Pay-to-Play: An Incentive Mechanism for Chaining Protocols for Video Streaming

J.-F. Pâris, A. Nagar, A. Amer and T. Schwarz, S. J.

Abstract--**We present a voucher-based incentive mechanism for peer-to-peer systems distributing videos on demand using a chaining protocol. In our proposal, clients collect vouchers from the downstream clients to which they forward data, and use these vouchers to reward the upstream clients that send them data. Because these vouchers are signed with the public key of the server, they circulate among the clients without any server intervention. We illustrate the operation of our pay-to-play mechanism with chaining protocols for video-on-demand and discuss how it affects the server's workload.**

Keywords— **P2P, video-on-demand, chaining.**

I. Introduction

Video constitutes now 40 percent of consumer Internet traffic and is expected to reach 63 percent by the end of 2015 [C10]. This traffic is currently supported by a very expensive infrastructure comprising huge server farms, high-bandwidth Internet connections and extensive content delivery networks. The enormous size of this infrastructure translates into equally enormous power consumption. This situation is alarming because these power requirements will continue to increase as long as video traffic continues to grow.

Peer-to peer (P2P) technology avoids this predicament by letting clients share the burden of distributing video data. Most of the work now done by the server infrastructure becomes delegated to clients that will require little additional power to forward the video data they are receiving to their "peers." As a result, P2P solutions eliminate the need for large, capital-intensive, power-hungry server farms. Since each client is a potential co-worker, together they can handle very large and sudden surges of demand, such as those caused by flash crowds. In addition, P2P solutions do not require any special support from the network, be it IP multicast or any specific content distribution infrastructure.

Adapting P2P technology to VoD is not a trivial task as VoD presents some important differences from other P2P applications, such as file sharing. First, extant file sharing systems do not account for the real-time needs of streaming applications. As they do not download video data in sequence, these data remain unusable until the download is complete [XH+02, VIF06]. Second, these real-time needs mean that we will have largely unidirectional data transfers among peers rather than data exchanges: peers that are already watching the video will forward their video data to more recently arrived peers without receiving any video data from them. We need incentive policies that motivate peers to forward to their successors the video data they have received from their predecessors. At the same time, these policies should have the lowest possible overhead.

We propose here a voucher-based mechanism that requires clients to "pay" for the video data they receive from other clients and lets them collect "payments" whenever they forward video data to other clients. All vouchers used in the payments are generated, numbered and signed by the server. While our mechanism is specifically tailored for chaining protocols, it would apply to all P2P solutions where each peer receives all its data from a single predecessor peer.

The remainder of the paper is organized as follows. Section II reviews previous work on chaining protocols and on incentive mechanisms for video streaming. Section III introduces our pay-to-play mechanism and Section IV discusses how it would interact with the various chaining protocols. Finally Section V has our conclusions.

II. Previous Work

For brevity, we will focus our discussion on chaining protocols for video-on-demand and on incentive mechanisms for streaming protocols. Readers interested in a survey of P2P video streaming systems are referred to the work of Liu *et al.* [LGL08].

A. *Chaining protocols*

Standard chaining [SHT97] constructs chains of clients such that (a) the first client in the chain receives all its data from

Jehan-François Pâris, University of Houston, Houston, TX, USA, jparis@uh.edu
Anurag Nagar, University of Houston, Houston, TX, USA, andynagar@yahoo.com
Ahmed Amer, Santa Clara University, Santa Clara, CA, USA, a.amer@acm.org
Thomas Schwarz, S. J., Universidad Católica del Uruguay, Montevideo, Uruguay, tschwarz@ucu.edu.uy

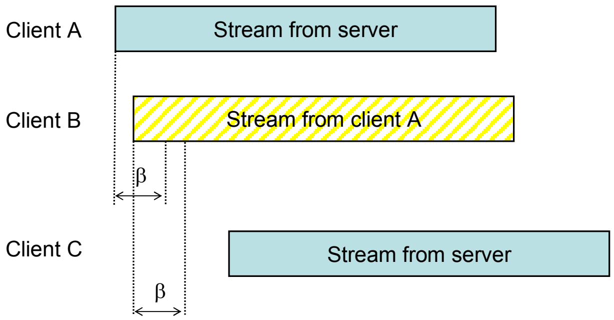

Fig. 1. How chaining works.

the server and (b) subsequent clients in the chain receive all their data from their immediate predecessor. As a result, video data are "pipelined" through the clients belonging to the same chain. Because chaining does not require clients to have very large data buffers, a new chain has to be restarted whenever the time interval between two successive clients exceeds the capacity β of the buffer of the previous client. Figure 1 shows three sample client requests. Since client A is the first customer, it will get all its data from the server. As client B arrives less than β minutes after client A, it can receive all its data from customer A. Finally client C arrives more than β minutes after customer B and must be serviced directly by the server.

The main weakness of standard chaining is its poor performance at low arrival rates, more precisely, whenever the average time interval between two consecutive requests exceeds β minutes. Several variants of the chaining protocol address that issue.

Advanced chaining [LZ+08] proposes to bridge this gap by inserting every β minutes idle peers that will relay the data. *Optimal chaining* [SH+02, SH+05] addresses the same issue by managing all client buffers as a single shared resource. As a result, clients can "borrow" the buffers of other clients in order to bridge gaps between incoming requests. The protocol can also integrate streaming proxies in order to increase chain responsiveness and resiliency.

Expanded chaining, also known as the *cooperative* video distribution protocol [P05], takes advantage of the larger buffer sizes of modern clients. It assumes that clients:

1. Have a buffer large enough to store the entire content of the video they are playing.
2. Have enough upstream bandwidth to forward video data at the video consumption rate.
3. Will stop forwarding data to their successors in the chain as soon as they have finished playing the video.

Consider now a pair of consecutive clients that are separated by a time interval Δt. When the second client arrives, the first one remains available for $D - \Delta t$ additional time units. As seen in Figure 2, we have to consider two cases:

1. If $\Delta t < D$, the incoming client will receive the first $(D - \Delta t)$ minutes of the video from the previous client and its last Δt minutes directly from the server.

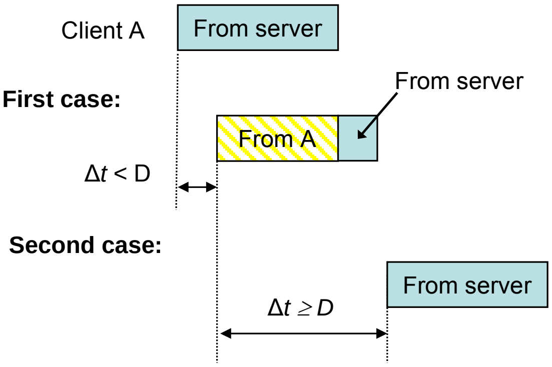

Fig. 2. How expanded chaining works.

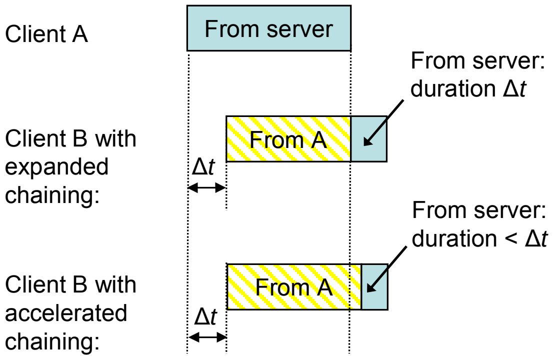

Fig. 3. How accelerated chaining compares with expanded chaining.

2. If $\Delta t \geq$ D, there is no overlap between the two requests; the server will then initiate a new transmission of the video.

Accelerated chaining [PS10, PAL11] improves upon expanded chaining by requiring clients to forward video data to their successor in the chain at a slightly higher rate than the video consumption rate, say, between one and ten percent faster. Since clients can now receive more data from their predecessors, they will need less data from the server. Fig.3 illustrates this concept. With expanded chaining, client B received the first $(D - \Delta t)$ minutes of the video from client A and the missing Δt minutes from the server. With accelerated chaining, client B will receive more than $(D - \Delta t)$ video minutes from client A and fewer minutes from the server.

More formally, let b denote the video consumption rate and $b_a > b$ the accelerated video forwarding rate. We define the forwarding acceleration factor f of the video as

$$f = b_a / b \, .$$

For convenience of notation, we define $\rho = 1/f$. Forwarding a video of duration D at the accelerated video forwarding rate b_a will take ρD time units. Conversely, during time T, a client can obtain video data to be displayed in fT time units. We consider now a pair of consecutive clients that are separated by a time interval Δt. When the second client starts up, the first one remains available for an additional $D - \Delta t$ time interval. Hence, the second client will

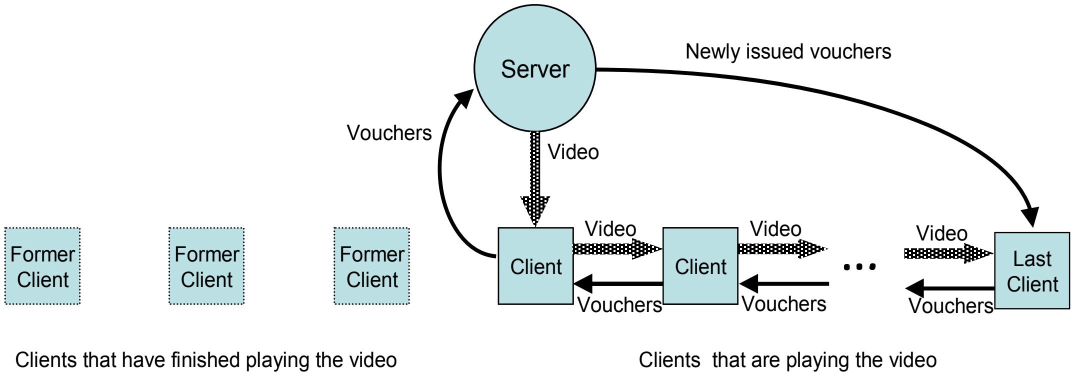

Fig. 4. An illustration of our pay-to-play incentive mechanism. Observe that vouchers and video data always move in opposite directions.

receive the entire video from the first client as long as $\rho D \leq D - \Delta t$. This condition is equivalent to

$$\Delta t \leq (1-\rho)D \, , \qquad (1)$$

or

$$f \geq \frac{D}{D-\Delta t} \, , \qquad (2)$$

Defining the threshold value $\Delta t^* = (1-\rho)D$, we can describe how accelerated chaining operates in a more detailed fashion:

1. If $\Delta t \leq \Delta t^*$, there is a sufficient overlap between the current request and the previous request to allow the second client to get all its video data from the first client.
2. If $\Delta t^* < \Delta t < D$, the second client will receive the first $f(D-\Delta t)$ minutes of the video from the first client and its last $D - f(D-\Delta t)$ minutes directly from the server. This transmission will start at time $t+f(D-\Delta t)$ and end at time $t+D$.
3. If $\Delta t \geq$ D, there is no overlap between the two requests; the server will then initiate a new transmission of the video, starting at time t and ending at time $t+D$.

As a result, the server workload becomes negligible once the request arrival rate produces interarrival times satisfying Equation (1). These savings are significant because the server will still have to manage client arrivals and departures and this workload will increase linearly with the arrival rate of requests.

B. *Incentive mechanisms for video streaming*

Earliest work in this area focused on bartering schemes in which peers exchanged video chunks amongst each other. Such exchanges are analogous to those found in the BitTorrent protocol [C03]. More recently, Mol *et al.* [MP+08] proposed a "Give-to-Get" algorithm that addresses the problem of free-riding by favoring peers who prove to be good forwarders of video content.

A second class of incentive schemes comprises schemes that credit peers for each upload and debiting in case of download. As a result, these schemes can reward multilateral exchanges of video chunks among the clients. Their sole drawback is increasing the server's workload by requiring it to act as a virtual banker. Many such virtual currency models have been proposed for P2P networks such as Micro-Payments [GL +01] or Dandelion [SP+07]. In a recent work, Wang *et al.* [WW+10] proposed a virtual currency model in conjunction with a game-theoretic framework to analyze the peers' optimum resource sharing behavior in VoD systems. In addition, Aperjis *et al.* [AF+08] proposed a multi-lateral, market based system called PACE (Price Assisted Content Exchange) for content distribution using virtual currency.

III. Pay-to-Play

As several authors [GL+01, AF+08, MP+08, WW+10] have observed, tit-for-tat incentive mechanisms are poorly suited to VoD applications. These mechanisms were designed to ensure a fair exchange of data within pairs of peers. Such exchanges are fairly rare in VoD applications because incoming clients have little to offer to their predecessors (and much to ask of them).

Currency-based mechanisms do not have this drawback: they let instead clients "purchase" video data from their predecessors with the currency they collect "selling" their own video data to their successors.

A key issue in the design of any currency-based incentive mechanism is its overhead. This is especially true for mechanisms requiring the server to act as a banker and maintain accounts for all the service clients. This can seriously limit the scalability of the system and its handling of flash crowds. To address this issue, Wang *et al.* proposed a lightweight currency-based mechanism that does not keep track of all video data transfers [WW+10]. In their proposal, only peer*s* that have a complete copy of the video get rewarded when they forward video data to other peers while video data exchanges among these so-called *audience peers* are not recorded. While this approach reduces the

Chain number	Sequence number	Time-stamp	Expiration Time	Data amount

Fig. 5. A voucher.

account management overhead, it does so by eliminating any reward mechanism for the audience peers.

Our approach is quite different. We focus on streaming protocols where each peer receives all its video data from a single peer and forwards them *in real time* to one or more peers. This restriction allows us to eliminate peer accounts and their maintenance overhead.

As shown on Fig. 4, we let the server issue vouchers that can circulate among the clients without any further server intervention. As a result, our *pay-to-play* mechanism limits server interventions to two cases:

1. When a new client *X* arrives, the server finds an existing client *Y* that had no successor, cuts the flow of free vouchers to that client, informs it that it will now have to forward data to the new client *X* and starts a flow of free vouchers to that client. This flow will continue until client *X* has an assigned successor, at which time the server will again redirect the flow of free vouchers to the new successor.
2. When a client *Y* departs or gets disconnected before having played the whole video, the server locates its predecessor *X* and its successor *Z.* if such successor exists, the server instructs client *X* to start forwarding video data directly to client *Z,* thus bypassing the departed client. If client *Y* had no successor, the server restarts a flow of free vouchers to client *X.*

Note that *pay-to-play* takes no specific action when a client terminates after having played the whole video. Its successor will merely turn to the server to obtain the missing portion of the video and "pay" the server with the vouchers it keeps receiving from its successor.

A. Cheating prevention

Essential to our proposal is the design of its vouchers. It should prevent clients from forging vouchers or resubmitting copies of vouchers that they have already sent. In addition, their predecessors should be able to detect forged or resubmitted vouchers without having to consult the server. As shown on Fig. 5, pay-to-play vouchers contain:

1. A *chain number* unique to each group of clients such that subsequent clients in the group receive their data from their immediate predecessor,
2. A *sequence number* unique to each voucher sharing the same chain number,
3. A *timestamp*,
4. An *expiration time*,
5. A *value* specifying the amount of data that can be purchased with it. (This field can be made implicit—and omitted if all vouchers have the same value).

In addition, each voucher will be signed with the private key K_{ss} of the server. As a result,

1. Peers can easily verify the authenticity of the vouchers they receive from their successors by checking their digital signature.
2. They can also detect duplicates of previously received vouchers by checking their sequence numbers and their expiration times.
3. Vouchers are specific to the chain for which they were created and cannot be sued in another chain.

B. Limitations

Owing to its simplicity, our pay-to-play mechanism has some important limitations.

First, our mechanism cannot reward clients that keep forwarding video data to their successors after they have received all the data they need. While we expect many clients to keep forwarding data as long as they are playing the video, this behavior remains altruistic in nature.

Second, our mechanism assumes that a client will receive all its video data from a single predecessor client. This is essential for preventing a rogue client to submit the same voucher more than once. The sole exception to this rule is when the predecessor of a peer stops forwarding video data to its successor. In that case, the predecessor will be excluded from the chain and the client will receive the remainder of the video from the predecessor of its former predecessor.

Finally, our mechanism requires that all clients in a chain exchange their data and their vouchers at the same rate because no client can forward video data faster than it gets them from its predecessor or "pay" for these data with vouchers it has not yet received from its predecessor.

C. Extending pay-to-play functionality

As pay-to-play has no mechanism for rewarding clients that keep forwarding video data to the clients after they have received all the data they need, they cannot reward clients that keep forwarding video data after they have finished playing the video. The most precious of these clients are those that have kept in their buffer the whole contents of the video. Wang *et al.* call these clients *upload peers* [WW+10]. They are often referred to as *seeds* [C03].

An easy way to reward these seeds would be to let these upload peers open accounts with the server where they could deposit their unspent vouchers for later use. Security considerations would require these deposits to be made promptly to let the server ascertain the validity of these vouchers. Clients wanting to spend their accumulated credit from their account would be issued new vouchers in replacement of the old ones.

In its essence, this new extended pay-to-play mechanism would combine the best features of our proposal with those of Wang's *et al.* mechanism as it would reward

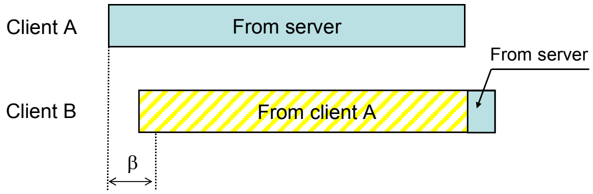

Fig. 6. How standard chaining works with selfish clients.

all cooperating clients without incurring the cost of maintaining accounts for all of them.

IV. Application to Chaining Protocols

In this section, we investigate how pay-to-play interacts with various chaining protocol. As in a previous work [PS10], we will focus on the behavior of customers watching entire videos from beginning to end and assume that customer requests arrive continuously and independently of each other with a constant rate λ. The time between arrivals is then governed by the exponential distribution, whose probability density function is $p(t) = \lambda e^{-\lambda t}$.

In all our models, D will denote the video duration and t the time elapsed between two consecutive requests. Bandwidths will always be measured in multiples of the video consumption rate.

A. Standard Chaining

Recall that chaining assumes that all clients keep forwarding data after they have finished watching a video. As a result, clients arriving up to β minutes after their immediate predecessor will receive all their data from it. Conversely, other clients will get all their video data from the server.

The average server workload w for a video of duration D will be

$$w = \int_0^{\beta} 0\lambda e^{-\lambda t}dt + \int_{\beta}^{\infty} D\lambda e^{-\lambda t}dt = De^{-\lambda\beta} \tag{3}$$

and the total server bandwidth B will be

$$B = \lambda w = \lambda D e^{-\lambda\beta}, \tag{4}$$

which goes to zero when λ goes to infinity.

We consider now what would happen if the clients were selfish and stopped forwarding video data as soon as they have received all the video data they need, which happens just when they have finished playing the video. As Fig. 6 shows, a client B arriving t time units (but with $t < \beta$) after its predecessor A would not be able to get all its video data from the previous client because that client would only be willing to forward the first $D - t$ minutes of the video to client B before disconnecting. As a result, client B would have to get t minutes of video from the server. The average server workload w for a video of duration D will be

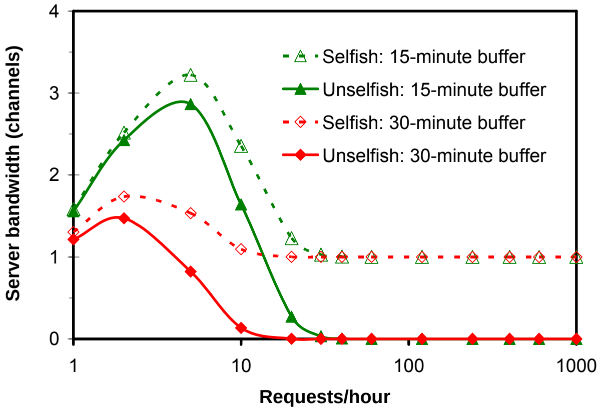

Fig. 7. How the performance of standard chaining is affected by client selfishness and the size of the client buffer β.

$$\begin{aligned} w &= \int_0^{\beta} t\lambda e^{-\lambda t}dt + \int_{\beta}^{\infty} D\lambda e^{-\lambda t}d \\ &= \frac{e^{-\lambda\beta}}{\lambda}(e^{\lambda\beta} + \lambda(D-\beta) - 1) \end{aligned} \tag{5}$$

and the total server bandwidth B will be

$$B = \lambda w = e^{-\lambda\beta}(e^{\lambda\beta} + \lambda(D-\beta) - 1), \tag{6}$$

which goes to one as λ goes to infinity.

Figure 7 compares the bandwidth requirements of the standard chaining protocol for two buffer sizes assuming the two extreme positions that (a) all clients are unselfish and (b) all clients are selfish. Request arrival rates are expressed in arrivals per hour and bandwidths are expressed in channels, whose bandwidths are equal to the video consumption rate. As one can see, the maximum impact on server bandwidth of selfish and unselfish behaviors happens at very high request arrival rate and never exceeds one video channel. Thus implementing a more complex incentive mechanism would only result in a limited benefit.

B. Advanced Chaining and Optimal Chaining

These two protocols work better with an incentive mechanism that rewards clients that keep forwarding video data to the other clients after they have finished playing the video.

C. Expanded Chaining

Recall that expanded chaining assumes that all clients have a buffer capable of holding an entire video but does not require them to forward video data once they have finished playing the video. Hence the total server bandwidth B for a video of duration D can be obtained by replacing β by D in Equation (6) giving

$$B = \lambda w = 1 - e^{-\lambda D}, \tag{7}$$

which remains less than one for all finite values of the arrival λ rate and goes to one as λ goes to infinity.

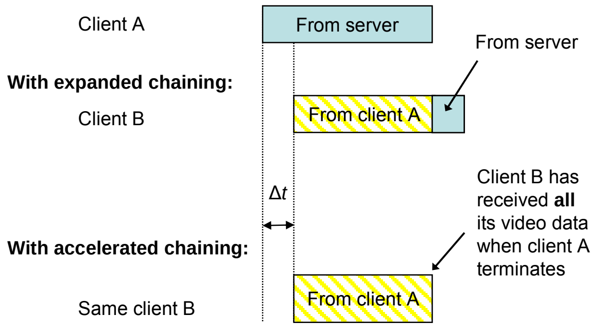

Fig. 8. How accelerated chaining lets some clients receive all their video data before they are finished playing the video.

Expanded chaining is an ideal match for our pay-to-play protocol as the time at which a client has received all its data coincides with the time at which it has finished playing the video.

D. *Accelerated Chaining*

Accelerated chaining improves upon the performance of the expanded chaining protocol by requiring clients to forward video data to their successors at slightly higher rate than the video consumption rate [PAL 10]. Interestingly enough, significant performance improvements can be achieved with video acceleration factors *f* not exceeding between 1.01 and 1.10, that is, by having clients forward video data 1 to 10 percent faster than the video consumption rate.

These very good results assume that clients remain willing to forward video data until they have finished playing the video. This is true for expanded chaining because clients receive all their video data in real time and thus remain motivated to collect vouchers from their successors until they have finished playing the video. As we can see on Figure 8, this is not always true with accelerated chaining. Clients that arrive shortly enough after their immediate predecessor will get all their video data from their predecessor before they have finished playing the video. After that, they will have no incentive to keep forwarding video data to their successors.

To evaluate the impact of our pay-to-play incentive mechanism on the performance of accelerated chaining, we wrote a simple simulation program assuming that request arrivals for a particular video were distributed according to a Poisson law. Our program was written in C and simulated requests for a single two-hour video. Simulation durations were selected in a way that guaranteed that each run simulated a minimum of 10,000 hours of simulated time and a minimum of 100,000 request arrivals.

Since no data are shared among customers watching different videos, the total bandwidth of a server distributing several videos would always equal the sum of the bandwidths it dedicates to each video.

We considered two extreme values for the video acceleration factor *f*, namely 1.01 and 1.10. We measured the average server bandwidth at request arrival rates varying between one and one thousand requests per hour.

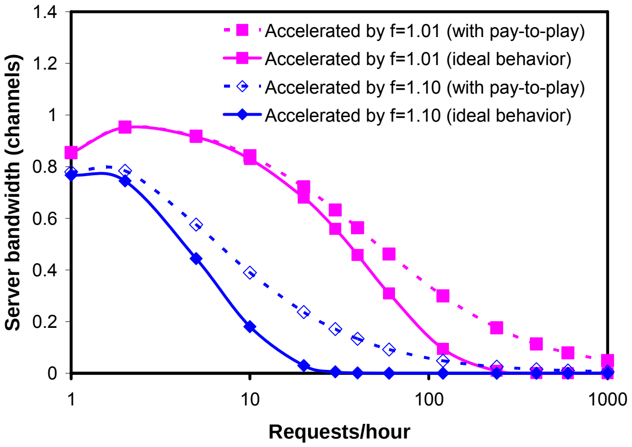

Fig. 9. Estimated performance impact of adding incentives to accelerated chaining.

Our results are summarized in Figure 9. As before, request arrival rates are expressed in arrivals per hour and bandwidths are expressed in multiples of the video consumption rate. We compare the server workloads corresponding to:

1. The ideal case where all clients keep forwarding video data to their successors until they have finished playing the video.
2. A limit case where all clients stop forwarding video data to their successors as soon as they have received all their video data.

As we can see, early retransmission terminations have a small but significant impact on the server workload at arrival rates between 5 and 120 requests per hour. This impact is negligible at lower arrival rates because very few clients get all their video data from the previous clients and becomes almost imperceptible at higher arrival rates as the server workload becomes very small.

The critical issue here is the relative frequencies of selfish and unselfish client behaviors. Let us consider first the main difference between the ways standard chaining and accelerated chaining define selfish client behaviors. For standard chaining, all clients that disconnect from the service immediately after they played the whole video are considered to act in a selfish manner. This is not true for accelerated chaining. Clients are considered to act in a selfish manner whenever they disconnect after they have received the entire video and they have not yet finished playing it. While some unselfish clients may decide to stay connected to the service and keep forwarding video data after they have played the whole video, they are not expected to do so.

We can safely assume that the default accelerated chaining client software will be programmed to operate in an unselfish manner. Hence the sole selfish clients will be those having installed non-standard client softwares that implement a selfish client policy. Hence, the most likely state of affairs will not be different from that of BitTorrent where selfish users are in the minority [HP05].

V. Conclusion

We have presented pay-to play, a voucher-based incentive mechanism for peer-to-peer systems distributing videos on demand using a chaining protocol. In our proposal, clients collect vouchers from the downstream clients to which they forward data and use these vouchers to reward the upstream clients that send them data. Our proposal requires these vouchers to be assigned sequence numbers and be signed with the private key of the server. As a result, vouchers circulate among the clients without any server intervention.

We have investigated the operation of our pay-to-play mechanism with several chaining protocols for video-on-demand and found out that:

1. Pay-to-play cannot ensure that the standard chaining protocol operates in an optimal fashion as they do not reward clients that keep forwarding data to their successors after they have finished watching a video.
2. Pay-to-play is not the right incentive mechanism for the advanced chaining and the optimal chaining protocols as both protocols rely heavily on the contributions of idle clients.
3. Pay-to-play provides the right incentives to ensure the optimal operation of the expanded chaining protocol.
4. Pay-to-play cannot ensure that the accelerated chaining protocol operates in an optimal fashion in the presence of selfish clients but, even then, the impact on server bandwidth remains minimal.

More work is still needed to develop variants of the accelerated chaining protocol that would better interact with our pay-to-play incentive mechanism and investigate how the incentive mechanism would support interactive commands such as pause, rewind and forward.

References

[AF+08] C. Aperjis, C. A. Freedman, and R, Johari. "Peer-assisted content distribution with prices," *Proc. ACM SIGCOMM Conference on emerging Networking Experiments and Technologies* (CoNext '08), Madrid, Spain, pp. 17:1–12, Dec. 2008.

[C03] B. Cohen. "Incentives build robustness in Bit Torrent," *Proc. Workshop on Economics of Peer-to-Peer Systems*, Berkeley, CA, 2003.

[C10] Cisco, Inc. *Cisco Visual Networking Index: Forecast and Methodology, 2010-2015,* Accessed August 18, 2011 from http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html.

[GL+01] P. Golle, K. Leyton-Brown, and I. Mironov. "Incentives for sharing in peer-to-peer networks," *Proc. ACM Electronic Commerce Conference* (EC '01), Tampa, Florida, pp. 264–267, Oct. 2001

[HP05] D. Hales and S. Patarin. *How to Cheat BitTorrent and Why Nobody Does*, Technical Report UBLCS-2005-12, University of Bologna, Bologna, Italy, May 2005

[LGL08] Y. Liu, Y. Guo and C. Liang. "A survey on peer-to-peer video streaming systems," *Peer-to-Peer Networking and Applications*, 1(1):18–28, March 2008.

[LZ+07] F. Lin, C. Zheng, X. Wang, X. Xue. "ACVoD: A peer-to-peer based video-on-demand scheme in broadband residential access networks," *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4)4, 2007.

[MP+08] J. J. D. Mol, J. A. Pouwelse, M. Meulpolder, D. H. J. Epema, and H.J. Sips. "Give-to-Get: An algorithm for P2P video-on-demand," *Proc. 13th Annual Multimedia Computing and Networking Conference* (MMCN 2008), San Jose, California, Jan. 2008.

[P05] J.-F. Pâris. "A cooperative distribution protocol for video-on-demand." *Proc. 6th Mexican International Conference on Computer Science,* Puebla, Mexico, pp. 240–246, Sep. 2005.

[PAL11] J.-F Pâris, A. Amer and Darrell D. E. Long "Accelerated chaining: a better way to harness peer power in video," *Proc. 26th ACM Symposium on Applied Computing* (SAC 2011), Taichung, Taiwan, pp. 534–539, Mar. 2011.

[PS10] J.-F. Pâris and T. J. Schwarz. "An analysis of chaining protocols for video-on-demand," *Proc. 9th International Information and Telecommunication Technologies Symposium* (I2TS 2010), Rio do Janeiro, RJ, Brazil, Dec. 2010.

[SH+02] T.-C. Su, S.-Y. Huang, C.-L. Chan and J.-S. Wang. "Optimal chaining and implementation for large scale multimedia streaming," *Proc. 2002 IEEE International Conference on Multimedia and Expo* (ICME 2002), Lausanne, Switzerland, Vol.1, pp. 385–388, Aug. 2002.

[SH+05] T.-C. Su, S.-Y. Huang, C.-L. Chan and J.-S. Wang. "Optimal chaining scheme for video-on-demand applications on collaborative networks," *IEEE Trans. on Multimedia*, 7(5): 972–980, 2005.

[SHT97] S. Sheu, K. A. Hua, and W. Tavanapong. "Chaining: a generalized batching technique for video-on-demand systems." *Proc. IEEE International Conference on Multimedia Computing and Systems* (ICMS '97), Ottawa, Ontario, Canada, pp. 110–117, June 1997.

[SP+07] M. Sirivianos, J. Park, X. Yang, and S. Jarecki, "Dandelion: cooperative content distribution with robust incentives." *Proc. 2007 USENIX Annual Technical Conference,* Santa Clara, California, pp. 157–170, June 2007.

[VIF06] A. Vlavianos, M. Iliofotou, and M. Faloutsos. "BiToS: Enhancing BitTorrent for supporting streaming applications." *Proc. 9th IEEE Global Internet Symposium*, Barcelona, Spain, pp.1–6, Apr. 2006

[WW+10] C. Wang, H. Wang, Y. Lin, S.i Chen, "A lightweight currency-based P2P VoD incentive mechanism," *Proc. 3rd International Joint Conference on Computational Science and Optimization* (CSO 2010), Huangshan, China, vol. 1, pp.272–276, 2010

[XH+02] D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava, "On peer-to-peer media streaming," *Proc. 22nd International Conference on Distributed Computing Systems* (ICDCS 2003), Wien, Austria, July 2002.