

Real-time Video Delivery using Peer-to-Peer Bartering Networks and Multiple Description Coding

J. A. Pouwelse J. R. Taal R. L. Lagendijk D. H. J. Epema
H. J. Sips

Delft University of Technology,
Faculty of Electrical Engineering, Mathematics and Computer Science,
Delft, The Netherlands

{j.a.pouwelse,j.r.taal}@ewi.tudelft.nl*

Abstract – *Broadband Internet access using ADSL or cable modems provides sufficient bandwidth for real-time video streaming. If television channels could be distributed on the Internet, each channel would get a world wide audience. However, television distribution from a single server does not scale and IP-level multicasting is complex and costly. We propose a solution based on application-level multicasting using a P2P network, called P2P-TV. Each peer receives a video stream and must also forward this stream to others. This paper presents an architecture for streaming video. Our P2P-TV proposal aims to solve three problems that are currently not addressed in other P2P streaming proposals, namely 1) maximizing the usage of all available peer bandwidth, 2) taking current network conditions into account (network awareness), and 3) the Freeriding problem. Multiple Description Coding is an integral part of our solution. By splitting the video stream into smaller streams we can utilize all the bandwidth of a peer. Multiple streams allow more efficient adaptation of the multicast tree to current network conditions. Bittorrent-like bartering of content is also enabled by using multiple streams.*

Keywords: peer-to-peer, television, multicasting, freeriding, multiple description coding, network awareness.

1 Introduction

Video streaming over the Internet is becoming more popular with news sites and broadcasting organizations offering complete TV-programs on their websites. However, the visual quality of these streams is often not very good, especially the frame rate is not competitive with real TV. For efficient streaming to many clients, multicast is a promising solution to handle the workload. In this paper we focus on Application-Level Multicast (ALM) because of its flexibility and since it does not require infra-structural changes like IP-level multicast. Our P2P-TV system tightly integrates an ALM / P2P network layer with rate-scalable Multiple Description Coding (MDC).

Three problems exist that need to be solved before ALM of video streams can be widely deployed. The first problem is enabling many clients with widely different and possibly changing bandwidths to display good-quality video without annoying artifacts. Fault-tolerance is an important concern in the design of any streaming system, since it will have to deal with packet losses, failing links, and failing nodes. The second problem is how to construct an efficient ALM tree which uses the underlying resources efficiently. To solve this problem we propose to measure the properties of the resources in real-time. P2P-TV tracks the latencies, bandwidths, and packet losses on the links between P2P nodes. With this information, we can modify the multicast tree to fully exploit the current network conditions. The third problem is freeriding. Freeriding occurs when peers do not donate resources to the P2P network. This is a major threat to P2P systems. By using bartering techniques we make unfair use of the P2P-TV system not worthwhile.

With the advent of P2P networks (Kazaa, eMule, Bittorrent, etc), large “overlay” networks have been created with thousands to millions of nodes. These overlay networks potentially offer the infrastructure to employ widespread ALM for video distribution. IP-level multicast was promised to be the solution for real-time video broadcasting with many clients. It turned out that IP-level multicast is very complex and ISPs and cable companies hesitate to support IP multicasting. In some sense, ALM is simpler than IP multicast. First, because it does not involve changes in the protocols (IP) and infrastructure. Secondly, the application (software) layer may have more information about the topology and connectivity, so that it is better able to solve the multicast problem in a dynamic way [1].

Streaming of video puts more constraints on the network than simple file downloading. Once a video stream has started, data should arrive at a continuous rate in order to guarantee proper display of the video. Since the TCP protocol offers guaranteed delivery but no guaranteed latency and certainly no constant bandwidth, video streaming is mostly using the UDP protocol. Any video streaming system should be able to deal with packet losses, out-of-time delivery, and

*0-7803-8566-7/04/\$20.00 ©2004 IEEE.

bandwidth fluctuations. With video streaming, when some packets are lost, the received data are still useful and the missing data will only cause a glitch or an image impairment.

In order to serve a video stream to a multitude of clients with possibly different bandwidths, the video content should be in such a form that several bandwidths are supported. MDC is an encoding technique for multimedia that generates several streams (or *descriptions*) of data, without any priority attached to each stream. Any combination of the streams is sufficient to decode and display the video, but the more descriptions are received, the higher the quality will be. MDC offers the required resilience to packet losses and is scalable in bandwidth by nature. This feature does not come without cost, the needed bit rate will increase when more error resilience is required. In that sense MDC can be compared with forward error correction, although in general MDC is more efficient in protecting video data.

In a P2P network there are link failures, node failures, premature leaving, congestion, changing topology of the base network etc. In any of these cases the system should not fail, only a small degradation of the delivered stream is allowed. By using a P2P network where all peers exchange data amongst each other and form ALM trees, all available bandwidth in network is exploited. In our architecture, nodes with a high reliability (uptime) are identified and are moved upwards in the ALM tree. The crucial concept is that each node must forward as many bytes of compressed video data as it has received. Malicious nodes that only consume bandwidth and refuse to cooperate and do not forward video content in the ALM tree, are automatically banned. The P2P-TV system architecture is specifically designed to make such malicious behavior unproductive.

In Section 2 we give a description of three common problems in P2P streaming networks. In Section 3 we first give an overview of the architecture of our P2P-TV system and then describe our solutions in depth. We conclude the paper in Section 4.

2 Problem Description

Numerous articles have been published on real-time video streaming. However, as of 2004, no fully functional system has been built which is suitable for widespread usage and most systems are designed in an ad-hoc manner.

We believe that the following three problems have not yet been sufficiently addressed, and need to be solved before P2P-TV-like systems can become superior to traditional television broadcasting. This paper presents a detailed overview of these problems and approaches to solve them. We summarize the problems as follows:

1. video – network rate matching,
2. network awareness,
3. freeriding.

Rate matching — In an ideal P2P streaming system, maximum video quality is offered to a peer and the video stream bit rate exactly matches the available network bandwidth. This

implies that a video source is encoded specifically for each peer and instantly follows network bandwidth variations. Because such a system is difficult to scale, the video layer often only supports just a few rates, from which all clients are served. We define *rate matching* as the capability of a streaming system to match for each client the video bit rate with the network bandwidth. Generally two solutions exist: layered coding and multiple description coding (MDC). Both offer several quality levels at different bitrates. Layered coding is however not really fault-tolerant. MDC is inherently fault tolerant, since each description is sufficient to display a reasonable quality video. Layered coding is mostly used in single server solutions, the client selects the number of layers that corresponds to the bandwidth to the server. For instance WaveVideo [2] is based on 2D wavelet encoding and offers different quality layers.

In Splitstream [3] the authors assume an odd/even frame splitting scheme as an MDC method. They only use the fault tolerance but not the scalability capabilities of MDC.

Network awareness — The ALM-tree is the heart of our P2P-TV system, but how to construct efficient ALM trees is still a poorly understood problem. Many researchers propose ad-hoc solutions and do not motivate their choices. The ALM tree should utilize the underlying resources efficiently and meet constraints set by the application, such as latency. Many network properties should be taken into account, when building an ALM tree. An efficient ALM tree is also highly dynamic due to changes in network conditions and because peers can join and leave at any moment. To date, most systems take at most one or two metrics into account. Often this is the latency between peers.

In Section 3, we propose to build an ALM tree by taking into account relevant network properties such as peer uptime, network bandwidth, network latency, and packet loss ratio. We define *network awareness* as the degree to which a P2P system takes into account the properties of the underlying resources.

In [4] an overview is presented of 17 software packages to measure numerous node and link properties, including IP aliases, packet reordering, and link utilization. For P2P-TV-like systems, latency is often the only network property that is measured. An exception is [1] which proposes to use both latency and a mechanism for balancing the workload across different physical links. However, this paper does not include a method to discover the underlying physical links.

A method to measure the topology of the underlying network is outlined in [5]. They developed RocketFuel that uses traceroute to determine the topology of a network and addresses the problem that at least 30% of the Internet routers have multiple IP numbers. In earlier work we carried out an extensive uptime measurement of P2P file-sharing clients [6]. We observed that this uptime can vary between 3 minutes and 3 months. We expect the same heterogeneity in uptime for P2P-TV peers, but current proposals do not take this into account [7, 8, 3, 9]. Another important network property which is ignored in the P2P video streaming literature is the increasing usage of firewalls and NATs. As of 2004, nearly 60% of Internet connected computers using P2P are fire-walled and

thus cannot be contacted directly [6].

Freeriding — A P2P system only functions properly if the participating nodes are donating resources. However, few people are willing to donate resources to complete strangers. We define *Freeriding* as using resources in a P2P system without donating at least an equal amount of resources. Note that we use a more strict definition of Freeriding than in literature. Simulations show that systems for P2P video streaming may cease to function when it is possible to avoid donating resources [10]. The Freeriding problem is still an open problem in the P2P video streaming context and seen in 2003 as an “interesting area of future work” [3]. The simulation in [10] shows the impact of Freeriding on the likelihood that peers are denied joining an ALM tree (peer rejection probability). For their DirectStream system, the presence of many freeriders (50%) increased the client rejection probability by roughly 40% for some workloads.

All the real-world experience with Freeriding and almost all of the related work comes from the area of P2P file sharing. In 2000, measurements showed that nearly 70% of the people using the Gnutella file sharing system were not donating any resources [11]. Note that such high numbers would increase the peer rejection probability in DirectStream to unacceptable levels. The first prototype that tried to solve the Freeriding problem was Mojonation (`mojonation.net`). Mojonation used micro payments and accounting for all actions and resources in a file sharing system, such as search, download, processor usage, and disk space. The Mojonation prototype was functional, but highly complex and inefficient due to overhead generated by the inclusion of hard cryptography, brokers, price bidding, and credit ratings.

As indicated by millions of users worldwide, Bittorrent is a popular file sharing system. Bittorrent is specifically designed with the Freeriding problem in mind [6]. Bittorrent divides a file into multiple chunks of fixed size to facilitate bartering. The key philosophy behind Bittorrent is that peers should barter for chunks of a file, i.e., upload content at the same time they are downloading it. This bartering diminishes parasitic behavior of users—it is not possible to download without sharing. Each peer is responsible for maximizing its own download rate. Peers do this by downloading from whoever they can and deciding on which peers to upload to, via a variant of tit-for-tat in which a peer responds in one period with the same action that its collaborator used in the previous period. As a result, peers with a high upload rate will probably also be able to download with a high speed. In this way, Bittorrent can achieve a high bandwidth utilization. □

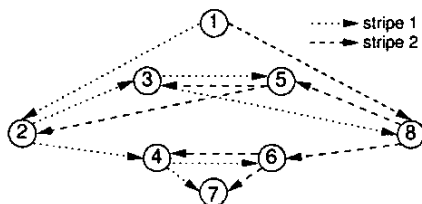


Figure 1: Splitstream multicast layout

We found that none of the existing systems gives satisfactory solutions for all three problems. The system that is closest to our aimed P2P-TV system is Splitstream [3]. Splitstream is the first design that takes an aspect of the underlying network into account. Splitstream uses a P2P system called Pastry which connects nearby peers together in a P2P network [12]. By building Splitstream on top of Pastry, an ALM tree is built that has reduced latency. The aim is to distribute the load of forwarding over all participating nodes. The data is split up in so-called stripes. Each stripe has its own ALM tree, formed in such a way that each node is a forwarding node in only one tree and a non-forwarding node in all the other trees. Besides this load-balancing effect, when one node is failing, only the ALM tree where that node is near the root of the tree is failing badly. An example of splitstream’s ALM tree is shown in Figure 1, the source node (1) produces two stripes. The ALM tree for stripe 1 is split at node 2 to nodes 3 and 4 and then to nodes 5,6,7 and 8. The ALM tree for stripe 2 splits at node 8 to nodes 5 and 6 and then to nodes 2,3,4 and 7.

In P2Cast [13] each video stream is delivered through a single ALM tree. Nodes that join the tree after the start of the streaming receive a patch containing the data they have missed. In this system no bartering or load-balancing is used. In P2VoD [9] and in [8], a similar approach is followed where no actual network parameters are used to build the ALM trees.

3 P2P-TV architecture

Our P2P-TV architecture is designed to solve the three problems of Section 2. It consists of a P2P network with bartering and multiple description coding. The design is based on a (modular or) layered approach as shown in Figure 2.

We assume the Network layer is the IP network that can transmit and receive IP packets. The Delft Network Awareness (DNA) layer is described in Section 3.2 and deals with obtaining information about the network conditions. For example, the DNA layer will store IP numbers of other peers, conduct network performance measurements, and register the up-times of peers. With only a few MByte of storage, the DNA layer is able to cache a considerable number of peer IP numbers. The DNA layer does not contact or exchange information with other peers, this is the functionality of the P2P layer.

The P2P layer offers P2P functionality to higher layers, such as discovery of both nodes and content. Upon startup, each peer tries to discover which peers are up and running. The P2P layer contacts the peers cached by the DNA layer to find active peers. Peers recursively contact other active peers to update their list of active peers. We assume that each peer generates a public/private key pair which can be used to identify peers and validate or sign information. The P2P layer is also responsible for maintaining and distributing meta data such as the available television channels and their programs. The P2P layer also contains the functionality to contact the peers in the ALM tree of a television channel.

The Tree layer builds ALM trees on top of the P2P network such that the MDC layer can distribute its separate de-

scriptions over different paths through the P2P network. The Tree layer uses the information from the DNA layer to create an ALM tree which is both efficient and fault tolerant. Peers with a high uptime and sufficient bandwidth should be placed high in the ALM tree. P2P-TV splits the measurement of the underlying resources and the efficient usage of these resources because such a layered architecture is easier to implement and improve. The MDC layer, responsible for real-time MDC-encoding and decoding is described in Section 3.1. The GUI layer, finally, provides the user interface for channel zapping, subtitles, and picture-in-picture etc, but is not further discussed here.

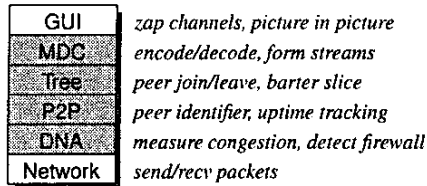


Figure 2: Layered system architecture and functionality

3.1 The MDC layer: rate matching

Multiple Description Coding is an integral part of our solution to combat the three problems of Section 2. MDC offers fault tolerance but also rate scalability for rate matching. Figure 3 show a typical MDC setup. The MDC-encoder encodes the data into multiple different description. Each description possibly follows a different path through the network to a joint destination. The decoder tries to decode whatever descriptions have arrived as good as possible, before displaying.

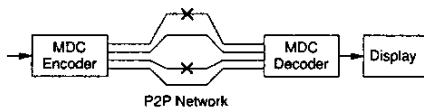


Figure 3: Multiple Description Coding

Without going into much detail of practical methods [14, 15, 16] we present a simple and insightful method for MDC. Suppose X_B is a low quality encoded version of our video stream, the so-called base-layer; the enhancement layer X_E contains information that can be used to improve the quality of the base layer. Description D_1 contains the base layer and half of the enhancement layer $\{X_B, X_{E,1}\}$, and description D_2 also contains the full base layer and the other half of the enhancement layer $\{X_B, X_{E,2}\}$. Upon reception of both descriptions, the base layer is received twice and the enhancement layer once. This means we have introduced redundancy $R(X_B)$, but we have also increased the error resilience of the transmission. If one of the descriptions is lost, we still can reconstruct our video by using the low-quality base layer and half of the enhancement layer. Besides offering the required error resilience, this also introduces rate scalability. Not every description is needed to display the video at a reasonably quality level, the more descriptions are received, the higher

the quality will be. For implementation of video-MDC, the required layers can be obtained by performing sub-band coding or wavelet coding [17, 18]. Other MDC techniques exist that are not based on layered coding, but it is beyond the scope of this paper to discuss them in depth. In [14], description one contains all even frames from a video stream, description 2 contains all odd frames. When one description (one frame) is lost, the surrounding frames are used to interpolate the missing frames. This system applies MDC in the temporal domain, whereas pixel based and sub-band or dct-based methods apply MDC in the spatial domain.

Another method that offers rate scalability is pure layered coding. Layered video coding also offers several streams of data (base layer and enhancement layers) and each layer is transmitted separately. The video is then available at different rates and quality levels. A major drawback of layered coding in P2P networks is that the data is prioritized. The base layer delivery should be guaranteed, whereas the enhancement layers have lower priority and can be discarded if necessary. The P2P network should in this case support prioritized delivery. MDC is more complex than pure layered coding, but more flexible in this respect, since it inherently protects the more important information, without requiring prioritized delivery.

We point out that by using MDC to introduce rate scalability and error resilience we make a tradeoff between complexity, optimal encoding performance, and latency. Instead of serving each client its own stream, clients share the intermediate links, so the available bandwidth is exploited. On the other hand, clients can select what parts and what descriptions of the video are received, such that the requested rate is matched with the available bandwidth. To combat packet losses, we have three options: 1) retransmitting packets, 2) applying erasure codes or FEC, or 3) MDC. End-to-end retransmissions greatly increase the latency of these packets, probably making these packets arrive too late. The second option is only suitable for file downloading, since it does not distinguish between important and unimportant data. MDC is better for video streaming, since it only protects the more important parts of the data, without increasing the latency.

For dealing with the rate matching problem, we introduce a method called partial subscription. We assume that the exact bandwidth between server and client is not known and is changing over time, and – as in any latency-constrained transmission – there will be packet losses due to congestion and late arrival. Furthermore links in the network can be out of order for some longer periods, implicating that the overlay network should adapt accordingly and meanwhile any transmission using that link will fail. By transmitting the video stream over different paths through the network, we greatly increase the robustness to link failures and packet losses. Apart from the fact that clients can scale the received stream by requesting only a subset of all the descriptions, we will add more resolution by introducing partial subscription. Our MDC system consists of a combination of *spatial* and *temporal* MDC¹. First assume the stream consists of basic blocks of M frames,

¹The exact implementation of this system with combined spatial and temporal MDC coding is outside the scope of this paper.

consecutive in time. So, each basic block is a time fragment of the video. Then each basic block is split up using MDC into N descriptions D_n (spatial MDC). Each description is also divided into M time slices $s_{n,m}$, $1 < n < N$, $1 < m < M$ (temporal MDC). To correctly decode the multiple descriptions, at each time instance m at least one slice should be received. By requesting more than one slice at each time instant, the quality of the decoded video will improve. So in total, in each block NM different slices can be obtained.

In our model the client C receives streams D_n from neighboring peers P_n . Assume that the P2P layer already has found suitable peers P_n to deliver the streams D_n where each stream corresponds to one description. On each link (P_n, C) , a bandwidth of B_n is available. Furthermore, we assume that all links share a bottleneck with bandwidth B_{last} , such that $B_n \leq B_{last}$, $1 < n < N$. The client will request from each peer P_n a fraction f_n of the slices $s_{n,1}, s_{n,2}, \dots, s_{n,M}$ of stream D_n . Suppose each stream has a rate $R(D_n)$, the total rate incoming for client C is then $R = \sum_{n=1}^N f_n R(D_n) \leq B_{last}$ such that $f_n R(D_n) \leq B_n$. Figure 4 shows this delivery model.

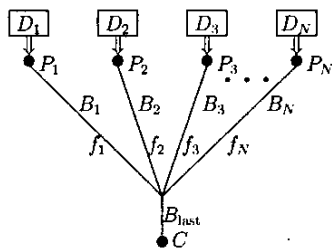


Figure 4: Delivery model. Client C receives stream D_n from peer P_n , $n = 1, \dots, N$

Figure 5 shows an example of a transmission scheme. Suppose $N = 4$, $B_{last} = 1024$, $R(D_n) = 512 \quad \forall n \in \{1, \dots, 4\}$, $B_1 = 512$, $B_2 = 128$, $B_3 = 256$, $B_4 = 512$. In this example the last link is the bottleneck, since the total bandwidth of the individual links is higher than the last link. By setting $f_1 = 3/4$, $f_2 = 1/4$, $f_3 = 2/4$ and $f_4 = 2/4$, the last link is filled completely while each description is received partly. In this example the slices are arranged such that at each time, two slices are received, permitting that one slice can be lost before a high distortion will occur. Also shown are cases where a slice of one stream is lost due to congestion or late arrival. MDC is resilient to these errors, since often another description of that slice has arrived correctly.

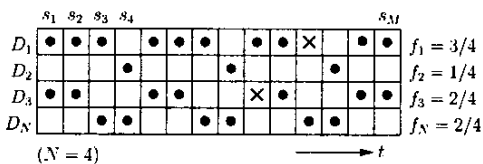


Figure 5: Example of an MDC transmission scheme. Our client receives particular slices s_m from each description D_n . Slices marked with x are lost due to packet loss.

3.2 The DNA layer: network awareness

To efficiently use the underlying resources in a P2P streaming system, the properties of these resources should be known in detail. We present a method to measure in real-time the properties of the resources. We exploit this information to form a network-aware ALM tree. The network properties are measured by the Delft Network Awareness (DNA) layer as shown in Figure 2. This information is shared among all peers by the P2P layer.

The DNA layer adds full network awareness to P2P-TV. DNA gathers information at three levels, namely, the peer level, the network level, and the link level. Two peers that both run the DNA software can exchange network performance numbers and exploit this new information without paying the bandwidth costs of discovering this information. These peers do not need to belong to the same ALM tree.

The following example shows the potential of measuring and sharing such a simple network property as node up-time. Peers that leave their computer locked onto the same television channel for several weeks can significantly boost the reliability of the ALM tree. By caching the IP numbers of high up-time nodes for several ALM trees, we can quickly join an ALM tree for a new television watching session.

The number of peers that have to be contacted for re-joining an ALM tree is below the $O(\log N)$ of Zigzag [8] when we assume that at least one of the peers in cache remains on-line.

The idea behind DNA is that millions of computers running P2P-TV or another P2P application can easily measure in real-time Internet bandwidth, latency, packet loss between *all* core Internet routers and exploit that to, for instance, accurately predict download speeds between two random IP numbers. The software to do such measurements already exists [5, 4]. But making such software an integral part of a P2P system was never proposed before.

Storing and updating this volume of performance information is the task of the P2P layer in Figure 2. The storage is locality based, i.e. the information is stored where it is measured. A good candidate for the actual distribution of this information among the peers is the Gossip [19] protocol. For example, a list with latencies between peers and routers is stored at the peer that measured them and at nearby (in terms of hop count) peers. Figure 6 shows the latencies and router IP numbers of routes between five peers on different continents and a target peer. The figure shows how traffic from a peer in Taiwan is routed through Chicago towards its final destination near Amsterdam. Such information provides good insight into the underlying topology of the Internet with only a few measurements.

To overcome the "IP discovery problem", each peer shares the information about which peers it knows are in its proximity, using a Gossip-style protocol. A complete map of an Internet area can be build by repeatedly querying peers for IP numbers and performance information. Due to the wide spread usage of dynamic address assignment (DHCP), each peer also needs to generate a unique identifier for identifying peers across different sessions.

The bandwidth overhead of network-awareness as imple-

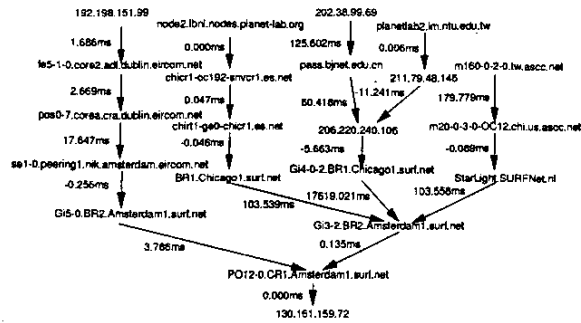


Figure 6: Latency tree from 5 peers to a single peer

mented by the DNA and P2P layer is small, especially when compared to the requirements of real-time television streaming. The gain is the possibility to construct an ALM tree with an near-optimal mapping to the underlying network in terms of efficiency and reliability.

3.3 Anti-Freeriding measures

The rate matching and network awareness parts of our P2P-TV system are specifically designed to enable a solution for the Freeriding problem. Freeriding is discussed last in this paper, but we feel it is the most difficult and most important problem to solve, since it is a threat to any P2P system.

The most basic solution to the Freeriding problem in video streaming is to enforce that each peer forwards a complete video stream [8, 9, 13]. This method does not provide rate scalability and lacks fault tolerance against packet losses. The MDC video splitting technique introduces fault tolerance but also enables bartering of all N video descriptions D_n . A single MDC stream produces only low-quality television, but by bartering MDC streams, the quality can be improved. Users therefore have a natural incentive to barter.

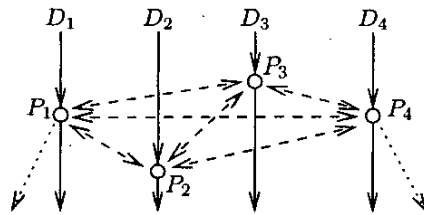


Figure 7: Layout of a barter team

A natural way to introduce bartering, is by using “barter teams”. Figure 7 shows four peers P that have formed a barter team and exchange all four MDC video streams. Each of the four peers receives one of the MDC streams ($N = 4$) and sends it to a node outside the barter team, depicted with the solid lines. By sharing each D_i with their team members they obtain all video streams, as depicted by the dashed lines. The spreading factor of a barter team is defined as the number of MDC streams send to peers outside the team minus the number of received MDC streams. The shape of the ALM tree is determined by the spreading factors. When all teams have a

spreading factor of zero, the tree height is maximal. The dotted lines in Figure 7 are only present in teams with a positive spreading factor. An advantage of the barter team method is that it localizes bartering in stable groups. A major disadvantage is that barter teams have no incentive to spread the streams with a positive spreading factor. A more sophisticated solution is therefore needed.

Our solution is twofold. First, with our MDC rate-matching solution, a peer can receive a number of slices $s_{n,m}$ from various descriptions D_n and forward one or more to a range of peers. The maximum number of bartering partners is then increased to $N \times M$. Second, we do not use barter teams, instead a peer can barter with any other node in the ALM tree. This way, joining an ALM tree consists of finding suitable bartering partners. Figure 8 shows five different “growth levels” of a peer along with the number of incoming and outgoing streams. A peer starts as depicted in Figure 8.a and only receives one slice of an MDC stream. In Figure 8.b and c the peer also forwards slices. Figure 8.d and e shows a peer that also receives two slices. Peers can “grow” from Figure 8.a towards Figure 8.e and beyond. This specific feature of P2P-TV results in faster response times. Response time is important when users zap through television channels, users expect an image to appear within a second. Fast response times are difficult to achieve for IP-level multicasting systems or single stream systems. With the “growth” feature of P2P-TV, a low-quality image can be already shown when only a single slice is received.

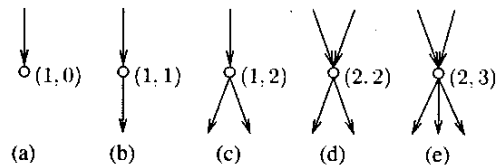


Figure 8: Different growth levels of a peer, annotated with (in-degree, out-degree) of each node

The tits-for-tats scheme prevents freeriding, but requires that peers *directly* donate as much bandwidth as they are consuming. With basic bartering the upload and download of each peer will be equal (Figure 8.b and d). Due to the asymmetry of Internet connections, peers often cannot donate as much bandwidth as they can consume. The tits-for-tats scheme therefore results in inefficient usage of download bandwidth. We call this the “asymmetry problem”. A mechanism of debts and credits between peers could solve this problem. Peers can then obtain a video stream at full capacity (B_{last} in Figure 4) and at a later time repay their debt.

A decentralized debit/credit system is needed, but without the overhead of strong cryptography. We propose to use a light-weight reputation mechanism where freeriders can be identified, loosely based on the ideas from [20, 21] which both required reliable servers. Our idea is based on the usage of “public barter records” where each peer publishes a list with previous bartering partners. Each barter record contains the amount of traffic, the starting time, the duration, and

the network location of the two peers (network part of IP only for privacy). These barter records are digitally signed by the two peers. Since these records are small, and the records are easily verified, the level of participation of each peer can easily be determined. Over time, such a system builds a “web of trust”. Peers that have been bartering for some time are listed in numerous barter records. It is then safe for other peers to increase the time span of a tit-for-tat from real-time bartering to bartering within several hours. Then, peers can alternate between being selfish (8.a) and donating bandwidth (8.c).

4 Conclusion

P2P video streaming can be a feasible solution to real-time video streaming over the Internet. In this paper we have established three important problems in the area of P2P video streaming. We have discussed several papers that gave partial solutions to these problems. We have presented the P2P-TV architecture to integrally solve each of these problems. Multiple Description Coding and partial subscription are used to solve the rate-matching issue and to provide fault tolerance. The DNA layer offers network awareness, so that the system can adapt to changes in network and to failing links and nodes. Finally, the Freeriding problem is solved by bartering chunks of video data amongst clients. By using fine-grained bartering of MDC slices, a natural incentive against Freeriding is created. Combined with a solution for the upload bottleneck, we believe we sufficiently addressed the Freeriding problem for P2P-TV. We see as future work the implementation of some or all of these layers and to conduct experiments to confirm the claims we made in this paper.

References

- [1] Y.-H. Chu, S. G. Rao, and H. Zhang, “A case for end system multicast,” in *ACM SIGMETRICS 2000*, (Santa Clara, CA), pp. 1–12, ACM, June 2000.
- [2] G. Fankhauser, M. Dasen, N. Weiler, B. Plattner, and B. Stiller, “Wavevideo: an integrated approach to adaptive wireless video,” *Mob. Netw. Appl.*, vol. 4, no. 4, pp. 255–271, 1999.
- [3] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, “Splitstream: high-bandwidth multicast in cooperative environments,” in *Proc. of the 19th ACM symposium on Operating systems principles*, pp. 298–313, 2003.
- [4] N. Spring, D. Wetherall, and T. Anderson, “Reverse-engineering the Internet,” in *Proc. of the 2nd Workshop on Hot Topics in Networks*, Nov. 2003.
- [5] N. Spring, R. Mahajan, and D. Wetherall, “Measuring ISP topologies with RocketFuel,” in *Proc. of the 2002 Conf. on applications, technologies, architectures, and protocols for computer communications*, pp. 133–145, 2002.
- [6] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, “A measurement study of the bittorrent peer-to-peer file-sharing system,” Tech. Rep. PDS-2004-007, Delft University of Technology, Apr. 2004.
- [7] A. Nakao, L. Wang, and L. Peterson, “MSB: Media Streaming Booster,” Tech. Rep. TR-666-02, Princeton University, dec 2002.
- [8] D. A. Tran, K. Hua, and T. Do, “A peer-to-peer architecture for media streaming,” *IEEE Journal on Selected Areas in Communications, Special Issue on Advances in Service Overlay Networks*, vol. 22, pp. 121–133, jan 2004.
- [9] T. Do, K. A. Hua, and M. Tantaoui, “P2VoD: Providing fault tolerant video-on-demand streaming in peer-to-peer environment,” in *to appear in the Proc. of the IEEE Int. Conf. on Communications (ICC 2004)*, jun 2004.
- [10] Y. Guo, K. Suh, J. Kurose, and D. Towsley, “A peer-to-peer on-demand streaming service and its performance evaluation,” in *Proc. of 2003 IEEE Int. Conf. on Multimedia & Expo*, vol. 2, (Baltimore), pp. 649 – 652, jul 2003.
- [11] E. Adar and B. A. Huberman, “Free riding on Gnutella,” *First Monday*, vol. 5, Oct. 2000.
- [12] A. I. T. Rowstron and P. Druschel, “Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems,” in *Proc. of the IFIP/ACM Int. Conf. on Distributed Systems Platforms Heidelberg*, pp. 329–350, Springer-Verlag, 2001.
- [13] Y. Guo, K. Suh, J. Kurose, and D. Towsley, “P2Cast: peer-to-peer patching scheme for VoD service,” in *Proc. of the 12th Int. Conf. on World Wide Web*, pp. 301–309, 2003.
- [14] J. Apostolopoulos, T. Wong, W. tian Tan, and S. Wee, “On multiple description streaming with content delivery networks,” in *Proc. IEEE INFOCOM*, July 2002.
- [15] V. A. Vaishampayan, “Design of multiple description scalar quantizers,” *IEEE Trans. on Information Theory*, vol. 39, pp. 821–834, May 1993.
- [16] V. Goyal and J. Kovacevic, “Generalized multiple description coding with correlating transforms,” *IEEE Trans. on Information Theory*, vol. 47, pp. 2199–2224, September 1999.
- [17] A. Reibman, Y. Wang, X. Qiu, Z. Jiang, and K. Chawla, “Transmission of multiple description and layered video over an EGPRS wireless network,” in *Proc. IEEE Int. Conf. on Image Processing*, vol. 2, pp. 136–139, October 2000.
- [18] M. Pereira, M. Antonini, and M. Barlaud, “Channel adapted multiple description coding scheme using wavelet transform,” in *Proc. Int. Conf. on Image Processing*, vol. 2, (New York), pp. 197–200, September 2002.
- [19] P. T. Eugster and R. Guerraoui, “Probabilistic multicast,” in *Proc. of the Int. Conf. on Dependable Systems and Networks*, pp. 313–324, IEEE Computer Society, June 2002.
- [20] M. Gupta, P. Judge, and M. Ammar, “A reputation system for peer-to-peer networks,” in *Proc. of the 13th Int. workshop on Network and Operating Systems support for digital audio and video*, pp. 144–152, 2003.
- [21] T.-W. Ngan, D. S. Wallach, and P. Druschel, “Enforcing fair sharing of peer-to-peer resources,” in *2nd Int. Workshop on Peer-to-Peer Systems (IPTPS)*, Feb. 2003.