An Analysis of Chaining Protocols for Video-on-Demand

Jehan-François Pâris
Department of Computer Science
University of Houston
Houston, TX 77204, USA

paris@cs.uh.edu

Thomas Schwarz, S. J.
Departamento de Informática y Ciencias de la Computación
Universidad Católica del Uruguay
11600 Montevideo, Uruguay

tschwarz@ucu.edu.uy

Abstract—**Chaining protocols for video-on-demand require each client to forward the video data it receives to the next client watching the same video. We present here the first analytical investigation of standard chaining, advanced chaining, optimal chaining, expanded chaining (also known as the cooperative protocol) and accelerated chaining. Our results agree fairly well with earlier results obtained through discrete-event simulation with a maximum absolute difference equal to 0.3 percent of the video consumption rate.**

I. Introduction

Chaining protocols [10] provide a simple and elegant way to reduce the cost of distributing videos on demand by involving clients in the video distribution process. Chaining organizes all clients watching a given video into chains where each client forwards the video data it has received to the next client in the chain. In its essence, this approach is the same as that of *peer-to-peer* (P2P) file sharing systems such as Gnutella [5] or BitTorrent [3].

Compared to other video-on-demand distribution protocols such *batching* [4], *pyramid broadcasting* [13] and *stream tapping* [2], chaining offers the major advantage of not requiring any multicast support, a feature missing in the vast majority of systems on the Internet [1]. In contrast, chaining only requires clients capable of forwarding video data at the rate they consume them

We present here the first analytical study of the original chaining protocol and its variants, among which *extended chaining*, *optimal chaining*, *advanced chaining* (also known as the *cooperative protocol*) and *accelerated chaining*. Our results agree fairly well with earlier results obtained through discrete-event simulation and offer a much more flexible tool for investigating the impact of protocol parameters, such as client buffer size and video duration on the performance of chaining protocols.

The remainder of the paper is organized as follows. Section 2 reviews extant chaining protocols. Section 3 introduces our methodology and presents our results. Finally Section 4 has our conclusions.

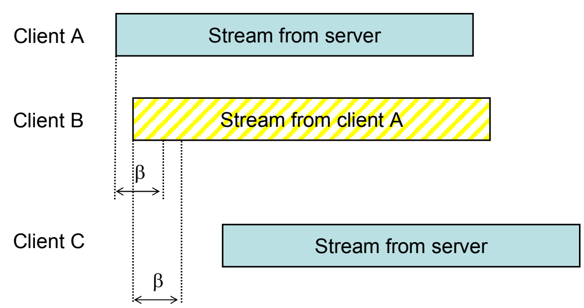

Fig. 1. How chaining works.

II. Chaining

In this section, we review the chaining protocols we will analyze, starting with the original chaining protocol.

A. Standard Chaining

Standard chaining [10] constructs chains of clients such that (a) the first client in the chain receives its data from the server and (b) subsequent clients in the chain receive their data from their immediate predecessor. As a result, video data are in some way "pipelined" through the clients belonging to the same chain. Since chaining only requires clients to have very small data buffers, a new chain has to be restarted every time the time interval between two successive clients exceeds the capacity β of the buffer of the previous client. Fig. 1 shows three sample customer requests. Since customer A is the first customer, it will get all its data from the server. Since customer B arrives less than β minutes after customer A, it can receive all its data from customer A. Finally customer C arrives more than β minutes after customer B and must be serviced directly by the server.

B. Advanced Chaining

The main weakness of chaining is its poor performance at low arrival rates, more precisely, whenever the time interval between two consecutive requests exceeds β minutes. *Advanced chaining* [6] proposes to bridge this gap by inserting every β minutes idle peers that will relay the data.

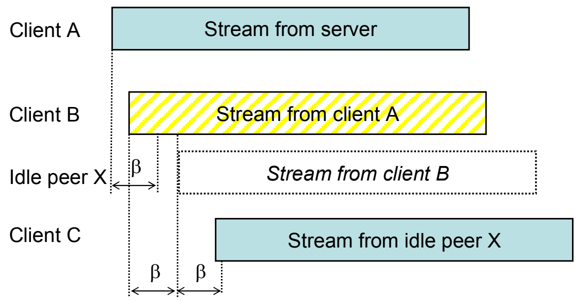

Fig. 2. How advanced chaining works.

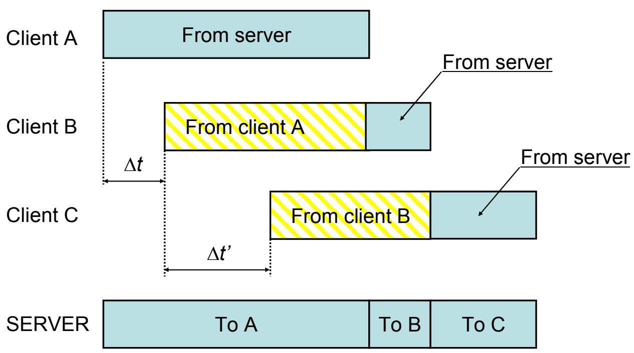

Fig. 3. How expanded chaining works.

In the previous example, we saw that client *c* could not get its video data from client *B* and had to receive them from the server because the interarrival gap between the two requests exceeded *β* minutes. As shown in Fig. 2, advanced chaining avoids that problem by scheduling an idle peer *X* as soon as a request interarrival time exceeds *β* minutes. This idle peer will receive all its video data from the precious client and relay them to any client arriving less than *β* minutes after the start of the relaying process. If needed, the process can be repeated every *β* minutes.

C. Optimal Chaining

Optimal chaining [11, 12] 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.

D. Expanded Chaining

Expanded chaining, also known as the *cooperative* video distribution protocol [8], extends the original chaining protocol by taking advantage of the larger buffer sizes of modern clients. At the same time, it assumes that clients will disconnect and stop forwarding data once they have finished playing the video. As in standard chaining, each client forwards to its immediate successor video data starting with the beginning of the video. When a client has finished playing the video, it disconnects itself and stops transmitting video data, lettting

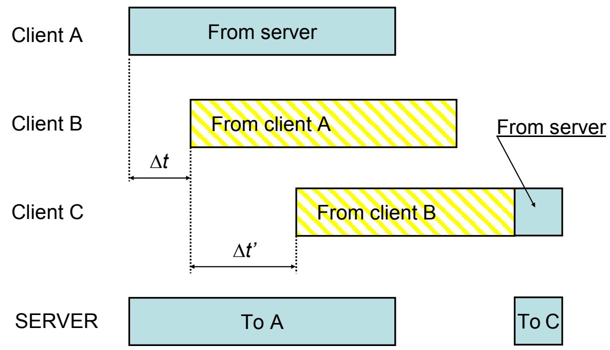

Fig. 4. How accelerated chaining works.

the server transmit the missing part of the video. Consider for instance how the protocol would handle the three requests displayed in Fig. 3 for a video of duration *D*. The first request to the video will be entirely serviced by the server. Since the second request arrives while the first request is still being serviced, the server will thus instruct client *A* to forward the first $D-\Delta t$ minutes of the video to client *B* and schedule a transmission of the last Δt minutes of the video to the same client at a later time. Similarly, the server will instruct client *B* to forward the first $D-\Delta t'$ minutes of the video to client *C* and schedule a transmission of the last $\Delta t'$ minutes of the video to the same client at a later time.

More generally, the amount of time spent by the server to service a request will always be given by min(D, Δt), where *D* is the duration of the video and Δt is the time interval between the request being serviced and its immediate predecessor. In addition, the service times of these requests will never overlap, which means that the server instantaneous bandwidth $B(t)$ will never exceed the video consumption rate.

E. Accelerated chaining

Standard chaining and advanced chaining completely eliminate the server workload whenever the time interval between two consecutive requests remains below *β* minutes. This excellent performance is unfortunately based on the assumption that clients will always keep forwarding data to their successor in the chain, even after they have finished playing the video. This is not a realistic assumption as most clients are likely to disconnect once they have played the video. In addition, a significant number of clients will disconnect without having played the full video.

Accelerated chaining [9] achieves similar bandwidth savings as these two chaining protocols 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. 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 $\rho \cdot D$ time units. Conversely, during time T, a client can obtain video data to be displayed in $f \cdot T$ time units. 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 time of $D - \Delta t$. Hence, the second client can receive the entire video from the first client as long as $\rho D \leq D - \Delta t$. This condition is equivalent to

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

or

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

Define

$$\Delta t^* = D(1-\rho)D, \quad (3)$$

Accelerated chaining will operate in the following 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 request arrival rate.

III. Our Models

All extant evaluations of the performance of chaining protocols [6, 8–10] have relied on discrete-event simulation. They thus share the common weakness of all simulation studies, namely, that they only produce numerical results.

We will focus on the behavior of customers watching entire videos from beginning to end. We reserve for a further study the case of customers interrupting their viewing before the end of the video, pausing while watching the video, moving forward and backward.

We will distinguish between *selfish clients,* who stop forwarding data as soon as they have finished watching the video, and *unselfish clients*, who are willing to keep forwarding data after they have finished watching a video. These unselfish clients correspond to the *seeds* of a conventional P2P protocol.

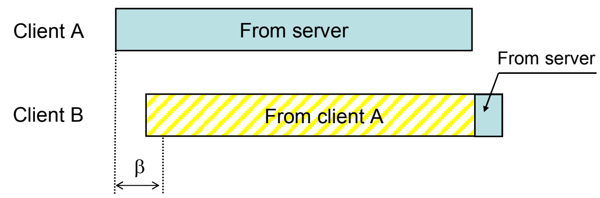

Fig. 5. How chaining works with selfish clients.

We will assume that customer requests arrive continuously and independently of each other with a constant rate λ. They are therefore modeled with a Poisson process. The time between arrivals is then governed by the exponential distribution. The probability density of this distribution 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. Chaining

Recall that chaining posits that all clients are unselfish and are willing to keep forwarding data after they have finished watching a video. As a result, a client arriving up to β minutes after its immediate predecessor will receive all its data from it. Conversely, a customer arriving more than β minutes after its predecessor will get all its video data from the server.

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

$$w = \int_0^\beta 0\lambda e^{-\lambda t} dt + \int_\beta^\infty D\lambda e^{-\lambda t} dt = De^{-\lambda\beta} \quad (2)$$

and the total server bandwidth B will be

$$B = \lambda w = \lambda De^{-\lambda\beta} \quad (3)$$

This function has a single maximum $B_{max} = D/\beta e$ for $\lambda_{max} = 1/\beta$. Observe that the maximum server bandwidth grows linearly with the inverse of the fraction β/D of the video that client buffers can store. Since

$$\lim_{\lambda\to\infty} \lambda De^{-\lambda\beta} = 0,$$

the server bandwidth requirements of basic chaining are actually decreasing as the request arrival rates increase over $1/\beta$ and quickly become negligible.

Consider now what would happen if the clients were selfish and stopped forwarding video data as soon as they have finished watching the video. As Fig. 5 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

$$
\begin{aligned}
w &= \int_0^{\beta} t\lambda e^{-\lambda t}dt + \int_{\beta}^{\infty} D\lambda e^{-\lambda t}dt \\
&= \frac{1}{\lambda} - \frac{(\lambda\beta+1)e^{-\lambda\beta}}{\lambda} + De^{-\lambda\beta} \\
&= \frac{e^{-\lambda\beta}}{\lambda}(e^{\lambda\beta} + \lambda(D-\beta) - 1)
\end{aligned} \quad (6)
$$

and the total server bandwidth B will be

$$B = \lambda w = e^{-\lambda\beta}(e^{\lambda\beta} + \lambda(D-\beta) - 1) \quad (7)$$

This function has a maximum at $\lambda_{max} = D(D-\beta)^{-1}\beta^{-1}$. The maximum bandwidth is then given by

$$B_{\max} = 1 + e^{-\frac{D}{D-\beta}}(\frac{D-\beta}{\beta})$$

As Fig. 6 shows, larger values for β result in smaller peak bandwidth attained at smaller arrival rates. The peak becomes almost negligible as soon as $\beta \geq D/2$. We can also see that

$$\lim_{\lambda\to\infty} B = 1$$

for all values of β.

B. *Advanced Chaining*

The performance of the advanced chaining protocol essentially depends on the number of available idle peers. With a sufficient number of them, we would be able to schedule one idle peer every β minutes whenever two successive requests are separated by more than β minutes. As a result, all client requests could be satisfied either by the previous client or an idle peer, thus eliminating the server workload.

C. *Optimal Chaining*

Analyzing optimal chaining raise the same issue as analyzing advanced chaining, namely the availability of peers that can help data forwarding between consecutive peers separated by more than β minutes. If enough peers are willing to offer enough buffer space for enough time, all incoming clients could receive their video data from their immediate predecessor, which would again eliminate the server workload.

D. *Expanded Chaining*

Recall that expanded chaining assumes that client buffers are large enough to store the whole contents of each video and that clients are selfish. Hence the average server workload w for a video of duration D can be obtained by replacing β by D in Equation (6) giving

$$w = \frac{1}{\lambda}(1-e^{-\lambda D}) \quad (8)$$

and the total server bandwidth B will be

$$B = \lambda w = 1 - e^{-\lambda D} \quad (9)$$

The function has no strictly positive extrema and

$$\lim_{\lambda\to\infty} 1 - e^{-\lambda D} = 1 .$$

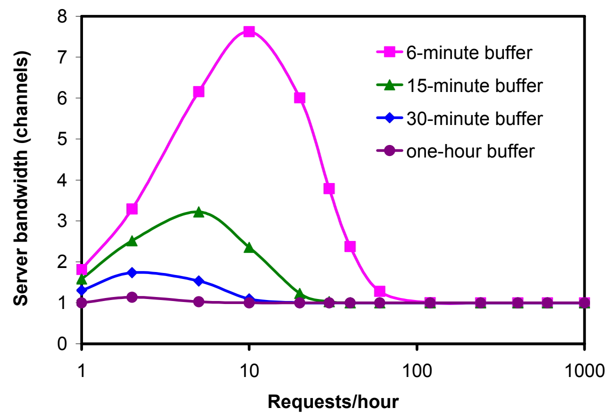

Fig. 6. Impact of buffer size on the server bandwidth requirements of the basic chaining protocol for a two-hour video distributed to selfish clients.

E. *Accelerated chaining*

We recall that if a request arrives more than D units after the previous request, then the server will have to provide the complete video. If the request arrives after a time interval t between $(1-\rho)D$ and D after the previous request, then the server will provide $(D-t)/\rho$ of the video. If the interval is less than $(1-\rho)D$, the client will receive all its video data from its predecessor. The average server workload is thus

$$
\begin{aligned}
w &= \int_{(1-\rho)D}^{D} (D-(D-t)/\rho)\lambda e^{-\lambda t}dt + \int_{D}^{\infty} D\lambda e^{-\lambda t}dt \\
&= \frac{\exp(-\lambda D)(\exp(\rho\lambda D)-1)}{\lambda\rho}
\end{aligned} \quad (10)
$$

The total server bandwidth is λw or

$$B = \frac{\exp(-\lambda D)(\exp(\rho\lambda D)-1)}{\rho} \quad (11)$$

This function attains its maximum at

$$\lambda_{\max} = \frac{-\log(1-\rho)}{\rho D}$$

with value

$$B_{\max} = (1-\rho)^{(1/\rho)-1},$$

which is lesser than one for all values of ρ corresponding to an acceleration factor f greater than one, as it should be.

As $B \rightarrow 0$ for $\lambda \rightarrow \infty$, the server bandwidth requirements of accelerated chaining become negligible as the arrival rate of requests increases.

F. *Comparison with Simulation Results*

We compared our analytical results with those previously obtained using discrete event simulation. Our simulation program assumed that request arrivals for a particular video were distributed according to a Poisson law and simulated requests for a single two-hour video. We measured the average server bandwidth at request arrival rates varying between one and one thousand requests per hour. We did not

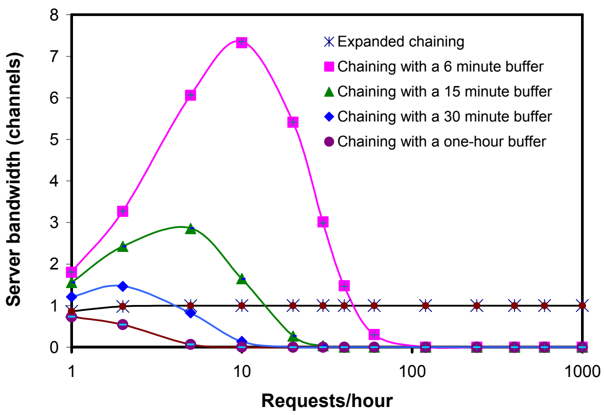

Fig. 7. Comparing analytical and simulation results for basic chaining and expanded chaining.

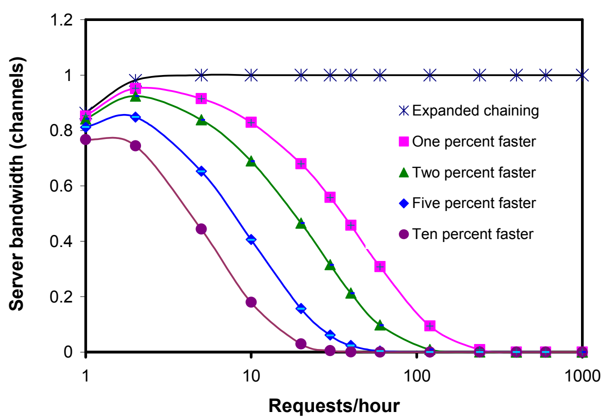

Fig. 8. Comparing analytical and simulation results for expanded chaining and accelerated chaining.

consider higher arrival rates as they seemed unrealistic. Each simulation run involved at least 100,000 arrivals over a simulated time period of at least 10,000 hours.

Our results are summarized in Fig. 7 and 8. Request arrival rates are expressed in arrivals per hour and bandwidths are expressed in multiples of the video consumption rate. Each specific point displays a simulation result while the continuous curves were computed using Equations 5, 7, 9 and 11. We did not include the values for advanced chaining and optimal chaining as we assumed their bandwidth requirements would be negligible as long as there are enough helping peers.

As we can see, there is a fairly strong agreement between our analytical results and our simulation measurements for all the chaining protocols we investigate. The highest absolute difference between our simulation data and our analytic models was 0.3 percent of the video consumption rate while the typical absolute difference was less than 0.06 percent.

IV. Conclusions

We have presented here the first analytical investigation of the *standard chaining* protocol and its variants, among which *advanced chaining*, *optimal chaining*, *expanded chaining* and *accelerated chaining*. Our results agree fairly well with earlier results obtained through discrete-event simulation and offer a much more flexible technique for investigating the impact of protocol parameters, such as client buffer size and video duration on the performance of chaining protocols.

More work is still needed to define the best implementation of the fast forward control and specify an incentive mechanism that motivates clients to forward video data at the appropriate rate [7].

References

[1] M. Ammar. "Why Johnny can't multicast: lessons about the evolution of the Internet," Keynote Address, *Proc. 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video* (NOSDAV 2003), Monterey, CA, p. 1, June 2003.

[2] S. W. Carter and D. D. E. Long. "Improving video-on-demand server efficiency through stream tapping," *Proc. 5th International Conference on Computer Communications and Networks* (ICCCN '97). Las Vegas, NV, pp. 200–207, Sep. 1997.

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

[4] A. Dan, P. Shahabuddin, D. Sitaram and D. Towsley. "Channel allocation under batching and VCR control in video-on-demand systems," *Journal of Parallel and Distributed Computing*, 30(2):168–179, Nov. 1994.

[5] P. Kirk. *Gnutella—A protocol for a revolution*. http://rfc-gnutella.sourceforge.net/

[6] 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.

[7] 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, CA, Jan. 2008.

[8] 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.

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

[10] 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), Berlin, Germany, pp. 110–117, June 1997.

[11] 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.

[12] 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 Transactions on Multimedia*, 7(5): 972–980, 2005.

[13] S. Viswanathan and T. Imielinski. "Metropolitan area video-on-demand service using pyramid broadcasting," *Multimedia Systems*, 4(4):197–208, Aug. 1996.