Content distribution support in modern wireless and wired networks

Constantinos Vassilakis*

Department of Informatics and Telecommunications
National and Kapodistrian University of Athens
Greece
Email: cvassilakis@noc.uoa.gr

Abstract. This thesis focuses on the distribution of live video streams over Peer-to-Peer (P2P) networks. Our approach to the problem is to preserve the inherent advantage offered by the P2P architecture and exploit it to enhance the quality experienced by the end user, by revealing those subtle processes that may affect the overall system performance and require a different design perspective that suits the new distribution environment. In the first part of this thesis we examine the impact of the adopted playout policy on the performance of P2P live streaming systems. We argue and demonstrate that (popular) playout policies which permit the divergence of the playout points of different nodes can deteriorate drastically the performance of P2P live streaming. Consequently, we argue in favor of keeping different playout points "near-in-time", even if this requires sacrificing (dropping) some late frames. In the second part of this thesis we deal with the natural instability of the distribution environment which poses a major problem in the case of a video streaming service. Taking into account the proven correlation between the experienced quality at each node and node churn we manage to improve the stability (lower node churn) and the offered quality of the system.

1 Introduction

Distributing a live video stream using a P2P streaming system has the advantage over a point-to-point client/server system of offering more resources to clients by effectively turning each one of them into a secondary server that assists in the distribution of the stream. These additional resources can yield improved scalability and/or resilience, depending on the design of the system.
Background: Several application-layer multicast systems have been proposed for addressing the low deployment of network-layer multicast. Initial application-layer multicast systems mimicked network-layer multicast and thus adopted a single tree topology [2–4] aiming at providing a similar performance to it with respect to *stress* and *stretch* (the stress metric is defined per-link and counts the number of identical packets sent by a protocol over each underlying link in the

* Dissertation Advisor: Ioannis Stavrakakis, Professor

network, the stretch metric is defined per pair of nodes and captures the ratio of path-length over the overlay to the corresponding path-length over unicast IP).

The first wave of improvements to these systems aimed at addressing the unreliability of end-nodes and at decreasing the control overhead. The Bullet system [5] proposed splitting a stream into multiple blocks and delivering disjoint subsets of these blocks to different nodes over an overlay tree. It then let the nodes search for "missing blocks" and download them from other nodes using additional mesh links. The Zigzag system [6] aimed at reducing the control overhead by first clustering peers and then building a multicast tree on top the formed clusters.

Additional improvements aimed at balancing the forwarding load and leveraging bandwidth heterogeneity. SplitStream [7] splits a stream into multiple stripes and sends each one over a different multicast tree. Load balancing in terms of forwarding is achieved by making a node internal into one tree and leaf in all the others. CoopNet [8] proposed using multiple-description-coding [1] and transmitting each description layer over a different tree so as to allow heterogeneous peers to join trees according to their bandwidth capacity. ChunkySpread [9] adopted ideas from SplitStream but used an unstructured approach for building the trees whereas SplitStream is based on the Pastry DHT [10].

Proposals like CoolStreaming [11] and PRIME [12] have abandoned trees in favor of BitTorrent-like [13] transmission based on swarming on top of a mesh topology. In these systems the stream is broken into different blocks and nodes obtain such blocks from multiple senders. "Buffer maps" are used for advertising the availability of blocks at each node.

P2P streaming using tree or mesh overlays: Organizing the nodes into a tree shoots for scalability by requiring only n overlay links, where n denotes the number of receivers. Having a minimal number of overlay links reduces the *stress* of the underlying physical links, i.e., the number of times that the same information can flow over the same physical link (this can happen as multiple overlay links may go through the same physical link). It also minimizes the amount of overlay link monitoring overhead for detecting congestion/churn and triggering handoffs in-time to avoid disruption of playout (assuming that the same amount of monitoring expenditure is paid at each link). These observations hold true for both single-tree/no-coding and multiple-tree/multiple-description-coding architectures.

Meshes on the other hand shoot for resilience to congestion/churn by providing each node with multiple parents from which it can receive the stream in parallel (using single-, multiple-description-, or network-coding). Since a mesh uses more than n overlay links, it can increase the stress of the underlying physical links and the monitoring overhead (although the latter is not necessarily true as a node can take advantage of the redundancy offered by having multiple parents and performing a more lazy monitoring of each incoming links). The discussion of which combination of topology and encoding is the "right-one"

[1] A layered coding scheme in which each layer/description/substream is independently decodable and full stream quality amounts to obtaining all the layers.

has been going-on for some time, and although it seems that recent mesh-based systems using coding have several advantages [1], it all depends in the end on the assumed operating environment and the desired cost/complexity of building and maintaining the system: well-behaving environments (e.g., dedicated cable networks) can benefit from the simplicity/economy offered by tree-based distribution; uncontrolled/variable environments (e.g., under-provisioned parts of the current Internet) can benefit from the redundancy offered by mesh topologies and coding.

Our work: The majority of works focus on overlay construction and coding neglecting the fact that functions well understood in the context of point-to-point video streaming require to be revisited in the context of P2P video streaming since the new setting adds new dimensions beyond our previous understanding. Our approach to the problem is to preserve the inherent advantage offered by the P2P architecture and exploit it to enhance the quality experienced by the end user, by revealing those subtle processes that may affect the overall system performance and require a different design perspective that suits the new distribution environment.

In the first part of this thesis we have examined the impact of the adopted playout policy on the performance of P2P live streaming systems. We argue and demonstrate experimentally that (popular) playout policies which permit the divergence of the playout points of different nodes can deteriorate drastically the performance of P2P live streaming. Consequently, we argue in favor of keeping different playout points "near-in-time", even if this requires sacrificing (dropping) some late frames that could otherwise be rendered (assuming no strict bidirectional interactivity requirements are in place). Such nearly synchronized playout policies create "positive correlation" with respect to the available frames at different playout buffers. Therefore, they increase the number of upstream relay nodes from which a node can pull frames and thus boost the playout quality of both single-parent (tree) and multiple-parent (mesh) systems. On the contrary, diverging playout points reduce the number of upstream parents that can offer a gapless relay of the stream. This is clearly undesirable and should be avoided as it contradicts the fundamental philosophy of P2P systems which is to supplement an original service point with as many additional ones presented by the very own users of the service. In section 2 we provide more details on the impact of playout scheduling on the performance of P2P streaming and summarize our findings.

In the second part of this thesis we have taken into consideration the fact that the quality experienced at a peer is highly correlated with the probability of this peer to churn. A novel churn model is introduced which associates the likelihood of churn with the experienced quality. The model produces a lifetime distribution that is in agreement with recent measurement studies. Considering such a model in a P2P environment with node churn, we explore peer selection strategies aiming at improving the stability (reducing node churn) and the offered quality of the system, by taking into consideration the distinct characteristics of a live video streaming service. In section 3 we provide a more extended overview on the problem as well as the main outcomes of our work. Finally in section 4 we conclude this article.

2 On the Impact of Playout Scheduling on the Performance of Peer-to-Peer Live Streaming

The (new) role of playout scheduling in P2P: The playout scheduler is the component of a video receiver that handles the buffering and rendering of received frames. Designing appropriate playout schedulers for video streaming application was one of the central research topics of the multimedia transmission community up to the emergence of P2P streaming systems, at which point the focus shifted onto overlay construction and coding issues. Although fairly well understood in the context of point-to-point video streaming [14], playout scheduling has received a rather limited attention in the context of P2P video streaming [15–17]. The new setting, however, perplexes playout scheduling beyond our previous understanding. In addition to achieving the desired tradeoff between interactivity and stream continuity, the playout scheduler must now jointly factor-in that different playout processes become coupled in the context of P2P; buffering, rendering, or dropping a frame affects not only the local process but also downstream ones that might connect and request frames from the local node. We argue that although seemingly subtle compared to topology construction and coding, playout scheduling still deserves some attention as a bad choice with respect to it can impact quite negatively the performance of P2P streaming systems, despite the existence of the other two powerful enablers.
Contribution: Locating upstream parents and content delivery path maintenance are recognized as two major challenges in the area of P2P streaming [18]. In this work we argue and demonstrate that the adoption of a playout policy and the level of synchronization it may achieve among different peers, has an impact on the performance since it directly affects the availability of upstream parents.

We are aware of only two works directly related to ours. In [19] the authors state that for gapless playout, peer selection should not only be done based on network quality criteria, but also on the buffer status of the candidate parent peer, effectively recognizing the phenomenon of "negative correlation" between buffer contents. However they do not associate this phenomenon with the level of synchronization resulting from different playout schedulers, which is the main contribution of our work. In [20] different receivers achieve different synchronization levels with the source as a result of the initial prefetching mechanism. In order to improve a peer's "liveness", the playout rate is slightly altered while a parent and a client peer may switch roles if the selected parent is behind in playback to facilitate catch up of the late peer. The connection between the synchronization of different receivers and their ability to cooperate by serving missing frames (what we call "availability" in the following) is not explored.

We consider a *delay preserving* playout policy called *Sync* and a *data preserving* one called *Async* in the context of a P2P streaming system. These two policies lay at the extremes of the spectrum of studied playout policies [14]. Under Sync, the playout scheduler enforces a fixed predefined time offset between the time that a frame is presented at a receiver and the time it was captured at the encoder. To do so, it has to drop "late" frames that arrive after their scheduled playout time, even if they are eventually received correctly and in their entirety. The data preserving Async policy on the other hand, imposes an initial buffering delay and then presents frames by draining the buffer at a constant rate. In the event of a buffer underflow, the playout freezes and resumes again upon the reception of the next frame. Not dropping late frames makes the offset between encoding and decoding times variable. In fact, in the absence of losses in the network, the offset increases with each underflow by an amount equal to the duration of the underflow.

We operate each one of these playout policies in a P2P streaming system with the following characteristics: (1) hierarchical structure, (2) threshold-based handoffs (change of upstream parent) based on partial or full information on the remaining network, (3) single-description coding. Such a setting resembles initial P2P streaming systems as the one presented in [21] and was chosen due to the popularity of such systems, their simplicity, and most importantly, in order to protect our evaluation of playout from issues that are orthogonal to it. In a sense, our chosen setting is the most fragile one as it includes a minimum amount of redundancy. Certainly one can design an over-provisioned system based on a dense overlay graph with multiple reception points and elaborate coding, but this would obscure the effects of playout policy which is what we want to isolate in this work.

We develop a simulation environment for the above policies and setting and use it to compare them across different levels of network load and heterogeneity with respect to link capacities. Our evaluation is based on "direct" metrics such as *Discontinuity*, which captures the percentage of time a user spents viewing some frozen frame, and *Loss*, which captures the percentage of content never presented to a user (both metrics are defined precisely later). To explain the observed results on these metrics we introduce a new "indirect" one – called *Availability* – which roughly amounts to the number of available upstream parents to which a node can perform a smooth handoff at a time of poor reception quality from its current parent. Based on several simulation scenarios for our control variables (load, heterogeneity, information on remote nodes, number of past frames kept) we arrive at the following main observations and conclusions:

1. Sync performs consistently better than Async with respect to both Discontinuity and Loss under a wide spectrum of load and heterogeneity. The improved performance can be explained by the fact that Sync maintains higher Availability and thus is able to perform smooth handoffs at times of poor reception. Under Async, the underflows contribute to the time divergence of playout points and the de-correlation of buffer contents. Thus when a node seeks a handoff it becomes difficult to find a parent with the missing frames for a smooth transition.

2. Sync is effective even under limited knowledge of remote nodes (used for performing handoffs). Having the playout nodes nearly synchronized means that any one of them can offer more or less the missing frames, so we don't need to have a global view of buffer contents – tracking a small set of alternative parents suffices for handoff operations. Async on the other hand needs to know the buffer contents of remote nodes so as to identify the one (if any) whose playout point is at the right distance for a gapless handoff.
3. Similarly, Sync is relatively immune to constraints on the number of downstream nodes that a parent can support. Having the playout points of different nodes near in time creates a natural load-balancing with respect to the handoffs because all nodes hold approximately the same frames and are equally good from the standpoint of a of a node seeking for a new parent. Contrary to this, in Async there are many cases where few nodes exist that are at the "correct" time distance from many other nodes, but cannot accommodate all of them due to these constraints and thus the seeking nodes are forced to perform handoffs that induce gaps in playout.
4. Although rather counter intuitive, Async's performance is favored by randomness in parent selection (imposed by restrictions such as the ones described above) since the latter eventually assists in keeping playout points "near-in-time"; peers are forced not to diverge a lot by performing handoffs that induce loss and thus restore, up to a point, their offset.
5. Unlike Sync, Async can benefit from keeping frames in the buffer even after they have been displayed locally. This, however, leads to several known complications (how much of it is needed to smooth out the disruption without making the offset exceedingly large) as well as some new ones (copyright restrictions permit the nodes of P2P streaming systems to buffer only a limited time window of copyright protected material [22].

All the above indicate that Sync is a better option for the considered P2P systems. At the core of its advantage is that it is conforming to the P2P character of the application. Async on the other hand, by virtue of the divergence that it fosters, goes against the P2P paradigm by effectively reducing the number of secondary service points that are available to a node.

3 Minimizing Node Churn in Peer-to-Peer Streaming

Node churn: While the main advantage of P2P technology is the high availability of resources, the main drawback is that the distribution network is formed by highly transient peers who join and leave the system (churn) at their own will. This natural instability poses a major problem, especially in the case of P2P streaming where there are strict timing requirements for the the delivery of content and an efficient and a stable efficient connection to the service is a highly desirable feature.

Organizing the nodes into a tree shoots for scalability while meshes on the other hand shoot for resilience to congestion/churn. Mesh-like distribution attempts to conceal the environment's instability from the end-user by providing several concurrent connections between peers (multiple-parent systems) providing this way higher reliability compared to tree-like distribution (single-parent systems). However, even if these systems ideally manage to always retain a peer's connectivity to the service, they do not always manage to effectively retain service quality.

Lifespan-based protocols: Several approaches address the problem of this given instability of the environment providing organizational protocols aiming at avoiding the effects of node churn to the performance. In [23], [24] the authors argue and demonstrate that taking into consideration the expected session times of peers (their lifespans) can yield systems with performance characteristics more resilient to the natural instability of their environments. Through active probing of over half-a-million peers in a widely-deployed P2P file sharing system, the authors determined that the session times of peers can be well modeled by a Pareto distribution. In this context, the implication is that the expected remaining session time of a peer is directly proportional to the sessions current length, i.e. the peers age. This observation forms the basis for the introduction of a new lifespan-based approach for organizational protocols.

In [25] the authors attempt to minimize node churn in a closed group of peers by appropriately selecting the members of the group. Similarly in [26] initially stable nodes are distinguished from others and then an architecture of two levels with stable and unstable nodes is created in order to improve system's performance. A peer's age is also exploited in these works to infer the stability of a peer.

Node churn in P2P streaming: None of the presented works has considered of dealing apart from the effects of churn with the causes of node churn attempting to minimize the phenomenon itself. Recent measurement studies have revealed that node churn in P2P streaming is fundamentally different from node churn in P2P file sharing. Contrary to the user behavior exhibited in a file sharing service, participating users in a streaming service are impatient and terminate their participation into the service either due to loss of interest or due to low observed performance [21, 27].

Thus, it proves that partially node churn not only affects performance but it is also a function of performance in terms of service quality. The existence of this twofold relation between churn and service quality opens a new perspective to the problem. Node churn in P2P streaming is a phenomenon that exists independently of the service quality but grows with the degradation of the latter. We argue that more elaborate organizational protocols are required for a P2P streaming service heading for connection stability and efficiency at the same time.

Contribution: In this work we motivate towards this direction by exhibiting the performance advantages of such an approach. We introduce a churn model that attempts to capture the twofold relation between churn and quality without yielding for its accuracy but mainly for its ability to produce this kind of correlation. However this model produces a Weibull lifetime distribution which reaches an agreement with most recent measurement results in P2P streaming systems [28]. This approach is quite different from that followed in several performance evaluations since peer lifetimes are not considered known "a priori" but they are the result of the delivered service quality.

On this basis we exhibit that peer selection strategies shooting either for efficiency or stability produce good performance only in specific areas of operation, thus, we propose a new robust peer selection strategy which balances efficiency and stability requirements to give performance gains in the entire spectrum of the underlying conditions [29].

4 Conclusions

Our approach to the problem of the distribution of live video streams over P2P networks is to preserve the inherent advantage offered by the P2P architecture and exploit it to enhance the quality experienced by the end user, by revealing those subtle processes that may affect the overall system performance and require a different design perspective that suits the new distribution environment. The main results of our work follow:

1. The adoption of a synchronized playout policy in a P2P live streaming system results in "positive correlation" of buffer contents among peers increasing this way the number of upstream relay nodes from which a node can pull frames and thus boost the playout quality of P2P streaming systems. The inherent advantage offered by the P2P architecture is preserved and can be exploited effectively on the benefit of playout quality.
2. Node churn in P2P streaming is fundamentally different from node churn in P2P file sharing. Delivered quality is a crucial parameter affecting a peer's decision to churn. Peer selection strategies should seriously consider this twofold relationship and aim at creating overlay connections that ensure at the same time content availability, connection efficiency and stability. Furthermore, considering selection and optimization in short time scales is more appropriate for a P2P streaming service.

References

1. Magharei, N., Rejaie, R., Guo, Y.: Mesh or multiple-tree: A comparative study of live p2p streaming approaches. In: INFOCOM 2007. (Anchorage, Alaska,6-12 May 2007)
2. Deshpande, H., Bawa, M., Garcia-Molina, H.: Streaming live media over peer-to-peer network. Stanford University, Technical Report (2001)
3. hua Chu, Y., Rao, S.G., Seshan, S., Zhang, H.: A case for end system multicast. IEEE Journal on Selected Areas in Communication (JSAC), Special Issue on Networking Support for Multicast **20(8)** (2002)
4. Banerjee, S., Bhattacharjee, B., Kommareddy, C.: Scalable application layer multicast. In: ACM Sigcomm 2002. (Pittsburgh, Pennsylvania, August 2002)
5. Kostic, D., Rodriguez, A., Albrecht, J., Vahdat, A.: Bullet: high bandwidth data dissemination using an overlay mesh. In: ACM SOSP '03. (New York, USA, Oct. 2003)

6. Tran, D.A., Hua, K.A., Do, T.T.: A peer-to-peer architecture for media streaming. IEEE Journal on Selected Areas in Communication (JSAC) **22(1)** (January 2004)
7. Castro, M., Druschel, P., Kermarrec, A.M., Nandi, A., Rowstron, A., Singh, A.: Splitstream: High-bandwidth multicast in a cooperative environment. In: ACM SOSP '03. (New York, USA, Oct. 2003)
8. Padmanabhan, V.N., Wang, H.J., Chou, P.A., Sripanidkulchai, K.: Distributing streaming media content using cooperative networking. In: ACM NOSSDAV 2002. (Miami Beach, FL, USA, May 2002)
9. Venkataraman, V., Yoshida, K., Francis, P.: Chunkyspread: Heterogeneous unstructured end system multicast. In: ICNP 2006. (Santa Barbara, California, 12-15 November 2006)
10. Rowstron, A., Druschel, P.: Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In: IFIP/ACM International Conference on Distributed Systems Platforms (Middleware). (Heidelberg, Germany, November 2001) pp. 329–350
11. Zhang, X., Liu, J., Li, B., Yum, T.S.P.: Donet/coolstreaming: A data-driven overlay network for peer-to-peer live media streaming. Volume 3 of Proc. INFOCOM 2005. (Miami, FL, USA, 13-17 March 2005) pp. 2102–2111
12. Magharei, N., Rejaie, R.: Prime: Peer-to-peer receiver-driven mesh-based streaming. In: INFOCOM 2007. (Anchorage, Alaska,6-12 May 2007)
13. Bittorent. (http://www.bittorrent.org)
14. Laoutaris, N., Stavrakakis, I.: Intrastream synchronization for continuous media streams: A survey of playout schedulers. IEEE Network Magazine **16(3)** (May 2002)
15. Vassilakis, C., Laoutaris, N., Stavrakakis, I.: On the benefits of synchronized playout in peer-to-peer streaming. In: CoNEXT '06: Proceedings of the 2006 ACM CoNEXT conference. (Lisboa, Portugal, 2006) 1–2
16. Vassilakis, C., Laoutaris, N., Stavrakakis, I.: The impact of playout policy on the performance of p2p live streaming...or how not to kill your p2p advantage. In: 15th Annual SPIE/ACM Multimedia Computing and Networking (MMCN '08). (San Jose, California, January 2008)
17. Vassilakis, C., Laoutaris, N., Stavrakakis, I.: The impact of playout scheduling on the performance of peer-to-peer live streaming. Elsevier Computer Networks (Special Issue on Content Distribution Infrastructures for Community Networks) **vol. 53** (issue 4. Mar. 2009) 456–469
18. Yiu, W.P.K., Jin, X., Chan, S.H.G.: Challenges and approaches in large-scale p2p media streaming. IEEE MultiMedia **14(2)** (Apr-Jun 2007) pp. 50–59
19. Yeh, C.C., Pui, L.S.: On the frame forwarding in peer-to-peer multimedia streaming. In: Workshop on Advances in Peer-to-Peer Multimedia Streaming (In conjunction with ACM Multimedia 2005). (11 Nov. 2005, Hilton,Singapore)
20. Jiang, H., Jin, S.: Nsync: Network synchronization for peer-to-peer streaming overlay construction. In: ACM NOSSDAV '06. (Newport, Rhode Island, May 2006)
21. Rao, S.: Establishing the viability of end system multicast using a systems approach to protocol design. Carnegie Mellon University, Phd Thesis, Technical Report CMU-CS-04-168 (Oct. 2004)
22. Hayes, D.L.: Advanced copyright Issues on the internet. Fenwick and West LLP (2007)
23. Bustamante, F.E., Qiao, Y.: Friendships that last: peer lifespan and its role in p2p protocols. In: Web content caching and distribution: proceedings of the 8th international workshop, Norwell, MA, USA, Kluwer Academic Publishers (2004) 233–246

24. Bustamante, F.E., Qiao, Y.: Designing less-structured p2p systems for the expected high churn. IEEE/ACM Trans. Netw. **16**(3) (2008) 617–627
25. Godfrey, P.B., Shenker, S., Stoica, I.: Minimizing churn in distributed systems. In: SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications, New York, NY, USA, ACM (2006) 147–158
26. Wang, F., Liu, J., Xiong, Y.: Stable peers: Existence, importance, and application in peer-to-peer live video streaming. In: Proc. of INFOCOM 2008. (2008) 1364–1372
27. Vu, L., Gupta, I., Liang, J., Nahrstedt, K.: Measurement and modeling a large-scale overlay for multimedia streaming. In: International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine 2007). (Vancouver, Canada, August 2007)
28. Silverston, T., Fourmaux, O.: Measuring p2p iptv systems. In: ACM Network and Operating Systems Support for Digital Audio and Video (ACM NOSSDAV'07), June 2007
29. Vassilakis, C., Stavrakakis, I.: Minimizing node churn in peer-to-peer streaming. Under submission (2008)