# Using Multicast for Streaming Videos across Wide Area Networks

Bing Wang[1], Subhabrata Sen[2], Micah Adler[1] and Don Towsley[1]
[1] Department of Computer Science
University of Massachusetts, Amherst, MA 01003
[2] AT&T Labs-Research, Florham Park, NJ 07928
UMass CMPSCI Tech. Report 03-28

**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.

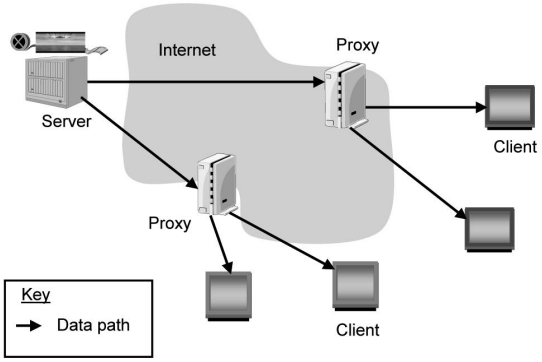



Fig. 1. **Streaming video in the Internet:** The video stream originates from a remote server and travels through the network to the end client. The proxies performing prefix caching are located close to the clients.

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 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 $\lambda_i^k$ be the access rate of video $i$ at proxy $k$ and $\lambda^k$ be the aggregate access rate to the video repository at proxy $k$; $\lambda_i^k = \lambda^k f_i^k$. Let $\lambda_i$ 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* $\vec{v_i} = (v_i^1, v_i^2, \ldots, v_i^K)$ specifies that a prefix of length $v_i^k$ seconds for each video $i$ is cached at proxy $k$, $i = 1, 2, \ldots, N$. 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 $\beta$ 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 $\beta$ 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 $\vec{m_i} = (m_i^1, m_i^2, \ldots, m_i^K)$, to be the saving in the transmission cost when storing $m_i^k$ units of prefix of video $i$ at proxy $k$ over the cost when video $i$ is not stored at the proxies, $k = 1, 2, \ldots, K$.

| 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 $k$ |
| $\lambda_i^k$ | Request rate for video $i$ at proxy $k$ |
| $\lambda^k$ | Aggregate request arrival rate for videos at proxy $k$ |
| $\lambda_i$ | 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 $i$, $\vec{v_i} = (v_i^1, v_i^2, \ldots, v_i^K)$ |
| $c_s$ | Transmission cost on server-proxy path (per bit) |
| $c_p$ | Transmission cost on proxy-client path (per bit) |
| $\beta$ | 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.

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

$$\text{maximize:} \sum_{i=1}^{N} saving(\vec{m_i})$$

$$\text{s.t.} \sum_{i=1}^{N} m_i^k \leq S_k, m_i^k \in A_i, 1 \leq i \leq N, 1 \leq k \leq K$$

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 $\vec{j} = (j_1, j_2, \ldots, j_K)$, 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 $i > 0$,

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

The value $B(N, \vec{S})$ is the maximum saving in transmission cost when all $N$ videos have been used, where $\vec{S} = (S_1, S_2, \ldots, S_K)$. 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 $O(NS^K G)$, 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 the appendix, 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 MMPatch, 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 the appendix.

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 $\theta$ 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 $\sum_{i=1}^{N} C_i(\vec{v_i})/(c_s b_i)$. Let

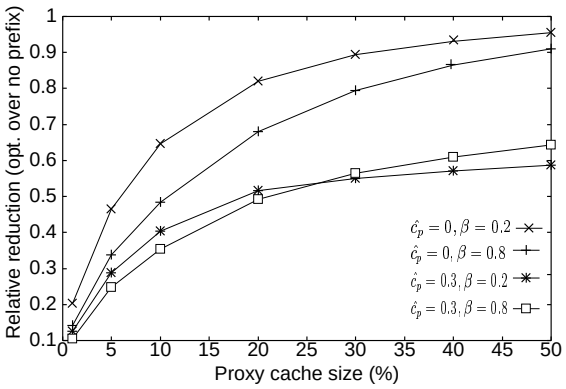



Fig. 2. Relative reduction under optimal proxy cache allocation over no prefix caching when $\lambda^k = 50$/min.

$\hat{c_p} = c_p/c_s$. 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.

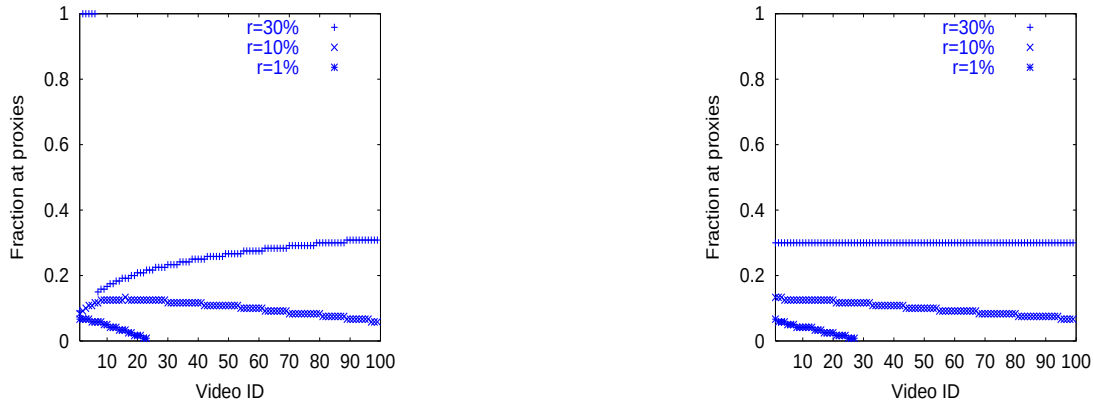
We first consider homogeneous proxies, that is, the configurations for all the proxies are the same, including proxy cache size, video access probabilities, aggregate arrival rate, $c_s$, $c_p$, etc. At the end of this section, we consider proxies that are heterogeneous in the arrival rate of the videos. 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 $\beta$ 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 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 $\beta$ 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.




(a) $\hat{c}_p = 0.3$. (b) $\hat{c}_p = 0$.

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

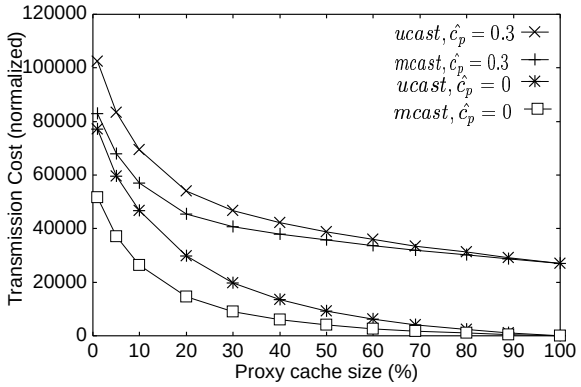



Fig. 4. Transmission cost (normalized) when using multicast ($\beta = 0.4$) and unicast on the server-proxy paths, $\lambda^k = 50$/min.

### *B. Optimal proxy cache allocation across the videos*

Fig. 3(a) depicts the optimal proxy cache allocation in MMPatch when $\lambda^k = 50$/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 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 $\beta$ 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 $\lambda^k = 50$/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

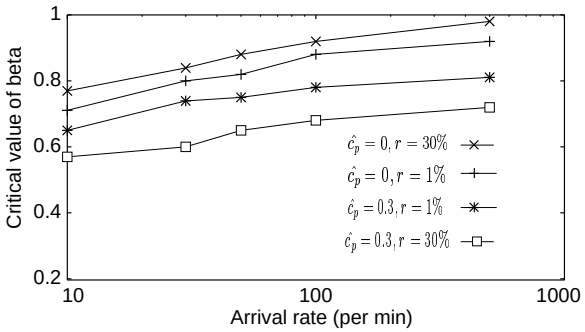



Fig. 5. Critical values of $\beta$ as a function of arrival rate $\lambda^k$.

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 $\lambda^k$ 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 $\beta$ to be $0.4$. As $\beta$ 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 $\beta$. We refer to the value of $\beta$ at which the costs of using multicast and unicast on the server-proxy paths are the same as the *critical value* of $\beta$. Fig. 5 depicts the critical value of $\beta$ as a function of the arrival rate $\lambda^k$, $\hat{c_p}$ 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 $\beta$ is higher for larger values of $r$; when $\hat{c_p} > 0$, the critical value of $\beta$ 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 MMPatch*

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 $\lambda^k = 10$/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. $6^{th}$ 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.

## APPENDIX

### A. Derivation of cost functions for MMPatch

We derive the overall transmission cost for MMPatch by modeling the system as a renewal process. We consider the interval between the initiation of two multicast suffix streams from the server. We assume Poisson arrival process throughout the derivation.

Suppose proxy $k$ issues the first request for the suffix of video $i$ in the renewal process. Let $E_i^k(v_i^k)$ be the transmission cost for video $i$ from proxy $k$. Suppose the first request arrives at time 0. Corresponding to this request, the server multicasts a suffix of length $L_i - v_i^k$ at time $v_i^k$. Let $D_i^k(v_i^k)$ be the cost for the suffix stream multicasted from the server. The difference between baseline and dynamic MMPatch lies only in $D_i^k(v_i^k)$. In baseline MMPatch, we have $D_i^k(v_i^k) = \beta K c_s(L_i - v_i^k)$ since all the proxies are in the multicast group. We next derive $D_i^k(v_i^k)$ for dynamic MMPatch which is classified into two cases depending on the relationship of $v_i^k$ and $T_i$. Suppose the bandwidth usage for a multicast group of size $m$ is $d(m)$.

- Case 1: $v_i^k \geq T_i$. In this case, the suffix stream of length $(L_i - v_i^k)$ is transmitted to all the proxies that join the multicast group by $T_i$. Therefore

$$D_i^k(v_i^k) = c_s(L_i - v_i^k) \sum_{m=1}^{K-1} d(1+m)p(m, T_i)$$

  where function $p(m,t)$ is the probability that $m$ out of the other $K-1$ proxies join in the multicast group by time $t$. We only derive $p(m,t)$ for homogeneous proxies. The probability that proxy $g (g \neq k)$ has requests for video $i$ by time $t$ is $(1 - e^{-\lambda_i^1 t})$. Therefore, for homogeneous proxies, we have

$$p(m,t) = \binom{K-1}{m} (1 - e^{-\lambda_i^1 t})^m (e^{-\lambda_i^1 t})^{K-1-m}$$

- Case 2: $v_i^k < T_i$. In this case, we divide the interval $(v_i^k, T_i]$ into subintervals of length $\delta$. Then $D_i^k(v_i^k)$ can be approximated as

$$\begin{aligned} D_i^k(v_i^k) = \quad & c_s \delta \textstyle\sum_{n=1}^{(T_i - v_i^k)/\delta} \sum_{m=1}^{K-1} d(1+m) p(m, v_i^k + n\delta) \\ & + c_s(L - T_i) \textstyle\sum_{m=1}^{K-1} d(1+m) p(m, T_i) \end{aligned}$$

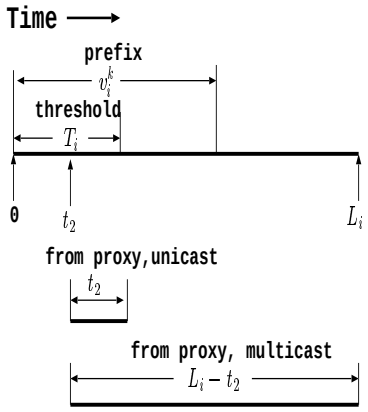
  The first term corresponds to the cost for the arrivals between $v_i^k$ and $T_i$. The second term corresponds to the cost after time $T_i$.

We next derive $E_i^k(v_i^k)$. Suppose a later request to proxy $k$ for video $i$ arrives at time $t_2$ , $0 < t_2 \leq T_i$. Video delivery for this client can be classified into the following two cases depending on the relationship of $v_i^k$ and $T_i$.

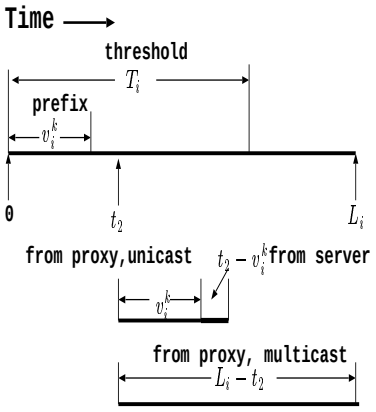
- Case 1: $v_i^k \geq T_i$. This is shown in Fig. 6 (a). The client receives segment $[0, t_2]$ from a separate channel via unicast from the proxy and segment $(t_2, L_i]$ via the ongoing multicast stream. Assuming a Poisson arrival process, we have

$$E_i^k(v_i^k) = D_i^k(v_i^k) + L_i c_p + \frac{\lambda_i^k T_i^2}{2} c_p$$

  In this case, the average total length of patches in the renewal process is $\lambda_i^k {T_i}^2/2$ as shown in [4].




(a) Case 1: $T_i \leq v_i^k \leq L_i$.




(b) Case 2: $0 \leq v_i^k < T_i$.

Fig. 6. Delivery to proxy $k$, which initiates the transmission of suffix from the server using multicast in MMPatch.

- Case 2: $v_i^k < T_i$. This is shown in Fig. 6 (b). If $0 < t_2 \leq v_i$, then the transmission mechanism is the same as in Case 1. If $v_i < t_2 \leq T_i$, the client receives segment $[0, v_i]$ from a separate channel via unicast from the proxy and receives segment $(t_2, L_i]$ via the ongoing multicast stream from the proxy. Segment $(v_i, t_2]$ is transmitted from the server to the client via the proxy using unicast. Assuming a Poisson arrival process, we have

$$\begin{aligned} E_i^k(v_i^k) = \quad & D_i^k(v_i^k) + L_i c_p + \tfrac{\lambda_i^k {v_i^k}^2}{2} c_p \\ & + \tfrac{\lambda_i^k (T_i^2 - {v_i^k}^2)}{2} (c_s + c_p) \end{aligned}$$

  In this case, the average total length of patches in the renewal process from proxy $k$ is $\lambda_i^k {v_i^k}^2/2$. The average total length of patches from the server is $\lambda_i (T_i - v_i^k)^2/2$. This is because the average number of arrivals in this time interval is $\lambda_i (T_i - v_i^k)$ with average length of patch of $(T_i - v_i^k)/2$.

We next consider later requests from another proxy $g, g \neq k$. Let $F_i^g(v_i^g)$ be the transmission cost for video $i$ from this proxy. It contains the costs for the complete stream (corresponding to the first request to proxy $g$) and the partial streams (corresponding to later requests to proxy $g$). Let $G_i^g(v_i^g)$ and $H_i^g(v_i^g)$ denote the two costs respectively. Then

$$F_i^g(v_i^g) = G_i^g(v_i^g) + H_i^g(v_i^g)$$

We next derive $G_i^g(v_i^g)$ and $H_i^g(v_i^g)$. Let $f(x)$ be the pdf of the arrival time of the first request for video $i$ at proxy $g$. Then by the assumption of Poisson arrival

$$f(x) = \lambda_i^g e^{-\lambda_i^g x}$$

Suppose the first request for video $i$ to proxy $g$ is at time $x$, $x \leq T_i$. The derivation of $G_i^g(v_i^g)$ is obtained by considering two cases classified by the relationship of $v_i^k$ and $T_i$.

- Case 1: $v_i^k \geq T_i$. In this case, if $v_i^k > v_i^g$, the suffix from the server (initiated by proxy $k$) is not sufficient for proxy $g$ and the video segment of $[v_i^g, v_i^k]$ has to be transmitted from the server to the proxy, leading to a cost of $(v_i^k - v_i^g)c_s$. Let $(v_i^k - v_i^g)^+ = \max(v_i^k - v_i^g, 0)$. The probability of having at least one arrival at proxy $g$ before $T_i$ is $(1 - e^{-\lambda_i^g T_i})$. Then

$$G_i^g(v_i^g) = [L_i c_p + (v_i^k - v_i^g)^+ c_s](1 - e^{-\lambda_i^g T_i})$$

- Case 2: $v_i^k < T_i$. If $x \le v_i^k$, then $G_i^g(v_i^g)$ is the same as in Case 1. If $x > v_i^k$, the length of the patch required from the server is $(x - v_i^g)$. Hence

$$\begin{aligned} G_i^g(v_i^g) &= \int_0^{v_i^k} [L_i c_p + (v_i^k - v_i^g)^+ c_s] dx \\ &+ \int_{v_i^k}^{T_i} f(x)[L_i c_p + (x - v_i^g)^+ c_s] dx \end{aligned}$$

We next derive the cost for partial streams $H_i^g(v_i^g)$, which also depends on the relationship of $v_i^g$ and $T_i$ as follows.

- Case 1: $v_i^g \ge T_i$. The average length of partial stream requested from the proxy is $(T_i + x)/2$ since the first request is at time $x$. On average, there are $(T_i - x)\lambda_i^g$ requests from the proxy. Therefore

$$H_i^g(v_i^g) = \int_0^{T_i} f(x) \frac{\lambda_i^g (T_i^2 - x^2)}{2} c_p dx$$

- Case 2: $v_i^g < T_i$, we have

$$\begin{aligned} &H_i^g(v_i^g) = \int_0^{v_i^g} f(x) \tfrac{\lambda_i^g (v_i^{g\,2} - x^2)}{2} c_p dx \\ &+ \lambda_i^g (1 - e^{-v_i^g \lambda_i^g})(T_i - v_i^g)[\tfrac{T_i + v_i^g}{2}(c_s + c_p) - v_i^g c_s] \\ &+ \int_{v_i^g}^{T_i} f(x) \lambda_i^g (T_i - x)[\tfrac{T_i + x}{2}(c_s + c_p) - v_i^g c_s] dx \end{aligned}$$

  This can be explained as follows. If $x > v_i^g$, the average number of requests is $\lambda_i^g(T_i - x)$ before the threshold. The average video length required is $(T_i + x)/2$ and the prefix $v_i^g$ is transmitted directly from the proxy. Therefore the average cost for one request is $[\frac{T_i + x}{2}(c_s + c_p) - v_i^g c_s]$, which is the factor in the third term. If $x \le v_i^g$, the cost is the sum of the first two terms. For any request before $v_i^g$, the partial stream missed by the client belongs to the prefix and can be served directly by the proxy, leading to a cost represented by the first term. For requests arriving after $v_i^g$, the cost is represented by the second term for similar reasons as explained for the third term.

Let $N_i(T_i)$ be the number of arrivals in the interval between the initiation of two multicast streams of video $i$ from the server in the renewal process. It can be approximated by $1 + \lambda_i T_i$. The exact expression is

$$\begin{aligned} N_i(T_i) &= 1 + \lambda_i^k T_i \\ &+ \textstyle\sum_{g=1, g \ne k}^{K} (1 - e^{-\lambda_i^g}) \\ &+ \textstyle\sum_{g=1, g \ne k}^{K} \int_0^{T_i} f(x) \lambda_i^g (T_i - x) dx \end{aligned}$$

Where $1 + \lambda_i^k T_i$ is the number of requests from proxy $k$. For proxy $g$, the probability of having at least one request before the threshold is $(1 - e^{-\lambda_i^g})$. If the first client comes at time $x$ $(x \le T_i)$, the total number of arrivals from proxy $g$ is $\lambda_i^g (T_i - x)$.

The overall transmission cost per unit of time for video $i$ is:

$$C_i(\vec{v_i}) = \sum_{k=1}^{K} \frac{\lambda_i b_i}{N_i(T_i)} \frac{\lambda_i^k}{\lambda_i} [E_i^k(v_i^k) + \sum_{g=1, g \ne k}^{K} F_i^g(v_i^g)]$$

This can be explained as follows. The probability that proxy $k$ issues the first request is $\lambda_i^k / \lambda_i$ and the total cost contains the cost from proxy $k$ and all the other proxies with later coming requests.

We next derive the average number of joins from all the proxies per unit of time in dynamic MMPatch. The average length of the interval between the initiation of two multicast streams of video $i$ from the server is $T_i + 1/\sum_{k=1}^{K} \lambda_i^k$. This is because on average it takes $1/\sum_{k=1}^{K} \lambda_i^k$ for a request for video $i$ to arrive. The probability that proxy $g$ has

a request from video $i$ by the threshold $T_i$ is $(1 - e^{-\lambda_i^g T_i})$. Let $M(g)$ be the average number of joins from proxy $g$ per unit of time. Then

$$M(g) = \sum_{i=1}^{N}(1 - e^{-\lambda_i^g T_i})/(T_i + 1/\sum_{k=1}^{K}\lambda_i^k)$$

The average number of joins from all proxies per unit of time is $\sum_{g=1}^{K} M(g)$.

### *B. Strong NP-Complete*

We next prove that the multiple-proxy cache allocation problem in Section III is NP-complete in the strong sense by reducing 3-SAT to this problem[1].

**Proof:** Let $\phi$ be an instance of 3-SAT. Let $k_\phi$ be the number of clauses and $v_\phi$ be the number of variables in $\phi$. In the multiple-proxy cache allocation problem, let each literal represent a video. Therefore, there are $2v_\phi$ videos. We define two types of proxies: variable type (V-type) and clause type (C-type). Each V-type proxy corresponds to a variable in $\phi$. Hence there are $v_\phi$ V-type proxies. Each C-type proxy corresponds to a clause in $\phi$. Hence there are $k_\phi$ C-type proxies. We illustrate the transformation by an example. Suppose

$$\phi = (X \vee Y \vee \bar{Z}) \wedge (\bar{X} \vee Y \vee Z)$$

Then $k_\phi = 2$ and $v_\phi = 3$. In the multiple-proxy cache allocation problem, there are $6$ videos corresponding to the $6$ literals, $3$ V-type proxies and $2$ C-type proxies. We next describe some assumptions.

*Access and size assumptions:* For a V-type proxy, two videos corresponding to the two literals of the variable are accessed from the proxy, with the same access probability of $\lambda$. In the example above, for the V-type proxy corresponding to variable $X$, two videos $X$ and $\bar{X}$ (the two literals for variable $X$) are accessed from this proxy. For a C-type proxy, three videos corresponding to the three literals contained in a clause can be accessed from the proxy, with the same access probability of $\lambda$. In the example above, for the C-type proxy corresponding to the first clause $(X \vee Y \vee \bar{Z})$ in $\phi$, videos $X$, $Y$ and $\bar{Z}$ are accessed from this proxy. We assume that the size of each video is 1. Let the size of a V-type proxy be 1 and the size of a C-type proxy be 2. There is no constraint on the size of the server. The caching grain is chosen to be the size of the video. That is, videos are stored in their entirety or not at all at the proxies.

*Cost Assumptions:* Corresponding to a client's request, if the video is stored at its proxy, the video is transmitted directly from the proxy to the client and we assume the transmission cost of the video is $0$. If the video is not stored at the proxy, the proxy forwards the request to the server and then forwards the video to the client. The first request for a video from the server initiates a transmission of a complete stream. A later request shares the ongoing transmission of the complete stream if it arrives before a threshold. Otherwise, it starts a new transmission of the complete stream. The threshold is chosen to minimize the cost. For a video, we consider the average cost between the interval of two complete streams. Let $C$ denote the average cost aggregated over all the videos. We assume the cost generated from a request that initiates a complete stream is 1. Furthermore, we assume all the later requests for all the videos generate a total cost less than 1. This, as shown later, can be satisfied by setting a proper value of $\lambda$.

We next prove that $\phi$ is satisfiable iff $C < v_\phi + 1$.

- $\phi$ is satisfiable $\Rightarrow C < v_\phi + 1$. Because $\phi$ is satisfiable, there is an assignment to the variables so that each clause is satisfied. At a V-type proxy, we store the video that corresponds to the literal that is false. At a C-type proxy, one video among the three videos to be accessed has to be accessed from the server since a C-type

[1]A problem is NP-complete in the strong sense: it remains NP-complete if any instance of length $n$ is restricted to contain integers of size at most a polynomial $p(n)$.

proxy can store at most 2 videos. We let the video corresponding to a literal that is true to be accessed from the server. In this way, the first accesses to the videos corresponding to false literals generate a cost of 1, with a total of $v_\phi$. The cost for all the later accesses generate a cost less than 1. Therefore we have $C < v_\phi + 1$.

- $C < v_\phi + 1 \Rightarrow \phi$ is satisfiable. At a V-type proxy, a video has to be accessed from the server since the proxy can store at most one video. Set the $v_\phi$ literals corresponding to the $v_\phi$ videos stored at the V-type proxies to be true. That is, the $v_\phi$ videos corresponding to the false literals are accessed from the server and generate a total cost of at least $v_\phi$. Suppose the three videos stored at a C-type proxy all correspond to literals that are false. Then one video corresponding to a false literal has to be served by the server since the proxy can store at most 2 videos, which leads to an extra cost of at least 1. Then the total cost $C$ becomes at least $v_\phi + 1$, which contradicts the fact that $C < v_\phi + 1$. Therefore, at least one literal in a C-type proxy has to be true. Hence $\phi$ is satisfiable.

We are left to prove that the cost assumption is satisfied when the arrival rate $\lambda$ is set to a proper value. Suppose video $i$ is requested at a rate of $\lambda$ from a V-type proxy. It is also requested from $(n-1)$ $(2 \leq n \leq k_\phi + 1)$ C-type proxies, each at a rate of $\lambda$. Without loss of generality, suppose a request from the V-type proxy initiates a transmission of a complete stream of video $i$ from the server. Let $T_i$ be the optimal threshold for video $i$. Let $C_{i1}$ be the average cost of transmitting the complete stream from the V-type proxy. Let $C_{i2}$ be the average costs of transmitting the unicast patches to the other $n-1$ C-type proxies requesting video $i$. Then $C_{i1} = 1 + 1/2\lambda T_i^2$ and $C_{i2} \leq C'_{i2} = 1/2(n-1)\lambda T_i^2$. To minimize $(C_{i1} + C'_{i2})$, we have

$$T_i = \frac{\sqrt{2n\lambda + 1} - 1}{n\lambda}$$

For convenience, we define

$$r_i(n, \lambda) = \frac{C_{i2}}{C_{i1}} \leq \frac{C'_{i2}}{C_{i1}} = \frac{n-1}{1 + f(n, \lambda)}$$

Where,

$$f(n, \lambda) = \frac{n^2\lambda}{n\lambda + 1 - \sqrt{2n\lambda + 1}}$$

We know $f(2, \lambda) > 0$ when $\lambda > 0$. Furthermore, $f(n, \lambda)$ is an increasing function of $n$ for a fix $\lambda$ when $n \geq 1$ and $\lambda > 0$. This can be shown by taking the derivative of $f(n, \lambda)$. Therefore $\min(f(n, \lambda)) = f(2, \lambda)$. Hence $\max(r_i(n, \lambda)) = n/(1 + f(2, \lambda))$.

Suppose the videos requested from the C-type proxies are video $i_1, i_2, \cdots, i_m$. The number of C-type proxies that request video $i_k$ is $n_k$, $k = 1, 2, \cdots, m$. Then

$$C \leq v_\phi + \sum_{i=1}^{m} r(n_i, \lambda) \leq v_\phi + \frac{\sum_{i=1}^{m} n_i}{1 + f(2, \lambda)} = v_\phi + \frac{k_\phi}{1 + f(2, \lambda)}$$

where $v_\phi$ corresponds to the cost from all of the complete streams transmitted from the server. In order to have $C < v_\phi + 1$, we need $\frac{k_\phi}{1+f(2,\lambda)} < 1$, which is satisfied when

$$\lambda < \frac{2(k_\phi - 1)}{(k_\phi - 3)^2}$$

We thus show that the cost assumption is satisfied by choosing proper value for $\lambda$ and complete the proof. ■