Proxy Caching for Video on Demand Systems in Multicasting Networks

Zhu, L.; Cheng, G.; Ansari, N.; Sahinoglu, Z.; Vetro, A.; Sun, H.

TR2003-55 July 07, 2003

Abstract

Streaming high quality videos consumes a significant amount of network resources. Researchers proposed prefix caching schemes to reduce bandwidth usage costs in streaming videos. In this paper, we introduce a wide-scale cost model for proxy caching that takes bandwidth consumption into consideration over an entire network for different multicasting tree topologies. The new cost model quantifies the overall usage of network resources more accurately. We have also investigated the feasibility of prefix caching at proxies in multicasting networks, and, contrary to a recent claim, found out that prefix caching is cost effective in a wide range of different network conditions and service request rates.

Conference on Information Sciences and Systems (CISS) 2003

Proxy Caching for Video on Demand Systems in Multicasting Networks

Li Zhu, Gang Cheng, Nirwan Ansari
Advanced Networking Lab
New Jersey Institute of Technology
{lz6, gc2, nirwan.ansari}@njit.edu

Zafer Sahinoglu, Anthony Vetro, Huifang Sun
Mitsubishi Electric Research Laboratories
Murray Hill, NJ07974
{zafer, avetro, hsun}@merl.com

Abstract-
Streaming high quality videos consumes a significant amount of network resources. Researchers proposed prefix caching schemes to reduce bandwidth usage costs in streaming videos. In this paper, we introduce a wide-scale cost model for proxy caching that takes bandwidth consumption into consideration over an entire network for different multicasting tree topologies. The new cost model quantifies the overall usage of network resources more accurately. We have also investigated the feasibility of prefix caching at proxies in multicasting networks, and, contrary to a recent claim, found out that prefix caching is cost effective in a wide range of different network conditions and service request rates.

I. INTRODUCTION

Recent advances in Internet and digital video technology have made Video on Demand (VoD) possible. VoD includes many applications such as distant learning, movie on demand, news on demand, etc. A VoD system usually consists of several central servers and distributed clients over the entire network. Pre-recorded videos are stored in central servers and sent to clients at their requests. There are two types of VoD services: the "true" VoD and the "near" VoD. For the "true" VoD, clients are served immediately after their requests are received. While for the "near" VoD, client requests may be served after being delayed by a certain amount of time, such as several minutes. With the properties of long lasting and high bandwidth consumption, streaming videos can significantly reduce network resources. There are many VoD schemes proposed to address this problem: *batching*, *patching*, *periodical broadcasting* and *prefix caching*.

In the batching scheme [1], the server batches requests for the same video clip together if their arrival times are close, and serve them by one multicast channel. The limit of batching is that it can only provide near VoD. In the patching scheme [2], the server sends the entire video clip to the first client. Later clients can join the existing multicast channel, and at the same time each of them requires a unicast channel to deliver the missing part of the video.

Periodical broadcasting [3], [4] is another innovative technique. In this approach, popular video clips are partitioned into a series of segments and these segments are continually broadcasted on several dedicated channels. Before clients start playing videos, they usually have to wait for a time length equivalent to the first segment. Therefore, only near VoD service is provided.

Proxy caching [5], [6] is also a promising scheme to alleviate the bandwidth consumption issue. In this approach, there exist proxies between a central server and client clouds. Partial video (or entire video) files are stored in proxies and the rest are stored in the central server. Proxies send cached videos to clients, and request the remaining from servers on behalf of clients. Zhang *et al*. [6] proposed a "video staging" algorithm, which stores the bursty part of video frames in the proxy so that the bandwidth requirement between the server and the proxy is significantly reduced. Sen *et al*. [5] proposed the "prefix caching," which stores in the proxies the beginning (prefix) of video files to reduce the traffic load between the server and the proxies. All these proxy-caching schemes support the true VoD service.

With the rapid emergence of applications with high bandwidth requirements, the bandwidth in the network will become more scarce and precious. Therefore, it is very important to minimize the bandwidth consumption for streaming video files. Researchers in [7], [8] associated a cost with transmitting a certain amount of data through the network. They adopted a similar cost model by assuming that the cost to deliver one unit of data from the server is one and the cost to deliver one unit from the proxy is β ; β is usually smaller than one. Chan *et al.* [7] studied the tradeoff between the network transmission cost, and the local storage cost and tried to minimize the total cost of the system for several proxy-caching schemes. Wang *et al*. [8] assumed that only unicast existed between the central server and the proxies, and proposed prefix-caching schemes so that the video transmission cost was greatly reduced.

A previous work in [9] developed simple cost models to investigate the feasibility of proxy caching. In their analysis they adopted bandwidth-skimming schemes [10], and reached some quite surprising conclusions. One of their major claims is that if multicast is available between the central server and the proxies, proxy caching is not cost effective unless the video request rate is very low, or β is as low as $1/P$, where P is the number of proxies. Their conclusions imply that even β is very small, it is still not beneficial to deploy many proxies over the entire network.

One of our contributions in this paper is that we have developed a more realistic and accurate cost model, which takes the bandwidth consumption over entire network into account. To our best knowledge, this is the first time that such a cost model is proposed. Starting from this model, we have reached very different conclusions from [9]. Our study showed that when multicasting is available in the network between the server and proxies, the proxy-caching scheme is cost effective within a much wider range of β .

The remainder of this paper is organized as follows. Section II provides the necessary background and presents our problem statement. Section III presents our proposed cost model and provides its performance evaluation for caching a single video file. Section IV provides the simulation results for caching multiple video files with a limited proxy storage capacity. Conclusions are drawn in Section V.

II. BACKGROUND and PROBLEM FORMULATION

We first briefly review the bandwidth skimming scheme. Readers are referred to [10] for more details. The basic idea of bandwidth skimming is to use hierarchical multicast stream merging (HMSM) to dynamically aggregate clients into larger and larger groups that share the same multicast streams. Each new client opens a new multicast channel and at the same time listens to the closest active channel (target channel). After having received the missing part from the target channel, the client is merged to the target stream. In the same way, the clients in the target stream can also be merged into another new target stream.

In this paper, we assume that there are one central server and a total number of *P* proxies. Each proxy has a fixed size disk space to store the beginning of each video file (*prefix*), and transmits the prefix to their serviced clients. The server can also stream the remainder of the video files (*suffix*) directly to clients instead of through the proxies. In this paper, we adopt BWSkim(2) scheme, in which the server, as well as proxies, uses the bandwidth skimming scheme to deliver video streams and each client can receive data from at most two streams at the same time.

The following notations similar to [9] are adopted:

λ : The average client request arrival rate for one video file

λ_k : The average client request arrival rate for one video file from the clients served by the k^{th} proxy

L : The video file length (in minutes)

N : The average client request arrival rate per L, ($N = \lambda L$)

P: The number of proxies

f : The fraction of each video file stored locally at each proxy

β : The ratio of cost per video stream from proxy to the cost per video stream from the server

F_{proxy} : The number of flows required to deliver videos from a proxy

F_{server} : The number of flows required to deliver videos from the server

In this paper we assume that the client request arrival rates at every proxy are the same and all proxies store the same fraction f of original video files in their local disks. Thus, $\lambda_k = \lambda / P$, for *k=1,2…P*. Accordingly, we have

$$\sum_{j=1}^{P} \lambda_k = \lambda . \quad (1)$$

If the entire video file is stored in the server, the average number of flows originated from the server can be computed as [10]:

$$F_{server} = \eta \ln(1+\frac{N}{\eta}) = \eta \ln(1+\frac{\lambda L}{\eta}) , \quad (2)$$

where $\eta = 1.62$. Therefore, if the fraction f of the video file is stored in the proxy, the number of streams originating from a proxy to deliver the prefix to its served clients is:

$$F_{proxy} = \eta \ln(1+N_{proxy}/\eta) , \quad (3)$$

where $N_{proxy} = \frac{\lambda}{P} fL$ is the average client request arrival rate per prefix length (*fL*) at each proxy. On the server side, the arrival rate of requests for the suffix from clients served by one proxy can be estimated as [9]:

$$\lambda^{'} = \frac{dF_{proxy}}{d(fL)} = \eta \frac{\lambda / P}{fT\lambda / P + \eta} . \quad (4)$$

Thus, the overall arrival rate of requests for the suffix at the server is

$$\lambda_{server} = P \times \lambda^{'} = P\eta \frac{\lambda / P}{fT\lambda / P + \eta} . \quad (5)$$

From (2), we can express the total number of flows originated from the server as

$$F_{server} = \eta \ln(1+N_{server}/\eta) , \quad (6)$$

where $N_{server} = \lambda_{server}(1-f)L = \eta \frac{(1-f)N}{fN/P+\eta}$ is the total client request rate per suffix length (*(1-f)L*) at the server. Readers are referred to [9], [10] for detailed derivation of the above formulas.

III. COST MODEL and PERFORMANCE EVALUATION

Assuming the bandwidth needed to transmit one stream is normalized to one, [9] proposed the following cost model:

$$C = F_{server} + P \times \beta \times F_{proxy} . \quad (7)$$

However, this model may not be suitable if we want to minimize the transmission cost over the entire network. It is true that F_{server} is the number of streams leaving the server, but it is not the number of the “end-to-end” streams in the network. When any single stream among these F_{server} streams leaves the server, it will split into multiple streams along the multicasting tree. This results in many more end-to-end streams than F_{server}, in other words, more bandwidth consumption. Therefore, it is more reasonable to take all the bandwidth consumption into account if the objective is to minimize the total usage of the network resource. In the remainder of this section, we propose a new cost model for two typical multicasting tree topologies. In this section, we assume there is only one video file and the proxy disk space is large enough to store the whole file locally. In the next section, we will consider the case of caching multiple video files with a limited disk space.

Fig. 1. shows a shared-link fan-out multicasting tree [11]. Each client cloud represents clients served by one proxy. Therefore, the number of client clouds is also P. In this paper, we assume that each client cloud is much smaller compared to the multicasting tree over the backbone. Thus, we neglect the network topology of the client clouds. On the other hand, if we take the network topology of the client clouds into account (this is our on-going work), it will change the value of β. According to (5) and (3), the request arrival rate from each client cloud is $\lambda' = \lambda_{server}/P$, and F_k, the average number of flows at link l_k ($k=1,2,\ldots P$), has the same value [11]:

$$F_k = \eta \ln(1+\lambda' fL/\eta) = \eta \ln(1+\frac{N_{server}/P}{\eta}) \quad (8)$$

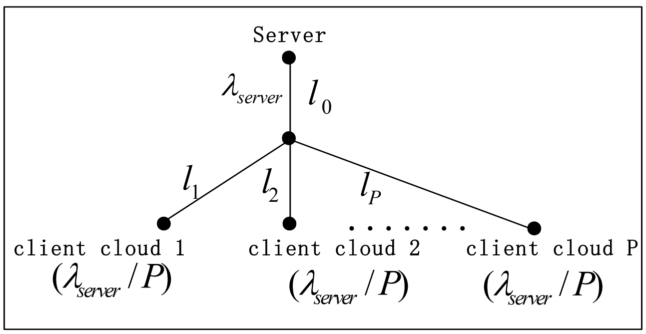

Fig. 1 Shared link fan-out topology

The overall request arrival rate at the server is $\lambda_{server} = P\lambda'$, and the average number of flows F_0 on link l_0 is just F_{server} in (6). If we choose the unit of transmission cost as the cost of delivering one end-to-end stream, the lower bound of the total cost of streaming the suffix from the server is:

$$C_{suffix}^{\min} = \frac{1}{2}\sum_{k=0}^{P} F_k = \frac{\eta}{2}\{\ln(1+N_{server}/\eta) + P\ln(1+\frac{N_{server}/P}{\eta})\}. \quad (9)$$

The total cost of delivering the prefix from all proxies is

$$C_{prefix} = \beta P F_{proxy}, \quad (10)$$

where F_{proxy} is expressed in (3).

Using (9) and (10) we can estimate the overall cost as:

$$\begin{aligned} C &= C_{suffix}^{\min} + C_{prefix} \\ &= \frac{\eta}{2}\{\ln(1+N_{server}/\eta) + P\ln(1+\frac{N_{server}/P}{\eta})\} \\ &\quad + \eta\beta P\ln(1+N_{proxy}/\eta) \end{aligned} \quad (11)$$

Next, we change the topology to a binary tree [11], as shown in Fig. 2. Again, we assume a homogeneous request arrival pattern, meaning that each client site has the request arrival rate $\lambda' = \lambda_{server}/P$ for the suffix from the server. For this binary tree with depth L, we have $P = 2^{L+1} - 2$. If we use the same definition for the unit of cost as that for the shared link fan-out topology, the lower bound of the total cost to deliver the suffix from the server can be estimated as [11]:

$$C_{suffix}^{\min} = \frac{2^L - 1}{2^L(L-1)+1}\eta \sum_{j=1}^{L} 2^j \ln(1+\frac{2^{L-j+1}N_{server}/P}{\eta}). \quad (12)$$

Therefore, the total transmission cost is:

$$\begin{aligned} C &= C_{suffix}^{\min} + C_{prefix} \\ &= \frac{2^L - 1}{2^L(L-1)+1}\eta \sum_{j=1}^{L} 2^j \ln(1+\frac{2^{L-j+1}N_{server}/P}{\eta}) \\ &\quad + \eta\beta P\ln(1+N_{proxy}/\eta). \end{aligned} \quad (13)$$

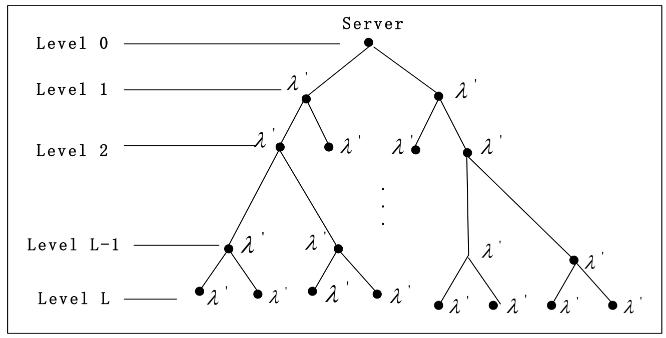

Fig. 2 Binary tree topology

Given different values of P, N/P and β, we change f in (7), (11), and (13) from zero to one respectively to find out at which f each cost reaches its minimum. The results are shown in Figs. 3 and 4 for $P=14$ and 62 respectively. The vertical axis stands for N/P, and the horizontal axis for β. There are three lines in each of these figures. Note that the cost model in [9] did not take the topology into account, and thus the results based on that model for different topologies can be shown in a single line. The left most line is based on the cost model in [9], the line in the middle is based on the binary tree topology, and the right most one is for the shared link with the fan-out topology. Each line divides the $(\beta, P/N)$ plane into two parts: the left side of the line is the $(\beta, P/N)$ region where proxy caching is cost effective; the right side of the line represents the $(\beta, P/N)$ region where proxy caching is not beneficial

We can see that under our cost model, β has a much larger range in which the prefix caching scheme is beneficial. In the case of $P = 14$, results from [9] indicate that prefix caching is feasible for large N/P provided that β is smaller than $1/P$ (<0.1), while our model allows β as large as 0.44. The difference is more significant in case of $P = 62$ as shown in Fig. 4: in our model, β can reach 0.27 and 0.5 for the binary tree topology and the shared link fan-out topology, respectively, while β is much smaller than 0.1 for large N/P using the cost model in [9]. Based on the above observations, we can claim that proxy caching is very attractive in saving the transmission cost.

IV. Multiple Videos Caching

In this section, we investigate the benefit of caching multiple videos with a limited proxy disk space. We assume there are 128 CBR video files of equal duration 120 minutes with the request probability drawn from the Zipf distribution with the skew factor $\theta = 0.271$ [12]. Our simulations focus on the heavy loaded server and proxy scenario, in which multicasting is more efficient than unicasting. At each proxy,

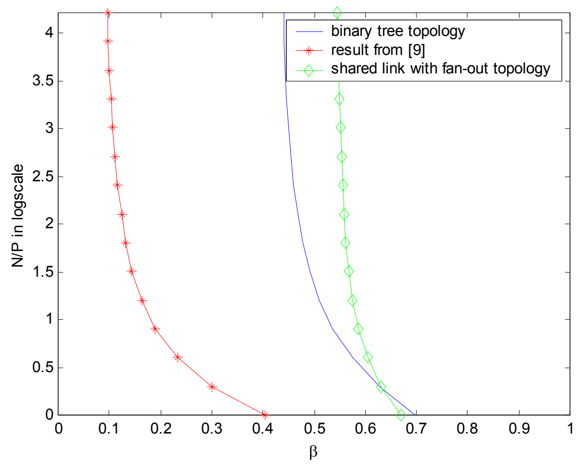

Fig. 3 Prefix caching for a single video file P=14.

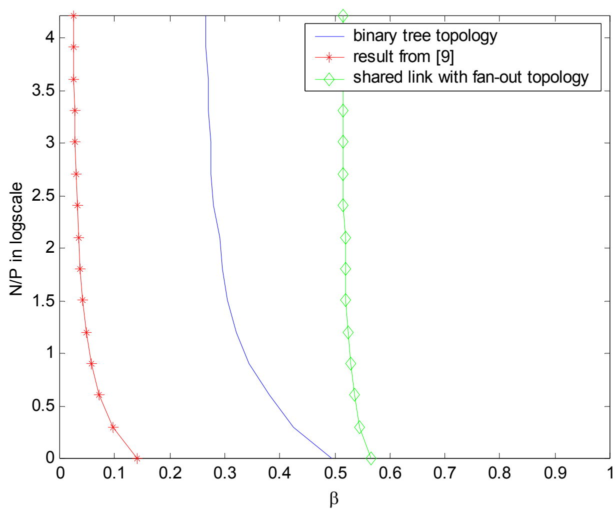

Fig. 4 Prefix caching for a single video file, P=62.

the total arrival rate K of requests for all videos is set to 10,000 per video length (120 minutes), meaning that there are about 80 requests per minute at each proxy.

By optimizing the stored fraction f for each video [8], we compute the transmission cost for the given values of β, K and P under different cost models presented in the previous section. The results are presented in Figs. 5, 6, 7 and 8. The horizontal axis stands for the proxy storage size in terms of the percentage of the total size of all 128 video files. The vertical axis represents the transmission cost normalized by the transmission cost without prefix caching.

Figs. 5 and 6 show the results for the shared link fan-out topology with $P = 14$ and $P = 62$, respectively. We can see that in both cases the transmission cost drops when the proxy cache size increases. If the proxy cache size is 50%, the transmission cost can be reduced to approximately 40% and 70% when $\beta = 0.1$ and $\beta = 0.3$, respectively. This is not possible under the model in [9], which claims that β should be smaller than $1/P$ for prefix caching to be beneficial. We also notice that there is very little difference between these two figures when P is changed from 14 to 62. This similarity is due to the fact that the feasible regions of the $(\beta, P/N)$ pair shown in Figs. 3 and 4 are almost identical in both cases.

Figs. 7 and 8 show the results for the binary tree topology with $P = 14$ and $P = 62$, respectively. We observe that the transmission can also be greatly reduced by caching prefix at the proxy. It should be noted that prefix caching is a little less beneficial in the binary tree topology than in the shared link fan-out topology. In particular, when $\beta = 0.3$ and $P = 62$, the transmission cost cannot be reduced too much by prefix caching. In most cases, proxy caching is very beneficial.

V. CONCLUSIONS

In this paper we have proposed a new cost model for proxy caching in multicasting networks. Our model considers the bandwidth usage over the entire network, and can reflect the network resource consumption more accurately. Based on the proposed cost model, we have investigated the feasibility and benefits of prefix caching of a single and multiple video files at proxies with a given limited proxy storage capacity. Our studies showed that prefix caching is very cost effective for both small and large P with a much larger range of β than previously thought [9]. An interesting research topic for the future is to investigate the effectiveness of proxy caching based on our proposed model in networks with more complex topologies.

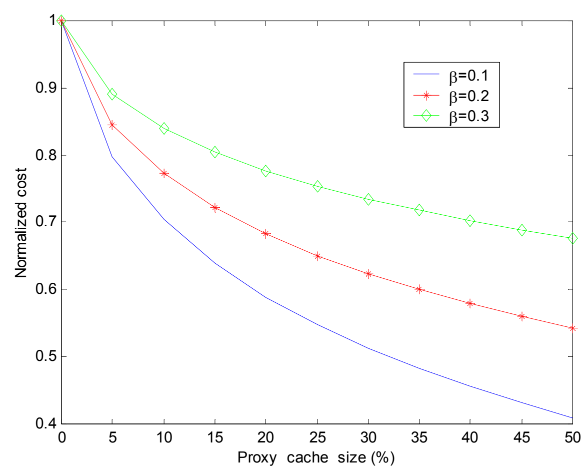

Fig. 5 Prefix caching for multiple video files; Shared link fan-out topology P=14, K=10,000.

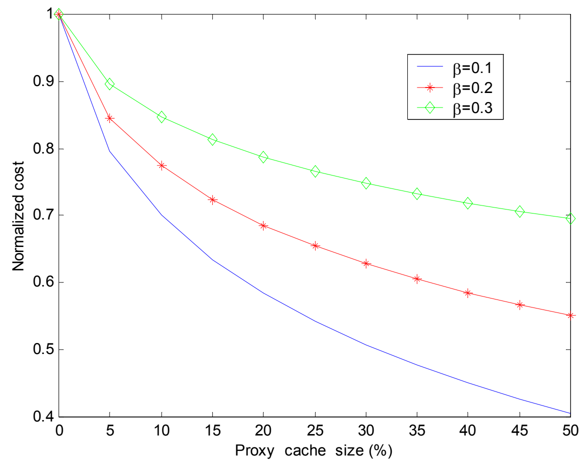

Fig. 6 Prefix caching for multiple video files; Shared link fan-out topology P=62, K=10,000.

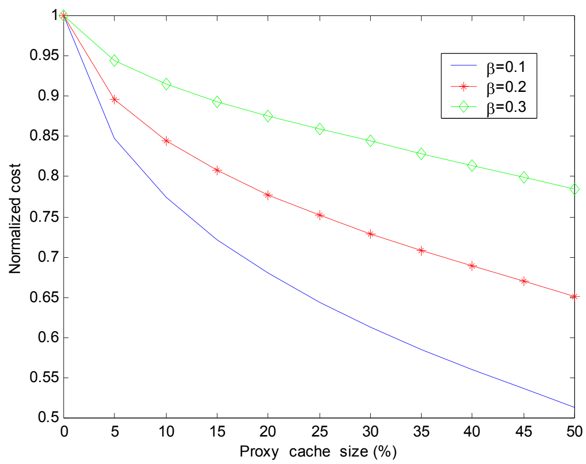

Fig. 7 Prefix caching for multiple video files; Binary tree topology P=14, K=10,000.

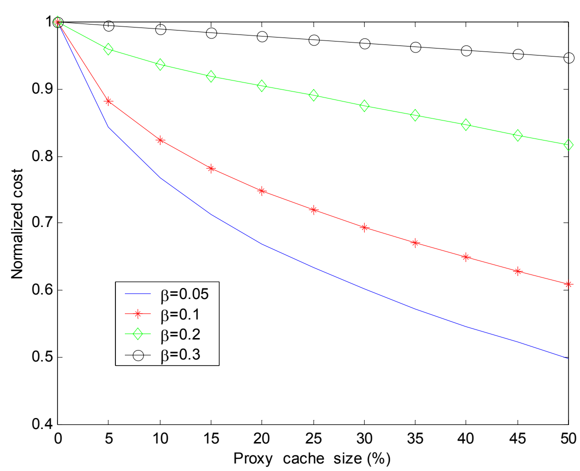

Fig. 8 Prefix caching for multiple video files; Binary tree topology P=62, K=10,000.

REFERENCES

[1] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling policies for an on-demand video server with batching," *Proc. of ACM Multimedia*, pp. 15-23, San Francisco, October 1994.

[2] Y. Cai, K. A. Hua and K. Vu, "Optimizing patching performance," *Proc. ACM/SPIE Multimedia Computing and Networking*, pp. 204-215, January 1999.

[3] K. A. Hua and S. Sheu, "Skyscraper broadcasting: a new broadcasting scheme for metropolitan VOD systems," *Proc. of the ACM/SIGCOMM'97*, pp. 89-100, Cannes, France, September 1997.

[4] D. Eager and M. Vermon, "Dynamic skyscraper broadcasts for video-on-demand," in *Proc. Inter Workshop on Network and Operating System Support for Digital Audio and Video*, July 1998.

[5] S. Sen, J. Rexford, and D. Towsley, "Proxy prefix caching for multimedia streams," in *Proc. IEEE Infocom*, vol. 3, pp. 1310–1319, 1999.

[6] Y. Wang, Z.-L. Zhang, D. Du, and D. Su, "A network conscious approach to end-to-end video delivery over wide area networks using proxy servers," in *Proc. IEEE Infoco*m, vol. 2, pp. 660–667, 1998.

[7] S.-H. Chan and F. Tobagi, "Distributed servers architecture for net-worked video services," *IEEE/ACM Trans. Networkin*g, vol. 9, pp. 125–136, Apr. 2001.

[8] B. Wang, S. Sen, M. Adler and D. Towsley, "Optimal proxy cache allocation for efficient streaming media distribution," *IEEE Infocom* 2002.

[9] J. M. Almeida, D. L. Eager, M. Ferris, and M. K. Vernon, "Provisioning Content Distribution Networks for Streaming Media," *IEEE Infocom*, 2002.

[10] D. L. Eager, M. K. Vernon and J. Zahorjan, "Optimal and efficient merging schedules for video-on-demand servers", *Proc. ACM Multimedia'*99, pp. 199-202, Orlando, FL, Nov. 1999.

[11] Y. Zhao, D. L. Eager, M. K. Vernon, "Network Bandwidth Requirements for Scalable On-Demand Streaming," *IEEE Infocom*, 2002.

[12] C. Aggarwal, J. Wolf, and P. Yu, "On optimal batching policies for video-on-demand storage servers," in *Proc. IEEE International Coference on Multimedia Computing and Systems*, pp. 253-258, June 1996.