twofold:

- we analyse the problem of p2p mesh-based streaming in heterogeneous environments
- we devise and evaluate a practical exploration algorithm that improves data throughput and reduces network distances between interacting nodes in the overlay

In Section 2, we introduce our general approach to mesh-based overlay streaming. Section 3 highlights issues that arise in heterogeneous environments and proposes an exploration algorithm that addresses these issues. In Section 4, we present results of our evaluation of the proposed exploration algorithm. Section 5 outlines recent related work. In Section 6, we conclude this paper and outline future work.

2 Mesh-Based Streaming

The mesh-based approach to data streaming originates from research on gossip/epidemic protocols [8], where nodes periodically exchange information among each other, which results in the eventual dissemination of all information to all nodes. The BitTorrent [7] 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 that 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 quality degradation.

A mesh overlay is created in a random fashion by joining nodes connecting with randomly selected nodes. Neighbouring nodes maintain local knowledge about data chunks they possess by informing each other whenever they receive a new chunk. A node requests missing chunks from neighbours in a random order, in an attempt to acquire different chunks than its other neighbours and thus be able to upload to them. A node needs to keep track of what chunks it has requested from every neighbour to avoid requesting the same chunk from multiple neighbours. A pipelining technique is used where a node can issue a request for a new chunk without waiting for a previous request to the

same neighbour to be satisfied. This helps to fill the download pipe to a certain neighbour and eliminates the request-response delay. The number of pipelined (outstanding) requests from a single neighbour, however, is limited to ensure that requests are spread out over all neighbours. The ability to concurrently upload/download data chunks from many neighbours (called swarming) is one of the advantages of mesh-based systems. Swarming enables the incoming bandwidth of a node to be fully utilised as the outgoing bandwidth of many neighbours can be used. Moreover, it improves resilience to congestion as more chunks will automatically be requested from other uploaders if a certain uploader becomes congested.

3 Exploration Algorithm for Mesh-Based Streaming

The general mesh-based approach to p2p streaming presented in the previous section achieves excellent bandwidth utilisation (and can support high data stream rates) in homogeneous environments, where all nodes have the same upload and download capacities and the network distances between all pairs of nodes are the same [14]. However, this is not a realistic scenario as today's Internet consists of heterogeneous nodes with asymmetric upload and download capacities and various distances (in terms of latency or hops) between each other [16]. In order to perform efficiently in such a heterogeneous environment, a random mesh overlay needs to be adapted to satisfy the following conditions:

- The overlay needs to guarantee that all nodes are able to receive data at a desired rate. A randomly formed mesh overlay may result in some nodes having neighbours with low upload bandwidth only, and hence, such nodes might be unable to receive the desired data rate.
- The overlay needs to support efficient utilisation of the upload bandwidth of all nodes. The data rate received by a node is limited by the global video streaming rate and the upload rate of its neighbours. A node cannot upload data to a single neighbour faster than it downloads this data from other neighbours. Thus, in order to utilise its upload bandwidth, it needs to upload to more neighbours. The number of neighbours that a node uploads to should be correlated with its upload bandwidth. In contrast, a random mesh overlay assigns, on average, the same number of neighbours to all nodes and thus some part of the upload of high capacity nodes is unused.
- Nearby nodes should be preferred over distant nodes when selecting neighbours. This is due to the fact that the cost of sending data to distant nodes is significantly larger for ISPs and is generally undesirable in the Internet core where it may cause congestion. Data transfers within a network of a single ISP helps to significantly lower costs.

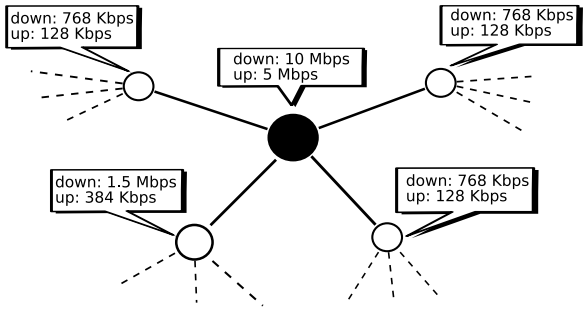


Figure 1. Random mesh overlay example.

Figure 1 shows an example of the inefficiencies in random mesh overlays. The high capacity node (black node) has four neighbours (which, in turn, have their own neighbours not shown in the figure) with low upload bandwidth, hence, its cumulative received data rate is low, likely to be below the video streaming rate. In addition, the black node's upload bandwidth is underutilised due to two facts: it cannot upload to a single neighbour faster than it receives data and the cumulative total download of all its neighbours is lower than its upload bandwidth, even if the neighbours were to download exclusively from the black node.

MeshTV uses an exploration algorithm that continuously adapts an overlay mesh to a heterogeneous environment. Each node in MeshTV maintains two sets of neighbours – *receivers*, which are the neighbours to which it uploads data and *senders*, which are the neighbours from which it downloads data. Naturally, a node has another node in its receiver set precisely when the latter has the former in its sender set. Whenever a node receives a new data chunk, it informs all its receivers, which may decide to request that chunk. All communication between nodes, including chunk requests and transfers, is performed using the TCP protocol.

The exploration algorithm is executed by each node independently and its goal is to adapt the node's set of senders to improve the download rate. The algorithm is executed by a node periodically in rounds and ensures that:

- A node has a constant (configurable) number of senders.
- A node replaces the sender from which it receives the worst download rate with a new (exploratory) sender selected randomly from all nodes in the overlay.

The random node selection is performed using a gossip protocol [10] that, in each round, provides a new random

sample of all nodes in the system. The exploration algorithm also requires that a node maintains estimated download rates ($drate_s$) for each sender s in order to determine the worst sender. Each node measures the download rate received from each of its senders between consecutive runs of the exploration algorithm and updates the current estimate using the formula

$$drate_s = \alpha \times newMeasurement_s + (1 - \alpha) \times drate_s$$

where α is a weighting factor that determines the significance of the latest measurement over historical ones. The weighted update is important as it prevents disconnecting a generally well performing sender due to temporary fluctuations in its performance. Such fluctuations may occur when the sender becomes temporarily overloaded by too many new receivers, which selected it as an exploratory sender. Its old receivers take into account the history of its performance and may decide to stay connected with the sender despite poor interim download rates. However, the new receivers replace the sender due to unsatisfying download rates received, thereby restoring its good performance for the old receivers.

The exploration algorithm adapts the mesh overlay so that *(i)* the upload bandwidth of all nodes is efficiently utilised, *(ii)* download rates of nodes are improved and *(iii)* network latency between interacting nodes is reduced. Upload bandwidth is utilised by matching a node's number of receivers with its available upload bandwidth. The reason for this is that a node continues to gain new receivers when it is underloaded and loses some receivers (i.e., receivers replace it with less loaded senders) when it is overloaded.

A node joining the network initially acquires a random set of senders. The exploration algorithm will then continuously attempt to improve the node's download rate by replacing the slowest senders with senders that can potentially provide higher transfer rates, thereby effectively optimising its set of senders.

This approach also decreases the network latency between neighbouring nodes as an effect of the TCP congestion control mechanism. The reason for this is that when multiple connections share a bottleneck link, TCP allocates more bandwidth to connections with lower network round-trip times (RTT) [12]. When a bottleneck occurs at the sender's uplink, more upload bandwidth is allocated to receivers with low latency. Similarly, when a bottleneck occurs at the receiver's downlink, more download bandwidth is allocated to senders with low latency. This causes receivers to replace distant senders (for which TCP allocates least bandwidth) with exploratory senders that are potentially closer.

Category	Downlink	Uplink	Ratio
A	10 Mbps	5 Mbps	15%
B	3 Mbps	1 Mbps	25%
C	1.5 Mbps	384 Kbps	40%
D	784 Kbps	128 Kbps	20%

Table 1. Node bandwidth distribution

Parameter	Value
stream rate	1500 Kbps
number of senders	5
weighting factor α	0.4
exploration round length	5 sec
chunk size	4 KB
pipelined requests	8
sliding window size	30 sec

Table 2. Protocol parameters

4 Evaluation

The evaluation of the improvements in throughput and network proximity achieved by the exploration algorithm presented in this paper requires that a detailed packet-level network simulator is used. Thus, MeshTV has been implemented in ns-2 [1], which provides a quite realistic model of the physical network and the TCP/IP stack (we use TCP New Reno), however, at the cost of reduced scalability, which limits the number of nodes that have been simulated to 500. Our previous experience in evaluating larger overlays in less accurate flow-level simulators lead us to believe that the findings presented here are also valid for larger overlays [4]. The physical network topology created for simulations is a full mesh with bandwidth being limited on the access links (uplinks and downlinks). The node bandwidth distribution has been derived from Gnutella p2p system measurements [16] and nodes have been categorised into 4 groups: A, B, C and D (see Table 1). Network latencies between nodes are selected uniformly at random between 2ms and 300ms. MeshTV parameters used in the experiments are presented in Table 2. The weighting factor α has been selected experimentally and represents a trade-off between a faster adaptation to persistent changes in the sender's upload rate (when α is high) and a better adaptation to momentary oscillations in the sender's upload rate (when α is low). The mesh overlay in the experiments is initially random, formed by nodes selecting random senders.

4.1 Throughput Improvements

This section demonstrates how MeshTV adapts the mesh overlay to improve video streaming throughput by over

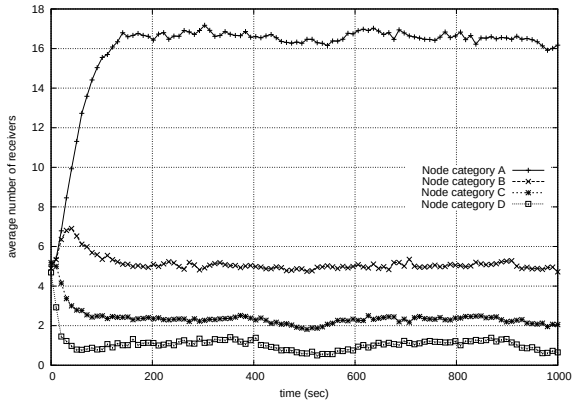


Figure 2. Adapting the number of receivers.

100% compared to unoptimised overlays. We first show how the algorithm improves upload bandwidth utilisation of nodes and then we illustrate how the algorithm results in higher data throughput received by nodes.

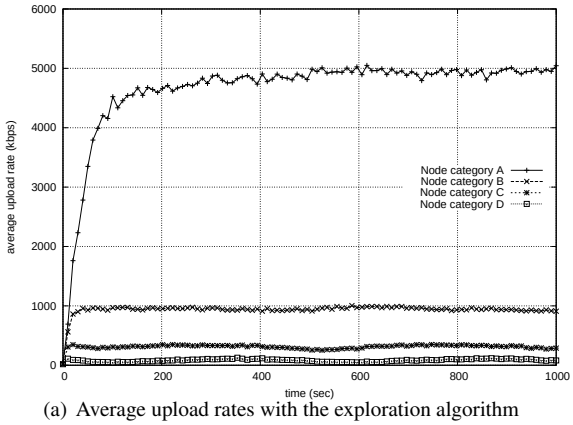
Upload Bandwidth Utilisation

As mentioned earlier, it is important that the algorithm matches the number of receivers of a node with that node's upload bandwidth. Figure 2 shows the average number of receivers for each node category over time. The adaptation process starts with all nodes connected to five random senders, which implies five receivers for each node on average. As the exploration progresses, nodes with high upload bandwidth acquire additional receivers, while nodes with low upload bandwidth reduce their number of receivers.

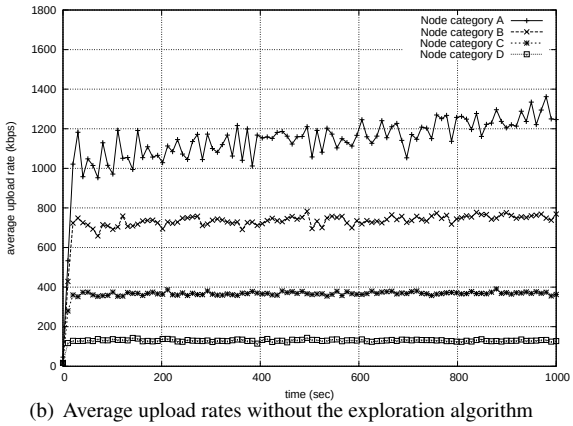
Figure 3 compares the average upload utilisation of the overlay with and without the adaptation (note the different scales on the y-axes). It shows that when the exploration algorithm is used, a node's upload reaches its maximum upload capacity as shown in Table 1. This means that nodes in all categories fully utilise their upload bandwidth. In contrast, when the exploration is not used, the upload bandwidth of nodes in the highest categories A and B is greatly underutilised. It can be observed from these figures and the given node bandwidth distribution that the total aggregated upload for adapted and not adapted overlays is about 550 Mbps and 260 Mbps respectively. This means that the upload bandwidth utilisation is improved by over 100% when the overlay is adapted by the exploration algorithm.

Data Throughput

The improved utilisation of the upload bandwidth results in nodes increasing their data throughput. Figure 4 compares the data rates with and without the exploration algorithm (note again the different scales on the y-axes). It shows that the data rates received in the adapted overlay are



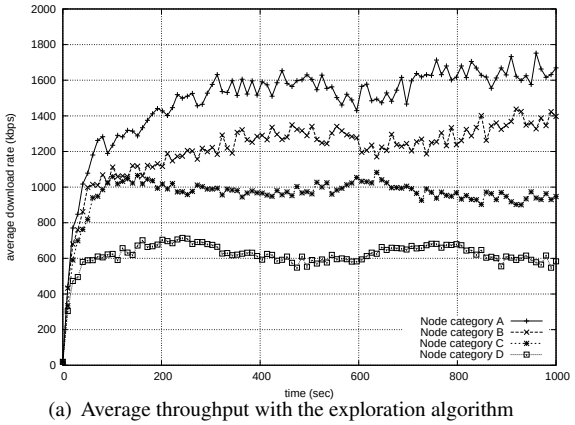
(a) Average upload rates with the exploration algorithm



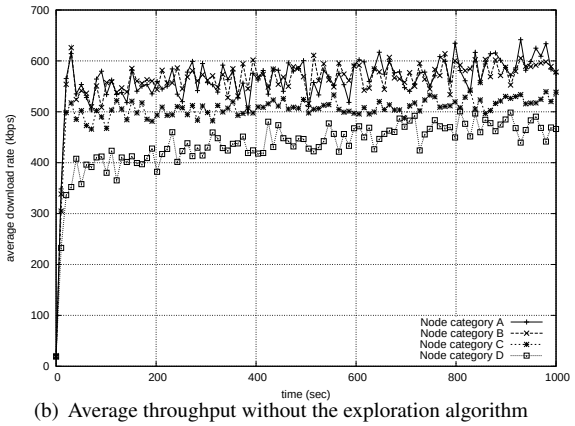
(b) Average upload rates without the exploration algorithm

Figure 3. Optimising average upload rates.

all much higher than in the case of a random mesh overlay. However, not only the upload bandwidth of senders, but also a node's own download capacity limits the received data rate. So, for instance, the download rate of nodes in category D is limited by their download bandwidth of 768 Kbps. Other node categories have higher download capacity and thus achieve higher download rates. MeshTV accommodates limited download bandwidth of some nodes and different data rates received by different node categories through the use of the Multiple Description Coding (MDC) technique and specifically MDC-FEC [9]. The MDC technique enables the original video stream to be split into a number of descriptions. A node can download any subset of all descriptions to recreate the video stream, however, the video quality corresponds to the number of successfully received descriptions. Thus, nodes with lower download bandwidth or nodes whose sender sets have not been adapted yet, download only a subset of descriptions and consequently receive video at a degraded quality. In turn, nodes with high download bandwidth (and adapted sets of senders) download a larger number of descriptions and can receive the same stream at a higher video quality. How-



(a) Average throughput with the exploration algorithm



(b) Average throughput without the exploration algorithm

Figure 4. Improving the average throughput.

ever, the optimal chunk selection mechanism for MDC is still an ongoing research and thus MDC has not been used in the experiments. Here we mention only that when MDC-FEC is used, all descriptions have equal importance for a node and thus even senders downloading few descriptions can provide the node with useful chunks that improve its received video quality. Senders that do not contribute sufficient number of useful chunks exhibit low transfer rates and are replaced with new exploratory senders by the exploration algorithm.

Data Throughput Under Churn

The previous experiments assumed that all nodes join the overlay at the beginning and leave at the end of an experiment. In this section, we show how the data throughput in the overlay is affected by node membership churn, i.e., node arrivals and departures. The methodology for simulating churn is similar to [15]. The overlay is initially random, consisting of 500 nodes, and the node churn starts after 200 seconds. Node departures follow a Poisson process and thus are uncorrelated with each other. A new node starts each

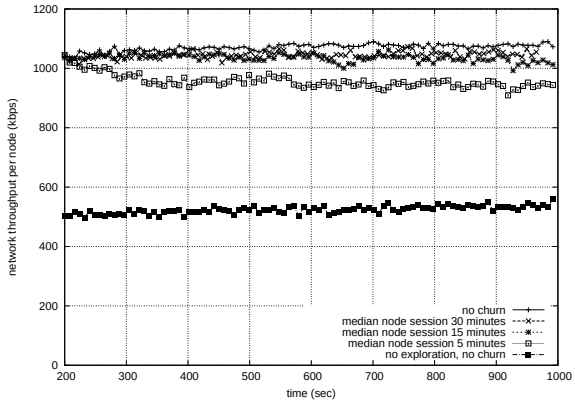


Figure 5. Improving throughput under churn.

time a node leaves the overlay, thus, maintaining a constant size of the overlay. Churn rates in p2p networks are often represented by a median session duration of a node (t_{med}). The rate of node arrivals can be calculated using the fact that inter-arrival times in a Poisson process with an arrival rate λ are exponentially distributed with a mean $\frac{1}{\lambda}$ and a median $\frac{ln2}{\lambda}$. It follows that an arrival rate of a single node is $\frac{ln2}{t_{med}}$, whereas the rate of any arrival in an overlay with N nodes is $N * \frac{ln2}{t_{med}}$. Therefore, a new node arrives (and one departs) on average every $\frac{t_{med}}{N*ln2}$ seconds and these times are exponentially distributed.

In our simulations we use median session durations of 30, 15 and 5 minutes, for which the mean inter-arrival times in an overlay with 500 nodes are 5.2, 2.6 and 0.9 seconds respectively. These settings generate node churn rates significantly higher than have been observed in real p2p systems such as Gnutella and Napster, for which the measured median session duration is approximately 60 minutes [16].

Figure 5 compares the average network throughput per node for different median session durations as well as in cases when no churn is present or the exploration algorithm is not used. The average network throughput is calculated as a sum of all data received divided by the number of nodes. The impact of churn on the data throughput results from the fact that when a node leaves the overlay, all its receivers need to discover new senders. The departure of a high capacity node has the highest impact on the data throughput as such a node maintains many receivers. However, Figure 5 shows that for median session durations of 30 and 15 minutes, the throughput does not degrade by more than 7%. Even excessive churn, where each node's median session duration is 5 minutes, reduces the throughput by only 12%. For the purpose of comparison, Figure 5 also shows that when no churn is present and the exploration algorithm is not used, throughput is reduced by approximately 50%.

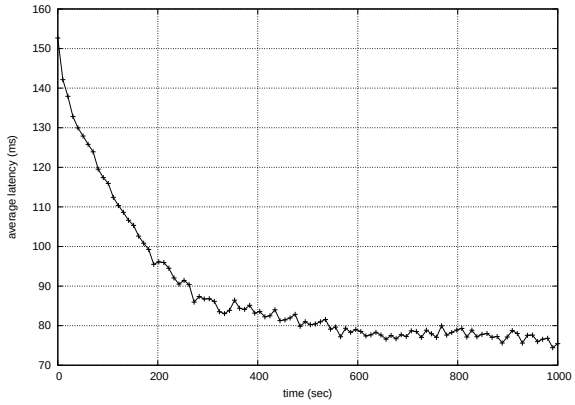


Figure 6. Improving proximity of neighbours.

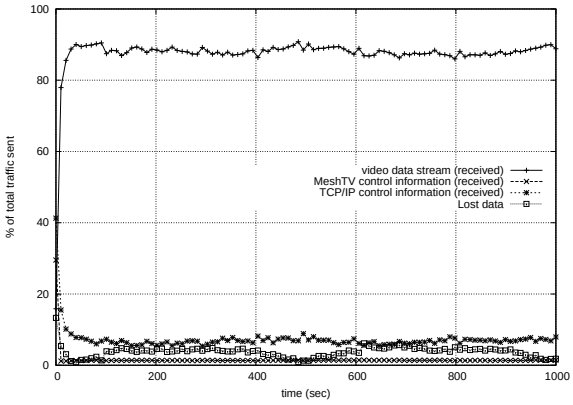


Figure 7. Breakdown of network traffic.

4.2 Node Proximity

Figure 6 shows how the exploration algorithm reduces the network latency between interacting nodes. Initially, the random mesh overlay has an average latency between neighbouring nodes roughly equal to 151ms as the latencies are assigned randomly between 2ms and 300ms. Since senders allocate more upload bandwidth to closer receivers, the overlay adapts, resulting in a reduction of the average latency to about 75ms, which is a 50% improvement. The exploration algorithm does not further reduce the distances between neighbouring nodes as this might degrade the data throughput. Connecting exclusively to the nearest senders implies two undesired effects. Nodes that share low-latency links with many other nodes might be overloaded and the overlay might be divided into disconnected clusters of nearby nodes. The exploration algorithm prevents these unwanted effects as it improves proximity only when this does not degrade the data throughput. This is because a high-latency underloaded node will provide higher data throughput than a low-latency overloaded node and thus will be preferred as a sender.

4.3 Control Overhead

Figure 7 illustrates the ratio of the overall control and data traffic received and lost as a function of the traffic sent over the course of the experiment. Almost 90% of all traffic sent constitutes the video data stream that is successfully delivered to nodes. MeshTV control information, which include notifications from senders about new chunks available, chunk requests and handshakes when two nodes become neighbours, represent only 1.5% of the total traffic. About 6% of traffic represents TCP/IP control overhead, which consists mainly of TCP/IP packet headers. Between 1% and 4% of data traffic is lost on node access links (uplink and downlink) due to congestion and is resent by TCP.

These results indicate that TCP/IP and MeshTV control overhead have moderate impact on the performance of mesh-based p2p streaming. MeshTV can utilise up to 90% of the overall network traffic for transmitting the video data.

5 Related Work

SplitStream [5] is a multiple tree-based multicast protocol built on top of a Distributed Hash Table (DHT). In SplitStream, the original data stream is divided into several substreams, each having an individual distribution tree. In an effort to utilise outgoing bandwidth of all nodes and improve the data throughput, trees are built such that each node is an interior node in exactly one tree. However, SplitStream has poor adaptation to node failures compared to mesh-based protocols. Moreover, it has been shown that multicast protocols built on DHTs suffer from a mismatch between the DHT identifier space and the node outgoing bandwidth, resulting in limited system throughput [3].

Bullet [11] splits the stream into chunks and uses a single tree on top of a mesh. Nodes receive a subset of chunks from their parents in the tree, while the remaining chunks are recovered from nodes in the mesh overlay. This has been shown to improve bandwidth utilisation compared to single tree approaches. Authors also noticed the need for adaptation to node bandwidth. Thus, nodes in the tree compute the best rate to send data to each of their children, while nodes in the Bullet mesh connect to the most useful neighbours. However, Bullet wastes bandwidth on receiving duplicate packets and its reliability to node failures is lower than in pure mesh-based protocols because of the requirement of maintaining the tree structure.

BitTorrent [7] is a file-sharing protocol that uses optimistic unchoking as an exploration mechanism. This is similar to the way nodes in MeshTV explore their poten-

tial senders, however, in BitTorrent exploration serves the purpose of ensuring fairness of bandwidth exchanges rather than optimising the data throughput. Nodes in BitTorrent upload to a constant number of neighbours and thereby are unable to fully utilise their upload bandwidth, whereas nodes in MeshTV continuously adapt their number of receivers to their available upload bandwidth.

Chainsaw [14] is a simple mesh-based p2p streaming protocol, which distributes a stream through a randomly formed mesh overlay. In this paper we have shown that random mesh overlays perform inefficiently in heterogeneous environments as they limit the data throughput and do not exploit the network proximity.

In previous research we proposed a MeshCast protocol [4], which extends Chainsaw with a gossip-based mesh overlay adaptation algorithm to exploit heterogeneity in the environment. However, MeshCast requires that nodes discover their own upload bandwidth, assumes that congestion may occur only at the ISPs and does not exploit node proximity. In contrast, our new MeshTV protocol addresses all these issues.

PRIME [13] is a p2p streaming protocol based on similar observations as MeshCast in terms of throughput optimisation. It suffers from the same issues as MeshCast and additionally relies on a centralised bootstrap server to enable a node to adapt its neighbours.

CoolStreaming/DONet [17] uses a form of exploration to help nodes identify better neighbours (called partners). However, since it limits the number of neighbours that a node can have, the upload bandwidth of high capacity nodes is not efficiently utilised as we show in this paper.

6 Conclusions and Future Work

In this paper we analysed mesh-based p2p live video streaming and identified a number of properties that the overlay needs to exhibit in order to perform efficiently in heterogeneous environments. We presented the MeshTV protocol that employs an exploration algorithm for adapting the mesh overlay. The algorithm improves upload bandwidth utilisation of all nodes by approximately 100% compared to unoptimised mesh overlays. This results in much higher data throughput and consequently enables a broadcaster to offer viewers a higher quality video, even in the presence of high node churn rates. The algorithm also reduces network distances of large volume data transfers by approximately 50%, which significantly reduces costs to ISPs and relieves the core of the Internet from the burden of forwarding this data. Finally, we demonstrated that both TCP/IP and MeshTV control information impose moderate overhead on p2p streaming. Our ongoing work focuses on chunk selection algorithms for p2p mesh-based streaming with multiple description coding.

References

[1] The network simulator - ns-2. http://www.isi.edu/nsnam/ns.

[2] PPLive website. http://www.pplive.com.

[3] A. R. Bharambe, S. G. Rao, V. N. Padmanabhan, S. Seshan, and H. Zhang. The impact of heterogeneous bandwidth constraints on DHT-based multicast protocols. In *IPTPS*, pages 115–126, 2005.

[4] B. Biskupski, R. Cunningham, J. Dowling, and R. Meier. 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*, pages 4–11, New York, NY, USA, 2006. ACM Press.

[5] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: High-bandwidth multicast in cooperative environments. In *SOSP'03: Proceedings of the nineteenth ACM Symposium on Operating Systems Principles*, pages 298–313, New York, NY, USA, 2003.

[6] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron. Scribe: A large-scale and decentralized publish-subscribe infrastructure. *IEEE Journal on Selected Areas in Communications (JSAC)*, 20(8):1489–1499, October 2002.

[7] B. Cohen. Incentives build robustness in BitTorrent. In *the 1st Workshop on Economics of Peer-to-Peer Systems*, Berkeley, CA, USA, June 2003.

[8] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic algorithms for replicated database maintenance. In *PODC '87: Proceedings of the sixth annual ACM Symposium on Principles of Distributed Computing*, pages 1–12, New York, NY, USA, 1987. ACM Press.

[9] V. K. Goyal. Multiple description coding: Compression meets the network. *IEEE Signal Processing Magazine*, 18(5):74–93, September 2001.

[10] M. Jelasity, R. Guerraoui, A.-M. Kermarrec, and M. van Steen. The peer sampling service: experimental evaluation of unstructured gossip-based implementations. In *Middleware '04: Proceedings of the 5th ACM/IFIP/USENIX international conference on Middleware*, pages 79–98, New York, NY, USA, 2004. Springer-Verlag New York, Inc.

[11] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat. Bullet: High bandwidth data dissemination using an overlay mesh. In *Proceedings of the nineteenth ACM Symposium on Operating Systems Principles*, pages 282–297, 2003.

[12] T. V. Lakshman and U. Madhow. The performance of TCP/IP for networks with high bandwidth-delay products and random loss. *IEEE/ACM Trans. Netw.*, 5(3):336–350, 1997.

[13] N. Magharei and R. Rejaie. PRIME: Peer-to-peer receiver-driven mesh-based streaming. In *26th Annual IEEE Conference on Computer Communications IEEE INFOCOM 2007*, 2007.

[14] V. S. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, and A. E. Mohr. Chainsaw: Eliminating trees from overlay multicast. In *IPTPS*, pages 127–140, 2005.

[15] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a DHT. In *Proceedings of the 2004 USENIX Annual Technical Conference*, pages 127–140. USENIX, June 2004.

[16] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking*, 2002.

[17] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum. CoolStreaming/DONet: A data-driven overlay network for live media streaming. In *Proceedings of IEEE INFOCOM'05*, 2005.