

Using Multicast for Streaming Videos across Wide Area Networks

Bing Wang¹, Subhabrata Sen², Micah Adler¹ and Don Towsley¹

¹ Department of Computer Science

University of Massachusetts, Amherst, MA 01003

² AT&T Labs-Research, Florham Park, NJ 07928

Abstract—In this paper, we study streaming multiple videos from a remote server to asynchronous clients through a group of proxies, using multicast on both the wide area server-proxy paths and the local area proxy-client paths. In this setting, we present an algorithm to determine the optimal cache allocation among videos at each proxy and develop an efficient streaming video distribution scheme. Our evaluations show the benefits of even a small proxy cache and quantify the gains from using multicast on the server-proxy paths.

I. INTRODUCTION

A range of multimedia applications stand to benefit from technology for bandwidth-efficient and scalable video streaming. The high bandwidth requirements and the long-lived nature of digital videos make this medium particularly resource-intensive, stimulating research into server and network bandwidth-efficient distribution techniques.

Early techniques, such as *batching*, *patching* and *stream merging* [1], [2], [3], [4], [5], use multicast and broadcast connections in innovative ways to reduce server and network loads. More recent work [6], [7], [8], [9], [10], [11] extends the above techniques to streaming content distribution systems consisting of remote servers and proxies close to clients, where proxies cache some video content locally and assist in video streaming from the server to the clients. Much of the existing research has focused on optimizing the usage of the server bandwidth or the network bandwidth in a broadcast LAN environment. In either case, the bandwidth usage for multiple clients is the same as that for a single client.

There has been much less work on optimizing network bandwidth usage when streaming multiple videos to asynchronous clients in wide area Internet-like settings. Our previous work [10] explores the setting where the wide area server-proxy paths are only unicast capable while the local area proxy-client paths might be multicast capable. In this paper, we investigate the setting where the server has multicast connectivity to the proxies. Although the deployment of IP multicast in the Internet has been slow, it is being increasingly used within many corporate intranets. From a practical perspective, understanding the potential gains from using multicast on the wide area server-proxy paths can aid the development of appropriate architectures and techniques for video distribution in WAN settings.

The goal of this paper is to understand the benefits of using multicast instead of unicast on the server-proxy paths. The

multicast capability between server and proxies essentially couples the proxies together: a request by one proxy can initiate a data stream capable of serving multiple proxies. This coupling considerably complicates the problem of optimal proxy cache allocation. In this paper, we (i) present an algorithm to determine the optimal cache allocation at each proxy among different videos, and (ii) develop an efficient proxy-assisted transmission scheme when server-client paths are multicast capable. We then study the effect of proxy caching coupled with our proposed scheme on the overall network bandwidth usage.

Our evaluation demonstrates the benefits of even a small proxy cache and quantifies the gains from using multicast on the server-proxy paths. We propose two variations of the transmission scheme that differ in bandwidth efficiency. One requires proxies to dynamically join multicast groups; the other does not have this requirement and is less bandwidth efficient. However we observe that the differences in bandwidth usage in these two variations are small when the video request rates are high.

In related work, [11] studies a multicast distribution setting, focusing on server and proxy bandwidth usage but not on the network bandwidth usage. [12] studies both server and network bandwidth usage to distribute a single video using periodic broadcast by assuming the server-client paths form a m -ary tree. In contrast, we consider the problem of distributing multiple videos with varying popularities in a general Internet setting, where the wide area server-proxy paths can have a different distribution tree model than the local area proxy-client paths (see Section II).

The rest of the paper is organized as follows. Section II presents the problem setting and the cost model. Section III and IV present our optimal proxy cache allocation technique and an efficient transmission scheme respectively. Our evaluations are presented in Section V. Finally, Section VI concludes the paper.

II. PROBLEM SETTING AND MODEL

Consider a server and a set of proxies, where each proxy is responsible for a group of clients as shown in Fig. 1. We assume multicast is available on both the server-proxy and proxy-client paths. We further assume that clients always request playback from the beginning of a video. A proxy streams the prefix directly to its clients if a prefix of the video

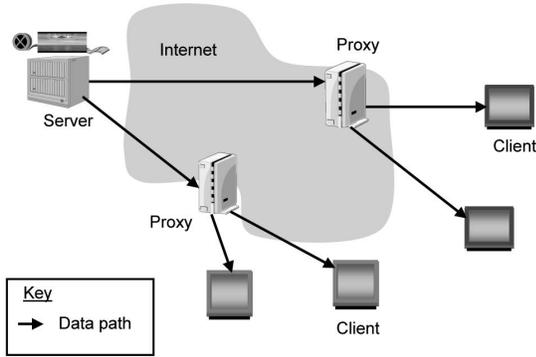


Fig. 1. **Streaming video in the Internet:** The video stream originates from a remote server and travels through the network to the end client. The proxies performing prefix caching are located close to the clients.

is present locally. If the video is not stored in its entirety at the proxy, the latter contacts the server for the remainder (suffix) of the stream. The server multicasts the required suffix to the proxies. A proxy further multicasts the suffix to a group of its clients that request the video.

A. System model

We next provide a formal model of the system, and introduce notation and key concepts, as presented in Table I. We use a superscript and subscript to represent the index of the proxy and the video respectively.

We consider a server with a repository of N Constant-Bit-Rate (CBR) videos and K proxies. We assume the access probabilities of all the videos and the aggregate access rate to the video repository at each proxy are known *a priori*. In a real system, these parameters can be obtained by monitoring the system. Without loss of generality, we order the videos in non-increasing order of their access probabilities. Let f_i^k be the access probability of video i at proxy k ; $\sum_{i=1}^N f_i^k = 1$. Let λ_i^k be the access rate of video i at proxy k and λ^k be the aggregate access rate to the video repository at proxy k ; $\lambda_i^k = \lambda^k f_i^k$. Let λ_i be the aggregate request arrival rate for video i at all the proxies; $\lambda_i = \sum_{k=1}^K \lambda_i^k$.

We introduce a *caching grain* of size u to be the smallest unit of cache allocation and all allocations are in multiples of this unit. The caching grain can be one bit or one minute of data, etc. We express the size of video i and the cache size at each proxy in multiples as the caching grain. Video i has playback bandwidth b_i bps, length L_i seconds, and size n_i units, $un_i = b_i L_i$. We assume that proxy k can store S_k units where $S_k \leq \sum_{i=1}^N n_i$. The *storage vector* $\vec{v}_i = (v_i^1, v_i^2, \dots, v_i^K)$ specifies that a prefix of length v_i^k seconds for each video i is cached at proxy k , $i = 1, 2, \dots, N$. Note that the videos cached at the proxy cannot exceed the storage constraint of the proxy, that is, $\sum_{i=1}^N b_i v_i^k \leq u S_k$.

On receiving a client request for a video, the proxy calculates a *transmission schedule* that depends on the transmission scheme in use. This transmission schedule specifies, for each video frame, when and on what *transmission channel* (unicast or multicast connection) it will be transmitted by the proxy.

The proxy also calculates a *reception schedule* for the clients that specifies which transmission channel the client should listen to in order to receive each frame. Note that a client may need to receive data from multiple transmission channels simultaneously. Under the transmission schemes we develop in Section IV, a client needs to receive from at most two channels simultaneously. This requirement is within the capacity of high bandwidth connections. Frames received ahead of their playback times are stored in a client-side workahead buffer. We assume the client has sufficient buffer space to accommodate an entire video. This assumption is justified by the disk space of most contemporary machines.

B. Cost model

We next describe the cost model. Let c_s and c_p respectively represent the costs associated with transmitting one bit of video data on a server-proxy path and on a proxy-client path using unicast. Our goal is to minimize the mean transmission cost per unit time aggregated over all videos in the repository, i.e., $\sum_{i=1}^N C_i(\vec{v}_i)$, where $C_i(\vec{v}_i)$ is the transmission cost per unit time for video i when the storage vector for video i is \vec{v}_i . In the rest of the paper, unless otherwise stated, we shall use the term *transmission cost* to refer to this metric.

Assuming a proxy and its clients are located in a LAN environment, the bandwidth required to send one bit from the proxy to multiple clients using multicast is still one bit. Therefore, the transmission cost to send one bit from the proxy to multiple clients is still c_p .

For server-proxy paths, we assume the cost to transmit a bit of data from the server to m proxies using multicast is $\beta m c_s$, where $\beta \in [1/m, 1)$ and is referred to as the *multicast scaling factor*. The minimum value for β is $1/m$, in which case the cost of transmitting a bit of data from the server to m proxies is c_s , similar to a LAN environment. Note that, in general, the cost when using multicast in a wide area network depends on a variety of factors including the multicast tree topology and the size of the multicast group [13], [14], [15], [16]. Since it is not yet understood what are realistic topologies for multicast trees in wide area Internet settings, we use a range of values for β instead of assuming a particular multicast tree topology in our performance evaluation (Section V).

Finally, note that when $c_p = 0$ and $\beta m c_s = 1$, the transmission cost reduces to be the server bandwidth usage. When $c_p = 1$ and $c_s = 0$, the transmission cost reduces to be the amount of outgoing traffic at the proxy.

III. OPTIMAL PROXY CACHE ALLOCATION

We next propose a general technique to determine the optimal proxy prefix cache allocation for any given proxy-assisted transmission scheme. Recall that a caching grain is the smallest unit of cache allocation (see Section II). The size of video i is n_i units and the cache size at proxy k is S_k units. Let $A_i = \{m_i \mid 0 \leq m_i \leq n_i\}$ denote the set of possible prefixes for video i , where m_i units is the size and $m_i u / b_i$ seconds is the length of a possible prefix of video i .

We define *saving*(\vec{m}_i), where $\vec{m}_i = (m_i^1, m_i^2, \dots, m_i^K)$, to be the saving in the transmission cost when storing m_i^k

Para.	Definition
N	Number of videos
L_i	Length of video i (sec.)
b_i	Mean bandwidth of video i (bits per sec.)
u	Caching grain
n_i	Size of video i (units)
K	Number of proxies
f_i^k	Access probability of video i at proxy k
λ_i^k	Request rate for video i at proxy k
λ^k	Aggregate request arrival rate for videos at proxy k
λ_i	Aggregate request arrival rate for video i at all proxies
S_k	The cache size (units) of proxy k
v_i^k	Length (sec) of cached prefix for video i at proxy k
\vec{v}_i	Storage vector of videos i , $\vec{v}_i = (v_i^1, v_i^2, \dots, v_i^K)$
c_s	Transmission cost on server-proxy path (per bit)
c_p	Transmission cost on proxy-client path (per bit)
β	The multicast scaling factor
$C_i(\vec{v}_i)$	Transmission cost per unit time for video i when the storage vector for video i is \vec{v}_i

TABLE I
PARAMETERS IN THE MODEL.

units of prefix of video i at proxy k over the cost when video i is not stored at the proxies, $k = 1, 2, \dots, K$. Our goal is to maximize the aggregate savings and, hence, minimize the aggregate transmission cost over all the videos. The optimization problem can therefore be formulated as

$$\begin{aligned} & \text{maximize: } \sum_{i=1}^N \text{saving}(\vec{m}_i) \\ & \text{s.t. } \sum_{i=1}^N m_i^k \leq S_k, m_i^k \in A_i, 1 \leq i \leq N, 1 \leq k \leq K \end{aligned}$$

Note that this formulation is a variant of the 0-1 knapsack problem, where the items to be placed into the knapsack are partitioned into sets and at most one item from each set is chosen. We next use the following dynamic programming algorithm to determine the optimal allocation.

Let B be a $(K+1)$ -dimensional matrix. An entry in B , $B(i, \vec{j})$, where $\vec{j} = (j_1, j_2, \dots, j_K)$, represents the maximum saving in the transmission cost when using the first i videos and j_k units of the storage are allocated at proxy k , $j_k \leq S_k$. When $i = 0$, $B(i, \vec{j}) = 0$. When $i > 0$,

$$B(i, \vec{j}) = \max_{\forall m_i^k \in A_i, k=1, 2, \dots, K} B(i-1, \vec{j} - \vec{m}_i) + \text{saving}(\vec{m}_i)$$

The value $B(N, \vec{S})$ is the maximum saving in transmission cost when all N videos have been used, where $\vec{S} = (S_1, S_2, \dots, S_K)$. The minimum transmission cost is $\sum_{i=1}^N C_i(\vec{0}) - B(N, \vec{S})$ since the saving is relative to storing nothing at the proxies. The execution time of the algorithm is $O(NS^KG)$, where $G = \max_{1 \leq i \leq N} |A_i|$ and $S = \max_{1 \leq k \leq K} S_k$. Note that the complexity of the algorithm is exponential with respect to the number of proxies K . In a longer version of this paper [17], we prove that, when K is not fixed, the problem is NP-hard in the strong sense. Therefore, we cannot hope for a more efficient algorithm.

When the server-proxy path is only unicast capable, determining the allocation for K proxies is equivalent to determining the allocation for each proxy separately. The above optimal allocation algorithm reduces to that we proposed in [10], which has the complexity of $O(NSKG)$.

Note that the above optimal allocation scheme assumes fixed parameters (including the access probabilities of the videos, the arrival rates of the videos, etc.). In practice, the parameters can be dynamic. Therefore the proxy cache allocation algorithm needs to be executed periodically to adapt to the changing parameters. Once an optimal proxy cache allocation is determined, the allocation is static, that is, the allocation is fixed until a new optimal proxy cache allocation is obtained.

IV. PROXY-ASSISTED TRANSMISSION SCHEME

In this section, we develop a transmission scheme MM-Patch, which is similar to patching [3], [4] in spirit but differs from patching in that it utilizes proxy prefix caching as an integral part for bandwidth-efficient delivery.

We next describe the MMPatch scheme in detail. When a client requests video i from its proxy, the proxy transmits the prefix stored locally to this client using multicast. At the same time, the proxy requests the suffix, the remainder of the video, from the server. Suppose proxy k issues the first request for the suffix of video i at time 0. Corresponding to the request, the server starts to transmit a suffix of length $L_i - v_i^k$ using multicast at time v_i^k . Proxy k multicasts the suffix from the server to a group of its clients that request the video. Later requests for video i (from proxy k or other proxies) have two options. They can start a new multicast suffix stream from the server. Or they join the ongoing multicast of the suffix and use separate unicast channels to obtain the missing data. Let T_i be a threshold to regulate the frequency at which the complete suffix stream is transmitted. If a later request arrives before T_i , it joins the ongoing multicast of the suffix. Otherwise, it starts a new complete suffix stream from the server. Threshold T_i is chosen such that the transmission cost is minimized.

We next present two variations of MMPatch: *dynamic* and *baseline* MMPatch. In dynamic MMPatch, when one proxy initiates a multicast suffix stream from the server, another proxy joins the multicast group to receive the stream only when at least one of its clients request the video. In baseline MMPatch, all the proxies receive the multicast stream; then each proxy either forwards this stream to its clients or discards it upon receipt when there is no need for the stream. Clearly dynamic MMPatch is more bandwidth efficient while requires more signaling than baseline MMPatch. In baseline MMPatch, the server can transmit all multicast streams using one channel (multicast connection) and all the proxies listen to that channel. In dynamic MMPatch, a multicast group has to be created for each multicast stream from the server and the proxies need to join multicast groups dynamically. We derive the average number of joins from all the proxies per unit of time in dynamic MMPatch and the overall transmission cost function for both baseline and dynamic MMPatch in [17].

When there is no proxy caching (i.e., no video prefix at the proxies and the proxies are used only as gateways), MMPatch

reduces to threshold-based patching [4]. When the server-proxy path is only unicast capable, the transmission from the server corresponding to the requests from one proxy is independent of that to other proxies. In this case, MMPatch reduces to MPatch, which we proposed in [10].

V. PERFORMANCE EVALUATION

In this section, we examine the resource tradeoffs for MMPatch under the optimal proxy cache allocation. We consider a repository of 100 CBR video clips with access probabilities drawn from a Zipf distribution with parameter $\theta = 0.271$ [1]. We also use more skewed ($\theta = 0$) and less skewed ($\theta = 0.5$) distributions. We only describe the results under $\theta = 0.271$ in detail; the performance trends under different values of θ are similar. For simplicity, we assume all the videos are two hours long, and have the same bandwidth. We normalize the transmission cost by both the video bandwidth and the value of c_s . That is, the normalized transmission cost is $\sum_{i=1}^N C_i(v_i)/(c_s b_i)$. Let $\hat{c}_p = c_p/c_s$. We assume $\hat{c}_p \in [0, 1]$. Observe that $\hat{c}_p = 0$ corresponds to $c_p = 0$ and $\hat{c}_p = 1$ corresponds to $c_p = c_s$. We represent the proxy cache size as a percentage, r , of the size of the video repository. We use one minute of data as the caching grain for the proxy cache allocation.

We first consider homogeneous proxies, that is, the configurations for all the proxies are the same, including proxy cache size, video access probabilities, aggregate arrival rate, c_s , c_p , etc. At the end of this section, we consider proxies that are heterogeneous in the arrival rate of the videos. We assume requests to a proxy follow a Poisson process and the aggregate arrival rate ranges from 10 to 500 requests per minute. The total number of proxies K is set to be 10 or 100. The multicast scaling factor β ranges in $[1/K, 1)$. The performance trends under $K = 10$ and $K = 100$ are similar. We therefore only report the results under $K = 100$.

We believe that the optimal allocation at homogeneous proxies should be the same, since the proxies are not distinguishable in contributing to the transmission cost. We are unable to provide a rigorous proof but our evaluation on two homogeneous proxies confirms this conjecture. Running the optimal proxy cache allocation algorithm on large number of proxies is difficult because of the complexity and the space requirement. In the following, the optimal allocation we describe is under the assumption that the allocation at homogeneous proxies is the same. We first investigate the effect of proxy caching on the transmission cost. We then describe the optimal proxy allocation across the videos and the gains from using multicast on the server-proxy paths over using unicast. All of the above are based on baseline MMPatch. Finally, we compare the performance of baseline and dynamic MMPatch.

A. The effect of proxy caching on the transmission cost

Proxy caching leads to lower network transmission cost in all the settings we study. This is expected since data from the server pass the proxies and hence transmitting a stream directly from a proxy incurs less cost than from the server. On the other

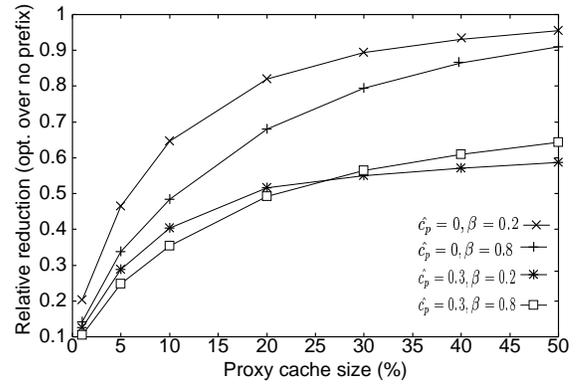


Fig. 2. Relative reduction under optimal proxy cache allocation over no prefix caching when $\lambda^k = 50/\text{min}$.

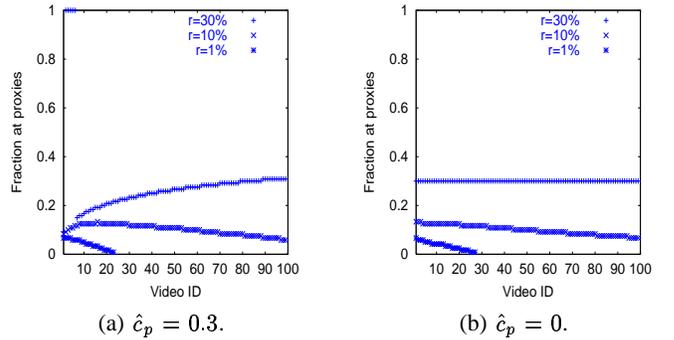


Fig. 3. Proxy cache allocation under optimal prefix caching when $\lambda^k = 50/\text{min}$ and $\beta = 0.2$.

hand, this is in contrast to the study in [11], which focuses on server and proxy bandwidth usages instead of network bandwidth usage and shows that proxy caching only reduces server and proxy bandwidth usages in some settings.

We define the *relative reduction* under optimal proxy cache allocation over no proxy caching to be the difference in the costs under these two settings divided by the cost without proxy caching. Fig. 2 plots the relative reduction thus defined when the aggregate arrival rate to a proxy is 50 requests per minute and β ranges from 0.2 to 0.8. We observe that a relatively small proxy cache (1%-10% of the video repository) is sufficient to realize substantial savings in transmission cost and the proxy cache size has a diminishing effect on the cost savings. Furthermore, the reduction is more dramatic for $\hat{c}_p = 0$ than for $\hat{c}_p = 0.3$. This is because, for lower values of \hat{c}_p , the savings from transmitting directly from the proxy cache to the clients is more dramatic. Finally, we observe similar characteristics for other request arrival rates.

B. Optimal proxy cache allocation across the videos

Fig. 3(a) depicts the optimal proxy cache allocation in MMPatch when $\lambda^k = 50/\text{min}$, $\beta = 0.2$ and $\hat{c}_p = 0.3$. We find that the size of the proxy cache allocated to a video is not a monotonically increasing function of the access probability. This is because the threshold tends to increase as the access probability decreases. Therefore some less popular videos may require larger prefixes than more popular videos to realize the

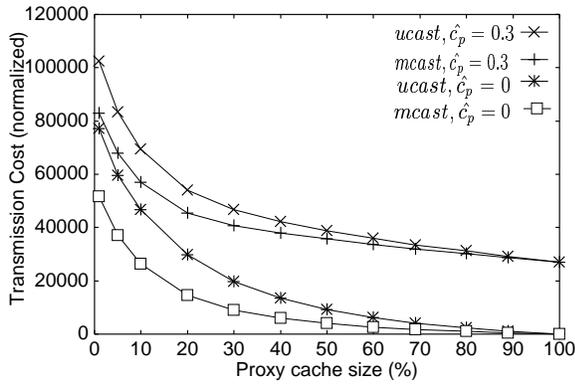


Fig. 4. Transmission cost (normalized) when using multicast ($\beta = 0.4$) and unicast on the server-proxy paths, $\lambda^k = 50/\text{min}$.

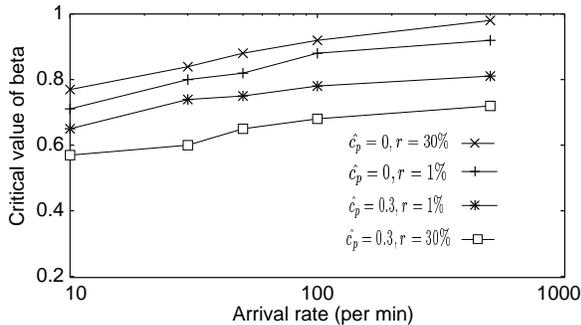


Fig. 5. Critical values of β as a function of arrival rate λ^k .

optimal threshold. When $\hat{c}_p = 0$ (see Fig. 3(b)), the proxy cache is more evenly distributed among the videos. For a value of β higher than 0.2, the optimal allocation across the videos is more skewed (figures not shown), closer to the allocation when using unicast on the server-proxy paths.

C. Benefits of using multicast along server-proxy paths

When using unicast on the server-proxy paths, MMPatch reduces to MPatch proposed in [10]. Fig. 4 shows the transmission cost when using multicast and unicast on the server-proxy paths as a function of the proxy cache size for $\lambda^k = 50/\text{min}$ and $\beta = 0.4$. Using multicast on the server-proxy paths leads to significant savings only for small and medium proxy cache sizes. When the proxy cache size is 1% of the video repository (i.e., $r = 1\%$), using multicast on the server-proxy paths reduces the transmission cost by 32% and 19% over that using unicast for $\hat{c}_p = 0$ and $\hat{c}_p = 0.3$ respectively. As the proxy cache increases, more contents are transmitted from the proxy cache directly and the cost on the server-proxy paths becomes less dominant in the total cost. That explains why transmission costs using multicast and unicast on the server-proxy paths become close for large proxy cache sizes. We observe that the cost savings from using multicast on server-proxy paths are more significant for small values of \hat{c}_p . This is because, in that case, the cost on the server-proxy paths is more dominant in the total cost and the benefits of using multicast are more manifest. We also observe that the cost savings from using multicast on the server-proxy paths increase with the arrival

rate, especially for small and medium proxy cache sizes (figure not shown). For instance, when λ^k increases from 50/min to 500/min, for $r = 1\%$ and $\beta = 0.4$, the cost saving from using multicast on the server-proxy paths increases from 32% to 40% for $\hat{c}_p = 0$ and from 19% to 24% for $\hat{c}_p = 0.3$.

In the above, we fixed β to be 0.4. As β increases, the cost of using multicast on the server-proxy paths approaches and then surpasses that of using unicast. This is because, in MMPatch, using multicast on the server-proxy paths becomes less bandwidth efficient for higher values of β . We refer to the value of β at which the costs of using multicast and unicast on the server-proxy paths are the same as the *critical value* of β . Fig. 5 depicts the critical value of β as a function of the arrival rate λ^k , \hat{c}_p and r . We observe that the critical value increases as the arrival rate increases since, as observed before, the savings from using multicast on server-proxy paths increase with the arrival rate. The critical value is higher for lower values of \hat{c}_p , since the cost on server-proxy paths is more dominant in the total cost when \hat{c}_p is low. When $\hat{c}_p = 0$, the critical value of β is higher for larger values of r ; when $\hat{c}_p > 0$, the critical value of β is higher for smaller values of r . This is because the gains from using multicast on the server-proxy paths is the highest at medium and small proxy cache sizes for $\hat{c}_p = 0$ and $\hat{c}_p > 0$ respectively.

D. Performance comparison of baseline and dynamic MM-Patch

We now assume the multicast bandwidth usage follows Chuang-Sirbu law [13], which states that the cost of using multicast for a multicast group size of m is $m^{0.8}$ times as that using unicast. In dynamic MMPatch, the size of a multicast group ranges from 1 to K . In baseline MMPatch, the size of the multicast group is K and the cost of transmitting one bit of data from the server to all the proxies using unicast is Kc_s . Therefore, $\beta = K^{0.8}/K$ in baseline MMPatch under the assumption of Chuang-Sirbu law. That is, $\beta \approx 0.4$ for $K = 100$. We found the performance of dynamic and baseline MMPatch are very similar except for low arrival rate of 10 requests per minute, where the difference in their transmission costs is within 10%. This is expected since in dynamic MMPatch, the size of the multicast group for a stream with high arrival rate is close to the total number of proxies K , in which case the transmission costs under dynamic and baseline MMPatch are similar. However, dynamic MMPatch requires much more dynamic joins than baseline MMPatch. For instance, when $\lambda^k = 10/\text{min}$, the average number of joins per minute from all proxies in dynamic MMPatch is around 30000 (figure not shown). For higher arrival rates, the average number of joins per minute is even higher since the threshold tends to decrease as the request arrival rate increases.

E. Heterogeneous proxies

We next examine the resource tradeoffs for a special case of heterogeneous proxies: the proxies are heterogeneous in the sense that the aggregate arrival rates to the proxies are different. We assume proxies belong to two classes: one class with an aggregate arrival rate of 50 requests per minute and the

other one with 500 requests per minute, 10 times larger than the first class. The number of proxies in each class is the same. We observe that the allocations for the two classes of proxies are the same. Furthermore, the allocation is very close to that for homogeneous proxies with the same average arrival rate, which is $(500 + 50)/2 = 275$ requests per minute. The reason might be that the cost on server-proxy paths is more sensitive to the aggregate arrival rate from all the proxies instead of the arrival rate from each individual proxy.

VI. CONCLUSIONS

In this paper, we study video streaming in the setting where both the server-proxy and proxy-client paths are multicast capable. We present an algorithm to determine the optimal cache allocation at each proxy and develop an efficient proxy-assisted transmission scheme. Our performance evaluation shows that: (i) under optimal prefix caching, a relatively small proxy cache (1%-10% of the video repository) is sufficient to realize substantial savings in transmission cost; (ii) using multicast on server-proxy path leads to significant savings over using unicast for small to medium proxy cache sizes and high arrival rates; In the other cases, using unicast is good enough; (iii) the simpler baseline MMPatch is as bandwidth efficient as dynamic MMPatch for relatively high arrival rates.

ACKNOWLEDGMENTS

This work was supported in part by NSF Research Infrastructure Award EIA-0080119, NSF ANI-9973092, NSF ANI-9977635 and NSF Faculty Early Career Development Award CCR-0133664. The authors would like to thank Shudong Jin for helpful discussions and the anonymous reviewers for their insightful comments.

REFERENCES

- [1] C. Aggarwal, J. Wolf, and P. Yu, "On optimal batching policies for video-on-demand storage servers," in *Proc. IEEE International Conference on Multimedia Computing and Systems*, June 1996.
- [2] L. Golubchik, J. Lui, and R. Muntz, "Adaptive piggybacking: A novel technique for data sharing in video-on-demand storage servers," *ACM Multimedia Systems Journal*, vol. 4, no. 3, 1996.
- [3] K. Hua, Y. Cai, and S. Sheu, "Patching: A multicast technique for true video-on-demand services," in *Proc. ACM Multimedia*, September 1998.
- [4] L. Gao and D. Towsley, "Threshold-based multicast for continuous media delivery," *IEEE Transactions on Multimedia*, vol. 3, pp. 405–414, December 2001.
- [5] D. Eager, M. Vernon, and J. Zahorjan, "Optimal and efficient merging schedules for video-on-demand servers," in *Proc. ACM Multimedia*, November 1999.
- [6] L. Gao, Z. Zhang, and D. Towsley, "Catching and selective catching: Efficient latency reduction techniques for delivering continuous multimedia streams," in *Proc. ACM Multimedia*, 1999.
- [7] S. Ramesh, I. Rhee, and K. Guo, "Multicast with cache (mcache): An adaptive zero-delay video-on-demand service," in *Proc. IEEE INFOCOM*, April 2001.
- [8] D. Eager, M. Ferris, and M. Vernon, "Optimized regional caching for on-demand data delivery," in *Proc. Multimedia Computing and Networking (MMCN '99)*, January 1999.
- [9] O. Verscheure, C. Venkatramani, P. Frossard, and L. Amini, "Joint server scheduling and proxy caching for video delivery," in *Proc. 6th International Workshop on Web Caching and Content Distribution*, June 2001.
- [10] B. Wang, S. Sen, M. Adler, and D. Towsley, "Optimal proxy cache allocation for efficient streaming media distribution," in *Proc. IEEE INFOCOM*, 2002.

- [11] J. Almeida, D. L. Eager, M. C. Ferris, and M. K. Vernon, "Provisioning content distribution networks for streaming media," in *Proc. IEEE INFOCOM*, 2002.
- [12] D. Choi, E. W. Biersack, and G. Urvoy-Keller, "Cost-optimal dimensioning of a large scale video on demand system," in *Proceedings of NGC*, 2002.
- [13] J. C.-I. Chuang and M. A. Sirbu, "Pricing multicast communication: A cost-based approach," *Telecommunication Systems*, vol. 17, no. 3, pp. 281–297, 2001.
- [14] G. Phillips, S. Shenker, and H. Tangmunarunkit, "Scaling of multicast trees: Comments on the chuang-sirbu scaling law," in *SIGCOMM*, pp. 41–51, 1999.
- [15] R. C. Chalmers and K. C. Almeroth, "Modeling the branching characteristics and efficiency gains in global multicast trees," in *Proc. IEEE INFOCOM*, 2001.
- [16] C. Adjih, L. Georgiadis, and P. Jacquet, "Multicast tree structure and the power law," in *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2002.
- [17] B. Wang, S. Sen, M. Adler, and D. Towsley, "Using multicast for streaming videos across wide area networks," Tech. Rep. 03-28, Department of Computer Science, University of Massachusetts, Amherst, 2003.