Fine-Grained Scalable Video Caching for Heterogeneous Clients

Jiangchuan Liu, *Member, IEEE*, Jianliang Xu, *Member, IEEE*, and Xiaowen Chu, *Member, IEEE*

***Abstract*—Much research has focused on caching adaptive videos to improve system performance for heterogeneous clients with diverse access bandwidths. However, existing rate-adaptive caching systems, which are based on layered coding or transcoding, often suffer from a coarse adaptation and/or a high computation overhead. In this paper, we propose an innovative rate-adaptive caching framework that enables low-cost and fine-grained adaptation by using MPEG-4 fine-grained scalable videos. The proposed framework is both *network-aware* and *media-adaptive*; i.e., the clients can be of heterogeneous streaming rates, and the backbone bandwidth consumption can be adaptively controlled. We develop efficient cache management schemes to determine the best contents to cache and the optimal streaming rate to each client under the framework. We demonstrate via simulations that, compared to nonadaptive caching, the proposed framework with the optimal cache management not only achieves a significant reduction in the data transmission cost, but also enables a flexible utility assignment for the heterogeneous clients. Our results also show that the framework maintains a low computational overhead, which implies that it is practically deployable.**

***Index Terms*—Fine-grained scalable video, proxy caching, resource allocation, streaming media.**

I. Introduction

OWING to the increasing demand for video distribution over the Internet, caching video objects at proxies close to clients has attracted a lot of attention in recent years [9]. However, video objects have several distinct features which make conventional Web caching techniques inefficient, if not entirely inapplicable. In particular, a video object usually has a high data rate and a long playback duration, which together yield a very high data volume. For example, a 1-h MPEG-1 video has a data volume of about 675 MB; caching the video in its entirety is clearly impractical, as several objects of such a size would exhaust the capacity of a typical cache.

Manuscript received January 12, 2005; revised November 4, 2005. An earlier version of this paper was published in IEEE INFOCOM'04. The work of J. Liu was supported in part by a Canadian NSERC Discovery Grant 288325, an NSERC Research Tools and Instruments Grant, a Canada Foundation for Innovation (CFI) New Opportunities Grant, and an SFU President's Research Grant. The work of J. Xu work was supported in part by grants from the Research Grants Council of the Hong Kong SAR, China (Projects HKBU 2115/05E and HKBU FRG/03-04/II-19). The work of X. Chu was supported by RGC HKBU2159/04E and HKBU210605. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Pascal Frossard.

J. Liu is with School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6 Canada (e-mail: jcliu@cs.sfu.ca).

J. Xu and X. Chu are with Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong (e-mail: xujl@comp.hkbu.edu.hk; chxw@comp.hkbu.edu.hk).

Digital Object Identifier 10.1109/TMM.2006.879859

To address this problem, many *partial caching* algorithms have been proposed in recent years [9], [16], [21]. Previous work has demonstrated that when a small portion of a video is stored at the proxy, the network resource requirement can be significantly reduced. Most of these proposals, however, assumed that the streaming rate (either constant or variable) of the video is predetermined. Because of this nonadaptability, they suffer from two limitations: first, it is difficult to meet the diverse bandwidth conditions of heterogeneous clients, as a single streaming rate would either overuse or underuse some client bandwidths; second, there is not enough flexibility to control the backbone (server-to-proxy) bandwidth consumption, since the streaming rates from the proxy to the clients are not adjustable.

One method of overcoming these two limitations is to use rate-adaptive videos. However, this poses significant challenges to caching. Most conventional rate adaptation mechanisms are executed during the encoding process (e.g., adjusting quantizers [8], [22]) and, hence, are difficult to apply to cached videos. There have been research efforts to combine proxy caching with video layering or transcoding [10], [14], [18], [20], but these adaptive systems suffer from either coarse adaptation granularity (due to the inflexible structures of existing layered coders) or a high computation overhead (due to the transcoding operations).

In this paper, we propose a novel video caching framework to achieve low-cost and fine-grained rate adaptation. The framework employs the MPEG-4 fine-grained scalable (FGS) video with bit-plane coding, which enables post-encoding rate control by partitioning the video stream at specific rates [7]. The post-encoding rate control operations can be efficiently implemented at the server or the proxy, resulting in a low computational cost and a fast response. The proposed framework is both *network aware* and *media adaptive*: clients can be of heterogeneous access bandwidths, and adaptive FGS videos are used to meet the clients' bandwidth conditions and control the backbone bandwidth consumption.

We examine the critical design and management issues in the proposed framework. We advocate a semistatic caching paradigm [9], [12], and there are two dimensions to explore of how to cache an FGS video: the *length* and the *rate* of the portion to be cached. We stress that caching decisions must take into account the interactivities in video playback; i.e., nonuniform accesses of different portions. Moreover, when a cached video is delivered to a client, different streaming rates can be selected as long as the rate is no higher than the client's bandwidth. Consequently, the proxy cache management becomes considerably more complex than that for a nonadaptive video based system. Efficient solutions are developed to optimize the resource uti-

lization as well as to offer satisfactory and fair services to the clients.

Working with the optimized proxy cache management, various aspects of the performance of the proposed framework are extensively examined. The results demonstrate its superiority over nonadaptive caching schemes for heterogeneous clients.

The rest of this paper is organized as follows. In Section II, we introduce the background and related work. Section III describes the system model for FGS video-based proxy caching. We then formulate, in Section IV, the problems of optimal cache allocation for a single video. The proposed framework and caching schemes are evaluated in Section V, and we compare them against video replication in Section VI. Finally, Section VII concludes the paper.

II. Related Work and Background

A. Streaming Video Caching

Many caching algorithms for Web proxies have been proposed in the past decade [24]. As mentioned in the Introduction, several distinct features of video objects, such as their large volume, make the conventional Web caching algorithms inapplicable. Hence, many partial caching (e.g., segment-based or interval-based) methods have been proposed [6], [12], [16], [21]. A survey on streaming video caching can be found in [9], and we are particularly interested in those focusing on caching with bandwidth heterogeneous clients.

A straightforward method of handling the heterogeneity of client bandwidths is to produce replicated video streams of different rates [8]. Though used in many commercial streaming products, this method suffers from a high replication redundancy. Other studies have introduced proxy services with active filtering, which reduces the bandwidth of a video object by transcoding [8], [17], [25]. However, they usually incur a much higher computation overhead due to transcoding operations.

A more efficient method is to use layered coding (also known as scalable coding [7]), which compresses a video into several layers: the most-significant layer, called the *base layer*, contains the data representing the most important features of the video, while the additional layers, called the *enhancement layers*, contain the data that progressively refine the quality of the reconstructed video. Layering has been widely used in live video multicast to heterogeneous clients [8]. For proxy-assisted streaming with layered videos, Rejaie *et al.* [14] studied cache replacement and prefecting policies with the objective of alleviating congestion for individual clients. Kangasharju *et al.* [10] simplified the system model by assuming that the cached contents are semistatic and that only complete layers are cached. They developed effective heuristics to maximize the total revenue based on a stochastic knapsack model. The use of other advanced scalability tools, in particular, MPEG-4 object scalability, was investigated by Liu *et al.* [27] and Schojer *et al.* [15]. In [15], a quality-based intelligent gateway, QBIX-G, is introduced, which performs both caching and filtering functionalities. QBIX-G can act as a broker accommodating heterogeneous user requirements and rate variations of the videos. Our work differs from these previous studies in two aspects. First, besides the performance perceived by an individual client, we also investigate the optimal resource allocation across the clients of a proxy, which is important from a system point of view. Second, the previous studies employed conventional layered coding, where the number of layers is restricted and the rate of each layer is often fixed. In contrast, our work employs fine-grained scalable video to enhance adaptability.

B. FGS Video

FGS coding generalizes the conventional layering (scalable coding) methods through a bitplane coding algorithm, which uses embedded representations for the enhancement layer (also called the *FGS layer*) [7]. For example, there are 64 (8×8) DCT coefficients for each video block of the enhancement layer; all the most-significant bits from the 64 DCT coefficients constitute bitplane 0, all the second most-significant bits constitute bitplane 1, and so on and so forth. In the output stream, the bitplanes are placed sequentially to reconstruct the coefficients, and a post-encoding filter can thus truncate this embedded stream of the enhancement layer. Specifically, it detects the boundary of bitplanes (fgs_bp_start_code) in each FGS frame to achieve a specified output rate, and the mismatches due to block boundary constraints can be quite small [7], [13]. It is important to stress that this rate control method has two advantages: 1) like transcoding, it enables fine-grained rate adaptation, but the computation overhead of the filter is much lower and 2) like layered adaptation, it can be applied to stored videos and enables the proxy to adjust the rate of a cached stream at a low cost.

FGS coding has been adopted in the MPEG-4 standard and is undergoing active improvement. We have seen proposals that make efficient use of FGS coding for adaptive video streaming [11], [28], but they did not consider proxy caching. To our knowledge, [26] is the only work that employed FGS videos in caching. The focus of that study, however, was on developing a general cache management framework; in particular, the replacement policies for mixed-media streaming, and the FGS coding was used for the purpose of performance evaluation only.

III. System Model and Notations

The video streaming system in our study consists of a server that stores a repository of videos and a set of proxies at the edge of the network. Selected videos are partially cached at the proxies. A client request is first forwarded to a nearby proxy, which intercepts the request and computes a schedule for streaming: the cached portion of the video can be delivered to the client directly; uncached portions, if needed, will be fetched from the server and then relayed to the client. Although this process is similar to that of many existing systems, our model has three novel features that make it more general and flexible.

First, we consider a more complex network behind the proxy, instead of a simple local area network (LAN) that is assumed in most existing studies. Examples include an enterprise network or a campus network, which remains highly heterogeneous in terms of client access bandwidth due to such factors as hardware configurations, connection methods (e.g., Ethernet, ADSL, or wireless LAN), and administrative policies (e.g., in a

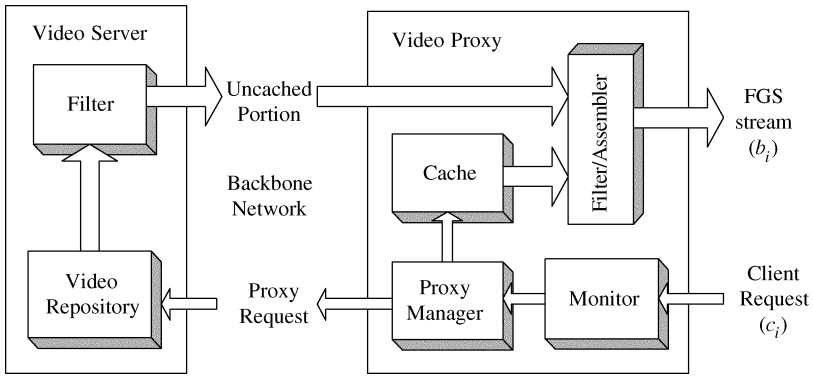

Fig. 1. Functionalities of the video server and a proxy.

campus network, the access bandwidth provisioned for a faculty member would be higher than that for a student). To reflect such heterogeneity, we assume that there are M classes of clients, and that the access bandwidth for a client (or simply *client bandwidth*) of class i is given by c_i, $i = 1, 2, \ldots, M$, which is an upper bound on the video streaming rate from the proxy to a client of class i. Without loss of generality, we number the classes in ascending order of the client bandwidth; that is, $c_1 < c_2 < \cdots < c_M$.

Second, we assume that the clients could terminate a video playback prematurely after they requested the playback from the beginning of a video. Existing studies on video server workloads [1], [2] have revealed that such *early terminations* occur quite often and, hence, should be considered in system dimensioning. One approach for modeling early termination is to partition a video into two parts: a *prefix* and a *suffix*, where the prefix could be a preview of the video. If a client feels uninterested after watching the prefix, it will terminate the connection; otherwise, it will continue playback by retrieving the suffix. We denote the lengths of the prefix and the suffix by L_t and L_s, respectively, and the total length of the video is $L_t (= L_t + L_s)$. Assume the probability of early terminations is p_{ET}, $0 < p_{\mathrm{ET}} < 1$. The probability of a client accessing the entire video (also the probability of accessing the suffix) is thus $1 - p_{\mathrm{ET}}$.

Finally, and most importantly, we advocate scalable adaptive videos in this system; in particular, the MPEG-4 FGS videos. To model an FGS video, we assume that the base layer of the video has a constant rate, r_{base}, which cannot be further partitioned along the rate axis. As such, the base layer represents an ensured minimum playback rate. On the other hand, neglecting the effect of block boundaries for bitplane truncation, the enhancement layer can be adaptively partitioned into a given rate using a filter, either at the server or the proxy. Thus, as illustrated in Fig. 1, a proxy manager can set the streaming rate to a client of class i in the range of $[r_{\mathrm{base}}, c_i]$.

The proxy manager also determines which portion of a video is to be cached. In the conventional nonadaptive video caching system, given a cache size for the video, the portion to be cached is simply determined by its length (in terms of playback time). However, with FGS videos, there is one more dimension to explore: the rate of the portion to be cached. Such flexibility potentially enables a better resource utilization, but also complicates the proxy cache management. In addition, there are two cases in which some uncached portion is to be fetched from the server: 1) the length of the demanded portion is longer than that of the cached portion; and 2) the streaming rate is higher than that of the cached portion. In the second case, the uncached bitplanes will be fetched from the server and then assembled with the cached portion to form a higher-rate stream.

As in many previous studies [12], [16], [21], we assume that the contents of the proxy cache are semistatic and updated periodically with changing system workloads. For each period, the key issues in proxy management are to find the optimal length as well as the optimal rate for partially caching a video, and to determine the streaming rate for each client of the video. To ensure fairness, we let the video streaming rate to any client of class i be identical, denoted by b_i, $b_i \leq c_i$, $i = 1, 2, \ldots, M$. For a multivideo case of N videos, the proxy manager also needs to determine the optimal sharing of resources (cache space and backbone bandwidth) among the videos.

We assume that all these parameters are known *a priori* or measured over time. In addition, for ease of exposition, we focus on the interactions among the origin server, a single proxy, and the clients of the proxy. Our results, however, are generally applicable to the multiproxy case where each proxy serves a nonoverlapping set of clients.

IV. Cache Allocation and Utility Assignment for Single Video

Our objective in designing the proxy cache management module is twofold: first, to maximize the client utility, which is the streaming rate to a client over its available bandwidth; and second, to minimize the transmission cost, as the video streaming imposes very high data delivery demands on the network. Clearly, the above two objectives can be conflictive, and it is important to find a trade-off between them.

A. Transmission Cost Minimization

We first consider the transmission cost for a single video with specified streaming rates for each class of clients, $b_1, b_2, \ldots, b_M$, $r_{\mathrm{base}} \leq b_i \leq c_i$, $i = 1, 2, \ldots, M$. We attempt to answer the following question. Given a limited cache size H for the video, which portion of the stream is cached (referred to as a *caching scheme*) such that the transmission cost is minimized? As suggested in [21], we assume that backbone (i.e., server-to-proxy) transmission dominates the overall cost,

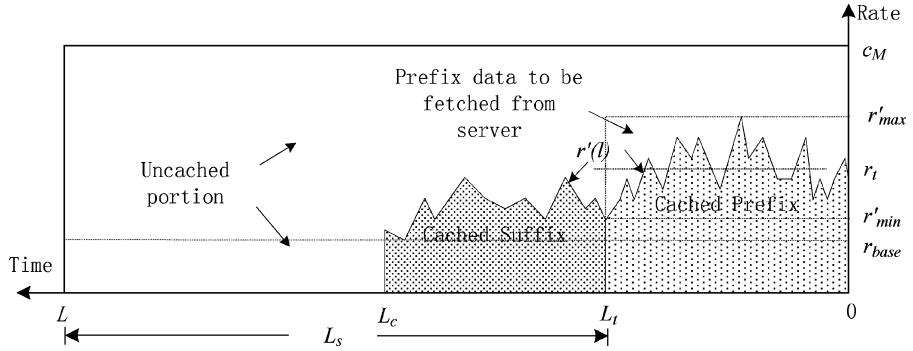

Fig. 2. Illustration of the proof of Lemma 2, where $r'_{\min}$ and $r'_{\max}$ are respectively the lowest and the highest rates in scheme $r'(l)$.

which is a nondecreasing function of the average backbone bandwidth consumption [measured in bits per second (bps)].

Note that the rates for different positions of the cached portion can be different when using FGS videos. Denote the rate for position l (measured by the time elapsed from the beginning of the video) of the cached video by $r(l)$, $l \in [0, L]$. A caching scheme for the video is therefore uniquely determined by the shape of $r(l)$. We say that the scheme is *valid* if 1) $\int_0^L r(l)dl \leq H$ and 2) for any $l \in [0 \ldots L]$, $r_{\text{base}} \leq r(l) \leq c_M$, or $r(l) = 0$. The above two constraints follow the cache size limit and the base layer rate limit, respectively.

Since the clients are heterogeneous in terms of the streaming rates b_i, the backbone bandwidth consumptions can be different for two caching schemes of the same cache size H. A scheme is optimal if it is valid and yields the minimum backbone bandwidth consumption for fetching the uncached portion. Let $\hat{V}$ denote the total volume of the video with rate c_M (the maximum client bandwidth). Obviously, if $H \geq \hat{V}$, the caching scheme $r(l) = c_M$, $l \in [0, L]$ is optimal. For the case of limited cache size, we show two lemmas that facilitate searching the optimal scheme.

Lemma 1: If $H \leq r_{\text{base}} \cdot L_t$, the caching scheme $r(l) = \begin{cases} r_{\text{base}}, & l \in [0, H/r_{\text{base}}] \\ 0, & l \in (H/r_{\text{base}}, L] \end{cases}$ is optimal, i.e., minimizing the backbone bandwidth consumption.

Proof: This scheme is clearly valid. For any client request, the data of volume H is saved from being transmitted over the backbone. This is the maximum saving per client request that a valid caching scheme could achieve under cache size limit H. **Q.E.D.**

Lemma 2: If $r_{\text{base}} \cdot L_t < H < \hat{V}$, assuming that the size of the cached prefix is fixed to $H_t (\in [r_{\text{base}} \cdot L_t, H])$, there exists an optimal scheme that minimizing the backbone bandwidth consumption

$$r(l) = \begin{cases} r_t, & l \in [0, L_t] \\ r_s, & l \in (L_t, L_c] \\ 0, & l \in (L_c, L] \end{cases},$$

where r_t and r_s represent, respectively, the (constant) rates of the cached prefix and cached suffix, and are given by $r_t = H_t/L_t$ and $r_s = \max\{r_{\text{base}}, (H - H_t)/L_s\}$; L_c is the total length of the cached portion, $L_c = L_t + (H - H_t)/r_s$.

Proof: It is easy to verify that $r(l)$ is valid. We prove its optimality by showing that its backbone bandwidth consumption is no higher than that of any valid scheme $r'(l)$, assuming that the size of the cached prefix is fixed to H_t. We will discuss the optimal choice of H_t later in this Section.

We assume that $r'_{\min}$ and $r'_{\max}$ are, respectively, the lowest and the highest rates in scheme $r'(l)$ (see Fig. 2). For scheme $r(l)$, the volume of the prefix data to be fetched from the server is zero for $b_i \leq r_t$ and $b_i \cdot L_t - H_t$ for $b_i > r_t$. For scheme $r'(l)$, however, the corresponding volume is zero only for $b_i \leq r'_{\min}$ and $b_i \cdot L_t - H_t$ only for $b_i > r'_{\max}$. For $r'_{\min} < b_i \leq r_t$, the volume is greater than zero; and for $r_t < b_i \leq r'_{\max}$, it is greater than $b_i \cdot L_t - H_t$. As a result, $r(l)$ yields a higher (or at least an equal) saving than $r'(l)$ for the prefix data to be transmitted over the backbone.

A similar argument applies to the saving of the suffix data to be transmitted. Combining these two proves the lemma. **Q.E.D.**

Now let's concentrate on the case of $r_{\text{base}} \cdot L_t < H < \hat{V}$ with a given H_t. It is easy to show that $r_t \geq r_s$ for an optimal caching scheme because a cached prefix will serve both early terminated requests and all other requests. The backbone bandwidth consumption for all requests from class i is therefore given by function

$$B^i_{H,H_t}(b_i) = \begin{cases} \lambda p_i \left[(1-p_{\text{ET}})(b_i L - H) + p_{\text{ET}}(b_i L_t - H_t)\right], & r_t \leq b_i \leq c_M \\ \lambda p_i (1-p_{\text{ET}}) \left[b_i L_s - r_s (L_c - L_t)\right], & r_s < b_i < r_t \\ \lambda p_i (1-p_{\text{ET}}) b_i (L - L_c), & r_{\text{base}} \leq b_i \leq r_s \end{cases} \quad (1)$$

where λ is the client request rate (the total number of client requests per second), p_i is the probability that a client is in class i; the metric of function $B^i_{H,H_t}(b_i)$ is thus bits per second (bps). The above relation can be easily validated from Fig. 3 ($r_s < b_i < r_t$ in this particular example). Since r_t, r_s, and L_c are all functions of H_t, it follows that the optimal caching scheme for $r_{\text{base}} \cdot L_t < H < \hat{V}$ can be found by a one-dimensional (1-D) search on H_t in the range of $[r_{\text{base}} \cdot L_t, H]$. The optimal solution, or the minimum backbone bandwidth consumption with the given b_i, $i = 1, 2, \ldots, M$, is thus

$$B_H = \min_{r_{\text{base}} \cdot L_t \leq H_t \leq H, r_t \geq r_s} \sum_{i=1}^{M} B^i_{H,H_t}(b_i). \quad (2)$$

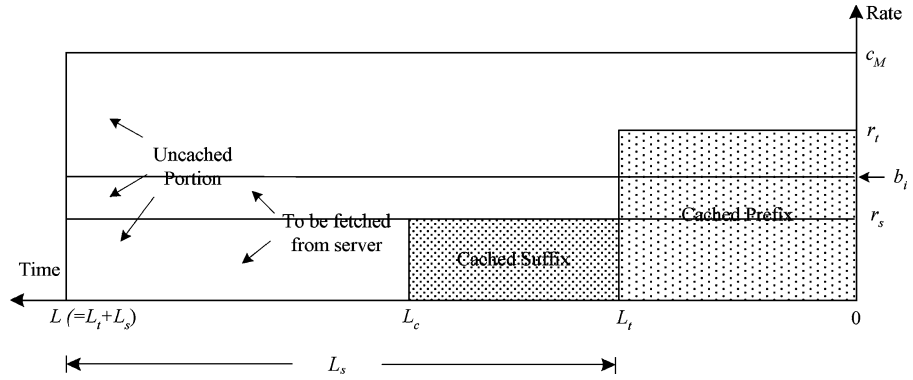

Fig. 3. Illustration of different portions of an FGS video stream ($r_s < b_i < r_t$ in this example).

This search basically finds the best partition between the prefix and suffix to be cached. Assume the minimum space allocation grain is v, which could be the size of a disk block or the size of a group-of-pictures of the video; the cache size H, as well as H_t, is always a multiple of the grain v. Then, the complexity of this 1-D search is bounded by $O(H/v)$ given grain v.

For $H \geq \hat{V}$ and $H \leq r_{\text{base}} \cdot L_t$, the optimal caching scheme can also be uniquely represented using the tuple r_t and r_s. The corresponding backbone bandwidth consumption can thus be calculated by (1) and (2) as well.

B. Trading Off Backbone Bandwidth With Client Utility

In the above optimization, we assume that the streaming rates for the clients of the video are specified. We now consider a more general and flexible scheme that makes good use of FGS videos.

We define the utility of a class i client as $\alpha_i = b_i/c_i$ for $r_{\text{base}} \leq b_i \leq c_i$. The ideal utility assignment is $\alpha_i = 1$, $i = 1, 2, \ldots, M$; i.e., the client bandwidth is fully utilized for every class. This ideal assignment may result in an unaffordable transmission cost. To reduce the cost, one solution is to block some of the client requests. Although this has been suggested in previous studies, it is unfair to the blocked clients. The FGS video, however, offers an alternative method of trading client utility for bandwidth conservation; i.e., assigning a relatively lower yet acceptable streaming rate to each class to save bandwidth consumption.

More explicitly, we would like to investigate whether there exists some utility assignment to each class of clients of the video, such that the backbone bandwidth consumption is no more than $\eta\hat{B}$, where $\hat{B}$ is the backbone bandwidth consumption with the ideal utility assignment and a zero-size cache (i.e., no caching); η thus reflects the factor of backbone bandwidth saving. We are particularly interested in a utility assignment that achieves the best "social welfare"; i.e., the total utility of the clients is maximized. This utility optimization problem for the single video can be formally defined as follows:

$$\begin{aligned}
&\textbf{MU}-\textbf{SV}: \\
&\quad maximize \quad U_{H,\eta} = \sum_{i=1}^{M} \lambda p_i \alpha_i, \\
&\quad s.t. \qquad r_{\text{base}}/c_i \leq \alpha_i \leq 1,\ i = 1, 2, \ldots, M, \\
&\qquad\qquad \alpha_{i-1} c_{i-1} \leq \alpha_i c_i, \quad i = 2, 3, \ldots, M, \\
&\qquad\qquad B_H \leq \eta\hat{B}
\end{aligned} \tag{3}$$

where $U_{H,\eta}$ is the total client utility per unit time for cache size H and bandwidth saving factor η. The constraint $\alpha_{i-1}c_{i-1} \leq \alpha_i c_i$ (equivalent to $b_{i-1} \leq b_i$) preserves the order (priority) of the client classes in resource sharing.

Let auxiliary function $U_{H_t}(i, j, z)$ represent the maximum total utility per unit time for classes 1 through i, with backbone bandwidth j consumed by classes 1 through i, backbone bandwidth z consumed by class i, and the size of the cached prefix is fixed to H_t. We have the recurrence relation in (4), as shown at the bottom of the page. Here, function $(B^i_{H,H_t})^{-1}(z)$ is a generalized inverse of function $B^i_{H,H_t}(b_i)$ [see (1)], representing the highest streaming rate for a class i client given that the backbone bandwidth consumed by this class is z. A special case is $z = 0$. Recall that we assume $\lambda p_i > 0$ and $1 - p_{\text{ET}} > 0$ in the system; zero backbone bandwidth consumption implies that $L_c = L$ and the streaming rate is no higher than r_s. Therefore, if $L_c = L$, we let $(B^i_{H,H_t})^{-1}(0)$ be r_s to maximize the total

$$U_{H_t}(i,j,z) = \begin{cases} \lambda p_1 \left[\left(B^1_{H,H_t}\right)^{-1}(z) \right] / c_1, & \text{if } i = 1, 0 \leq z \leq j \leq \eta\hat{B} \\ \max\limits_{0 \leq x \leq B^{i-1}_{H,H_t}\left[\left(B^i_{H,H_t}\right)^{-1}(z)\right]} \left\{ \lambda p_i \left[\left(B^i_{H,H_t}\right)^{-1}(z) \right] / c_i + U_{H_t}(i-1, j-z, x) \right\}, & \text{if } 1 < i \leq M, 0 \leq z < j \leq \eta\hat{B} \\ -\infty, & \text{otherwise.} \end{cases} \tag{4}$$

utility; otherwise, we set $(B^i_{H,H_t})^{-1}(0)$ to 0 and $B^{i-1}_{H,H_t}(0)$ to $-\infty$. For $b_i < r_{\text{base}}$ and $b_i > c_M$, $B^i_{H,H_t}(b_i)$ is undefined if directly inversing (1); in this case, we set $(B^i_{H,H_t})^{-1}(k)$ to 0 and c_M, respectively.

For $i = 1$ and $j = z$, the correctness of the recurrence relation is obvious, because only class 1 is considered and it consumes all the bandwidth. For $1 < i \leq M$ and $0 \leq z \leq j \leq \eta\hat{B}$, it can be viewed as adding class i to a case of $i-1$ classes. Assuming that the backbone bandwidth consumption for class $i-1$ is x and the backbone bandwidth consumption of class i is z, the maximum total utility for classes 1 through $i-1$ is thus $U_{H_t}(i-1, j-z, x)$. It follows that the maximum total utility for classes 1 through i is $\lambda p_i (B^i_{H,H_t})^{-1}(z)/c_i + U_{H_t}(i-1, j-z, x)$ for the given x, and $U_{H_t}(i, j, z)$ can be obtained by checking all possible values of x. Specifically, given the function of backbone bandwidth consumption $B^i_{H,H_t}(\cdot)$ and its inverse $(B^i_{H,H_t})^{-1}(\cdot)$, and assume that $b'_i = (B^i_{H,H_t})^{-1}(z)$, we have $0 \leq x \leq B^{i-1}_{H,H_t}(b'_i) = B^{i-1}_{H,H_t}[(B^i_{H,H_t})^{-1}(z)]$. This range limit of x is due to the constraint of $\alpha_{i-1}c_{i-1} \leq \alpha_i c_i$, $i = 2, 3, \ldots, M$, which ensures that the streaming rate is nondecreasing for higher classes. Other cases are all invalid and, hence, we set their utilities to $-\infty$.

Assume that there is a minimum backbone bandwidth allocation grain; the backbone bandwidth consumption for a class of clients is rounded to a multiple of w. The solution to problem **MU−SV** is thus given by $U_{H,\eta} = \max_{H_t \in \{m \cdot v | m=0,1,\ldots,H/v\}} \max_{z \in \{n \cdot w | n=0,1,\ldots,\lfloor \eta\hat{B}/w \rfloor\}} U_{H_t}(M, \lfloor \eta\hat{B}/w \rfloor \cdot w, z)$, which can be calculated in time $O(M \cdot \lfloor \eta\hat{B}/w \rfloor^3 \cdot H/v)$. The corresponding utility assignment can be obtained by backtracking the recurrence relation.

V. Performance Evaluation

In this section, we evaluate the performance of our FGS-based caching system via simulations. We examine the system along two dimensions: 1) the data transmission cost, or backbone bandwidth consumption and 2) the quality of the delivered streams; i.e., the client utility.

A. System Settings

We assume that there are five classes of clients and that the client bandwidths of the classes are exponentially spaced; i.e., $c_1 = 128$ Kbps and $c_i = 2c_{i-1}$ for i =2,3,4,5, which cover the bandwidths of a broad spectrum of access technologies. For the client population distribution among the classes, $(p_1, p_1, \ldots, p_5)$, we evaluated various settings in our experiments. Due to space limitations, in this paper, we present the simulation results for three typical distributions: 1) *Uniform*: (0.2,0.2,0.2,0.2,0.2); 2) *S-narrow*: (0.5,0.2,0.15,0.1,0.05); and 3) *S-wide*: (0.05,0.1,0.15,0.2,0.5).

The latter two are skewed distributions, respectively dominated by narrowband clients and wideband clients. The lengths of the entire video and the prefix are set to 100 minutes and 20 min, respectively. The probability of early terminations is set at 30%. Although a rigorous evaluation of the client termination behavior is beyond the scope of this work, we note that much higher probabilities of early terminations have been observed in practice [1], [2]; in this case, more benefits can be expected from the caching paradigm advocated in our system.

We assume that the requests follow a Poisson arrival process with a mean rate of one request per minute. We normalized the backbone bandwidth by $\hat{B}$ (the backbone bandwidth consumption with ideal utility assignment and no caching) and the cache size by $\hat{V}$ (the total volume of the video with rate c_M) for all results presented in this section. Hence, our conclusions are generally applicable when the parameters are proportionally scaled. Unless explicitly specified, the default space allocation grain and backbone bandwidth allocation grain are set at $\hat{V}/200$ and $\hat{B}/200$, respectively. Our experience shows that, under such a setting, the accuracy of the allocation algorithm is quite close to that with a extra fine-grained allocation, and the computation time of the algorithm is generally less than 5 ms on a common PC (Pentium IV 3 GHz), which is reasonably fast for semistatic caching with infrequent updates.

B. Backbone Bandwidth Reduction

In the first set of experiments, we investigate the backbone bandwidth consumption. We are interested in examining the backbone bandwidth reductions by employing the optimal caching scheme, as compared to the following two baseline schemes.

(1) $MaxLen$: $r(l) = \begin{cases} r', l \in [0, H/r'] \\ 0, l \in (H/r', L] \end{cases}$, where $r' = \max\{r_{\text{base}}, H/L\}$.

(2) $MaxRate$: $r(l) = \begin{cases} c_M, l \in [0, H/c_M] \\ 0, \ l \in (H/c_M, L] \end{cases}$.

These two schemes are nonadaptive because they are not aware of the client bandwidth distributions. They also resemble the caching schemes for a coarse-grained layering of two layers; i.e., caching the base layer only and caching both the base and enhancement layers.

Fig. 4 shows the backbone bandwidth reductions for different cache sizes and class distributions. The client bandwidth is assumed to be fully utilized for all classes; that is, $\alpha_i = 1$, $i = 1, 2, \ldots, M$. We observe remarkable reductions achieved by our optimal scheme over the two baseline schemes, which is generally over 10% and sometimes over 50%. The reduction depends on the class distributions; e.g., for the S-narrow distribution in Fig. 4(b), the reduction is particularly high when compared with the MaxRate scheme, as most of the clients have a relatively low bandwidth and, hence, caching the stream of the highest rate becomes wasteful. In this case, increasing the length of the cached stream is a better alternative. However, compared to our optimal scheme, the MaxLen scheme still suffers from more than 10% bandwidth excess, because it is not flexible in setting the rates for the cached prefix and suffix to better accommodate early terminated requests. In contrast, with the S-wide distribution in Fig. 4(c), since most clients have high bandwidth demands, the MaxRate scheme is better than the MaxLen scheme, but is still suboptimal. Finally, with the Uniform distribution in Fig. 4(a), both MaxLen and MaxRate are far from satisfactory.

C. Utility Improvement

We now examine the tradeoff between the client utility and the backbone bandwidth consumption, given that the streaming

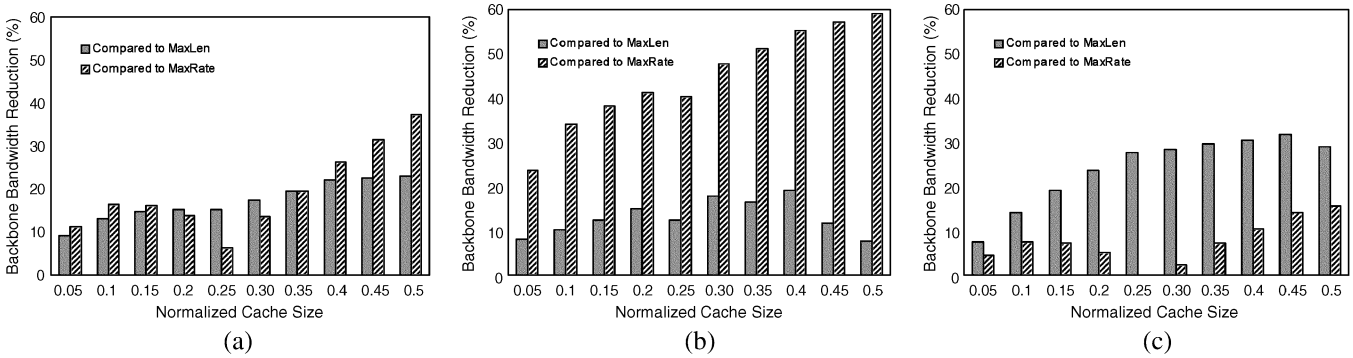

Fig. 4. Backbone bandwidth reductions achieved by the optimal caching scheme. All the cache sizes are normalized by $\hat{V}$. (a) Uniform class distribution. (b) S-narrow class distribution. (c) S-wide class distribution.

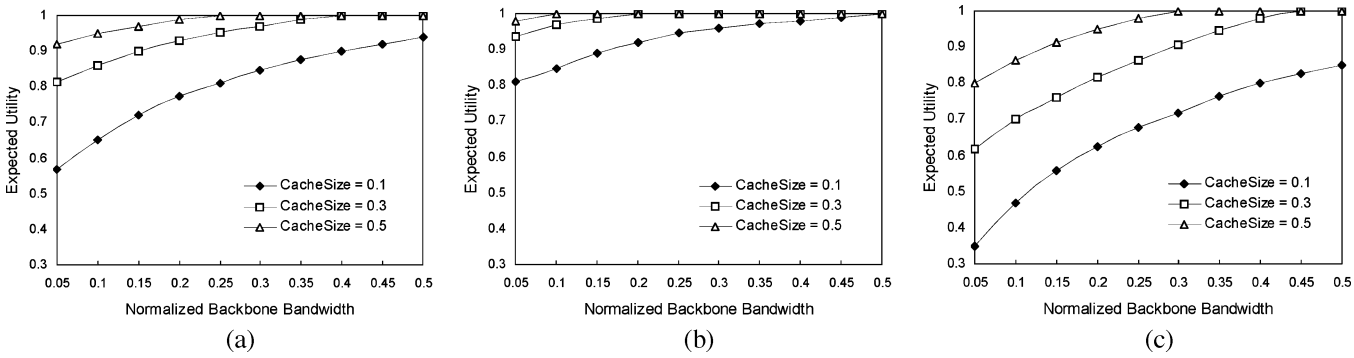

Fig. 5. Expected client utility as a function of backbone bandwidth consumption. All the cache sizes are normalized by $\hat{V}$. (a) Uniform class distribution. (b) S-narrow class distribution. (c) S-wide class distribution.

rates to the clients can be regulated using the filter/assembler at the proxy. We employ the optimal caching and utility assignment algorithms for our system, as described in Section IV-B. Fig. 5 shows the average utility of all clients as a function of backbone bandwidth consumption for different cache sizes and class distributions. Note that, according to (3), the normalized backbone bandwidth is essentially equal to η for the optimal solution.

It can be seen that, to achieve the optimal utility (=1), a relatively high backbone bandwidth is to be consumed if the cache size is very small; e.g., 40% of backbone bandwidth $\hat{B}$ with a cache size of 0.3 $\hat{V}$ for the uniform class distribution Fig. 5(a). However, there is a nonlinear relation between the backbone bandwidth consumption and the client utility. As a result, for the same setting, we can achieve an average client utility of 0.9 by consuming only 15% of $\hat{B}$; that is, a 10% utility reduction leads to a 62.5% backbone bandwidth reduction. This is particularly evident for the S-narrow distribution in Fig. 5(b), not only because a relatively high volume is cached for the stream to a narrowband client, but also because a slight reduction of the streaming rate to a wideband client will benefit the set of narrowband clients. In this case, the average utility is over 0.8 even with very limited resources (e.g., a backbone bandwidth of 0.05 $\hat{B}$ and a cache size of 0.1 $\hat{V}$). For the S-wide class distribution, the reduction is not that significant. However, in this case, the absolute backbone bandwidth consumption is much higher than that for the other two distributions; hence, a slight reduction in the normalized bandwidth would still lead to a great reduction in the absolute backbone bandwidth consumed.

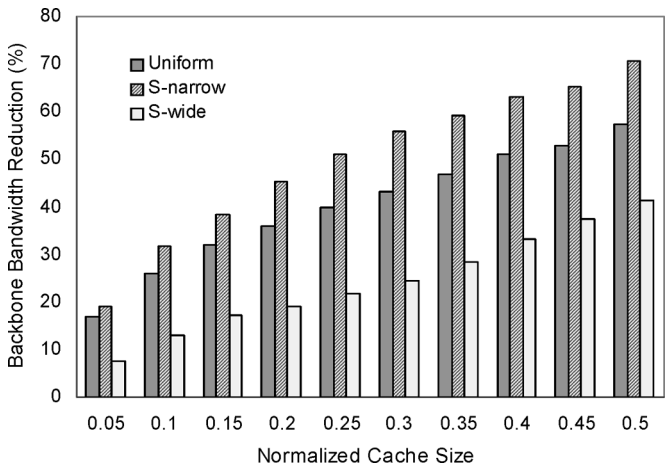

Fig. 6. Backbone bandwidth reduction of FGS video-based caching against stream replication-based caching for different cache sizes and class distributions.

Such an adaptive setting of utility offers a flexible space for a designer to explore how to either maximize the overall revenue or minimize the overall cost. In contrast, if the cache size is less than 0.1 $\hat{V}$ and the backbone bandwidth is less than 0.2 $\hat{B}$, a nonadaptive system that fixes the client utility to one does not even work for any class distribution.

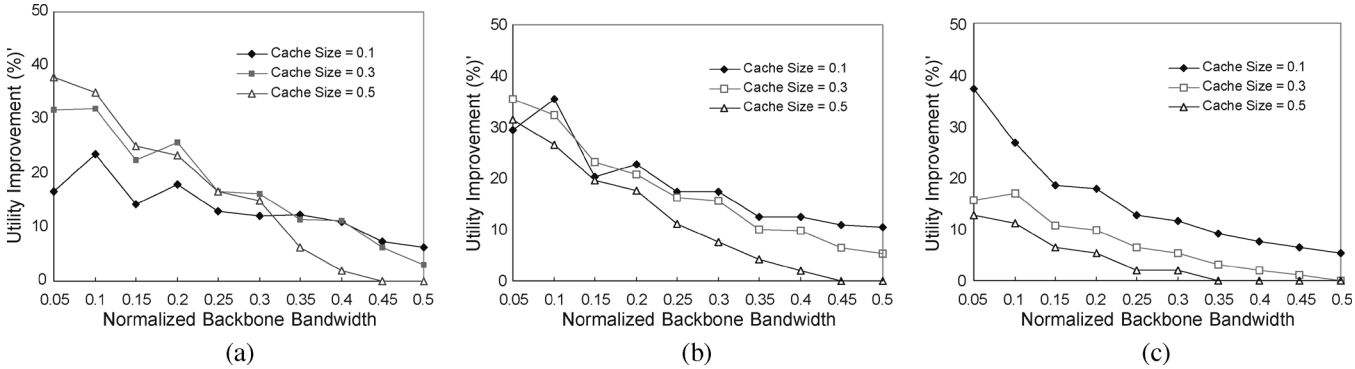

Fig. 7. Utility improvement of FGS video-based caching against stream replication-based caching. All the cache sizes are normalized by $\hat{V}$. (a) Uniform distribution. (b) S-narrow distribution. (c) S-wide distribution.

VI. Comparisons With Replicated Video

Our performance evaluation has shown that scalable videos can accommodate heterogeneous clients much better than can nonscalable videos. As mentioned in Section II, yet another solution to bandwidth heterogeneity is stream replication [3], [8]. It has several advantages, such as simplicity and compatibility with existing nonscalable video coders, and hence has been widely used in commercial streaming systems nowadays; e.g., the RealNetwork's SureStream. Nevertheless, replication also leads to data redundancy, not only in server storage but also in proxy cache and transmission.

Scalable video and replicated video-based streaming systems have been compared in [3] with no proxy caching. A cache-aware comparison was made by Hartanto *et al.* [5]. They considered coarse-grained layering (two layers) and the metric of interest was the request blocking probability. In this section, we offer a comparison from the client utility and bandwidth consumption points of view, with the use of FGS videos.

We first investigate the backbone bandwidth consumptions of the two approaches when the client bandwidth is fully utilized. To make a fair comparison, we develop the optimal caching scheme for the stream replication-based system. In this system, there are M replicated stream of different rates, each for a class of clients, and the cached portion of a stream serves the clients of its own class only. The problem is thus how to partition a cache with given size H for the M streams. This is a variation of the cache allocation problem for multiple heterogeneous videos, as solved in [19].

Using the optimal caching schemes, we compare the backbone bandwidth reduction of the FGS video-based system against the stream replication-based system, as shown in Fig. 6. For the Uniform and S-narrow distributions, the reduction is significant, often over 40% and sometimes over 60%. For the S-wide distribution, the reduction is smaller because the backbone bandwidth consumption is dominated by the requests from the wideband clients. Nevertheless, the absolute value of the saved bandwidth remains high enough in this case, as discussed in the previous section.

Next, we consider the case of flexible utility assignment, where the client utility can be lower than one. The problem of optimal caching and client utility assignment for the stream replication-based system can be formulated as follows:

$$\textbf{MU}-\textbf{REP}:$$

$$\begin{aligned} \text{maximize} \quad & U_{H,\eta} = \sum_{i=1}^{M} \lambda p_i \alpha_i \\ s.t. \quad & r_{\min}/c_i \le \alpha_i \le 1,\ i = 1,2,\ldots,M \\ & \alpha_{i-1} c_{i-1} \le \alpha_i c_i,\ i = 2,3,\ldots,M \\ & \sum_{i=1}^{M} h_i \le H, \text{ and } B_H \le \eta \hat{B} \end{aligned} \tag{5}$$

where h_i is the cache size allocated to stream i, and $r_{\min}$ is the lowest streaming rate, which is set at r_{base} in the experiments to ensure a fair comparison. This problem can also be solved by checking different partitionings of the cache and, for each partitioning, the optimal utility assignment can be obtained using an algorithm similar to that for problem $\textbf{MU}-\textbf{SV}$. Fig. 7 shows the utility improvement of FGS-based caching against stream replication-based caching. It can be seen that, when the backbone bandwidth consumption is lower than 0.25 $\hat{B}$, the improvement is often higher than 20%. It diminishes with increasing backbone bandwidth, since the client utility becomes saturated in both systems. Yet, comparing Figs. 5 and 7, it is clear that the stream replication-based system requires far more resources to reach such a maximum. For example, with the uniform class distribution, the client utility for our FGS-based system has reached one for a cache size of 0.5 $\hat{V}$ and a backbone bandwidth consumption of 0.25 $\hat{B}$ see Fig. 5(a); however, according to Figs. 5(a) and 7, the client utility for the replication-based system is still below 0.85 in this case.

VII. Conclusions

In this paper, we addressed the problem of proxy-assisted video streaming to a group of heterogeneous clients. We proposed a rate-adaptive proxy caching framework using FGS videos, and explored the benefits associated with FGS in handling client heterogeneity as well as reducing transmission cost. We also developed effective solutions to two important proxy management problems in this framework: which portion should be cached for each FGS video, and which streaming rate should

be employed for delivering the stream to each client. Simulation results showed that the proposed framework not only achieves significant backbone bandwidth reduction, but also enables flexible utility assignment for heterogeneous clients. We also conducted a comparison between the FGS-based and the replication-based video caching systems. The results demonstrated the superiority of FGS-based caching.

In our experiments, we used the function $\alpha_i = b_i/c_i$ to measure client utility. It is worth noting that the FGS coding incurs extra overhead for bitstream scaling, which potentially leads to quality degradation. A comprehensive study on this issue involves the use of perception-aware utility functions, which is out of the scope of this paper and is indeed an open research problem. However, we are aware that current studies indicate that the quality degradation caused by such an overhead is generally less than 10%, where the quality is measured by the PSNR [7]. The degradation can be further minimized using smart [23]. In view of this, and considering that FGS has been adopted by the MPEG-4 standard, we believe our FGS video-based caching system offers a promising cost-effective vehicle for streaming video to heterogeneous clients.

REFERENCES

[1] J. M. Almeida, J. Krueger, D. L. Eager, and M. K. Vernon, "Analysis of educational media server workloads," in *Proc. NOSSDAV'01*, Port Jefferson, NY, Jun. 2001.

[2] S. Chen, B. Shen, S. Wee, and X. Zhang, "Designs of high quality streaming proxy systems," in *Proc. IEEE INFOCOM'04*, Hong Kong, Apr. 2004.

[3] P. de Cuetos, D. Saparilla, and K. W. Ross, "Adaptive streaming of stored video in a TCP-friendly context: multiple versions or multiple layers," in *Proc. Packet Video Workshop*, Kyongju, Korea, Apr. 2001.

[4] S. Gruber, J. Rexford, and A. Basso, "Protocol considerations for a prefix-caching proxy for multimedia streams," in *Proc. World Wide Web Conf. (WWW'2000)*, May 2000.

[5] F. Hartanto, J. Kangasharju, M. Reisslein, and K. W. Ross, "Caching video objects: Layers vs versions?," in *Proc. IEEE Int. Conf. Multimedia and Expo (ICME'02)*, Lausanne, Switzerland, Aug. 2002.

[6] S. Jin, A. Bestavros, and A. Iyenger, "Accelerating Internet streaming media delivery using network-aware partial caching," in *Proc. IEEE ICDCS'02*, Vienna, Austria, Jul. 2002.

[7] W. Li, "Overview of the fine granularity scalability in MPEG-4 video standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 301–317, Mar. 2001.

[8] J. Liu, B. Li, and Y.-Q. Zhang, "Adaptive video multicast over the Internet," *IEEE Multimedia*, vol. 10, no. 1, pp. 22–31, Jan. 2003.

[9] J. Liu and J. Xu, "Proxy caching for media streaming over the Internet," *IEEE Commun. Mag.*, vol. 42, no. 8, pp. 88–94, Aug. 2004.

[10] J. Kangasharju, F. Hartanto, M. Reisslein, and K. W. Ross, "Distributing layered encoded video through caches," *IEEE Trans. Comput.*, vol. 51, no. 6, pp. 622–636, Jun. 2002.

[11] T. Kim and M. Ammar, "Optimal quality adaptation for MPEG-4 fine-grained scalable video," in *Proc. IEEE INFOCOM'03*, San Francisco, CA, Apr. 2003.

[12] D. L. Eager, M. C. Ferris, and M. K. Vernon, "Optimized caching in systems with heterogeneous client populations," *Perform. Eval.*, vol. 42, no. 2/3, Sep. 2000.

[13] H. Radha and M. van der Schaar, "Partial transcaling for wireless packet video," in *Proc. Packet Video Workshop*, Pittsburgh, PA, Apr. 2002.

[14] R. Rejaie, H. Yu, M. Handley, and D. Estrin, "Multimedia proxy caching mechanism for quality adaptive streaming applications in the Internet," in *Proc. IEEE INFOCOM'00*, Tel Aviv, Israel, Mar. 2000.

[15] P. Schojer, L. Böszörmenyi, H. Hellwagner, B. Penz, and S. Podlipnig, "Architecture of a quality based intelligent proxy (QBIX) for MPEG-4 videos," in *Proc. World Wide Web Conf. (WWW'2003)*, 2003.

[16] S. Sen, J. Rexford, and D. Towsley, "Proxy prefix caching for multimedia streams," in *Proc. IEEE INFOCOM'99*, New York, Mar. 1999.

[17] B. Shen, S.-J. Lee, and S. Basu, Streaming Media Caching With Transcoding-Enabled Proxies HP Labs, Oct. 2003, Tech. Rep.

[18] X. Tang, F. Zhang, and S. T. Chanson, "Streaming media caching algorithms for transcoding proxies," in *Proc. 31st Int. Conf. on Parallel Processing (ICPP'02)*, Vancouver, BC, Canada, Aug. 2002.

[19] B. Wang, S. Sen, M. Adler, and D. Towsley, "Optimal proxy cache allocation for efficient streaming media distribution," in *Proc. IEEE INFOCOM'02*, New York, Jun. 2002.

[20] J. Liu and B. Li, "A QoS-based joint scheduling and caching algorithm for multimedia objects," *World Wide Web*, vol. 7, no. 3, Jul. 2004.

[21] 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 INFOCOM'98*, San Francisco, CA, Apr. 1998.

[22] D. Wu, Y. T. Hou, and Y.-Q. Zhang, "Transporting real-time video over the Internet: challenges and approaches," *Proc. IEEE*, vol. 88, no. 12, pp. 1855–1875, Dec. 2000.

[23] F. Wu, S. Li, and Y.-Q. Zhang, "A framework for efficient progressive fine granular scalable video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 306–319, Mar. 2001.

[24] J. Xu, J. Liu, B. Li, and X. Jia, "Caching and prefetching for web content distribution," *IEEE Comput. Sci. Eng.(CiSE), Special Issue on Web Engineering*, vol. 6, no. 4, pp. 54–59, Jul./Aug. 2004.

[25] N. Yeadon, F. Garcia, D. Hutchison, and D. Shepherd, "Filters: QoS support mechanisms for multipeer communications," *IEEE J. Select. Areas Commun.*, vol. 14, no. 7, pp. 1245–1262, Sep. 1996.

[26] F. Yu, Q. Zhang, W. Zhu, and Y.-Q. Zhang, "QoS-adaptive proxy caching for multimedia streaming over the Internet," *IEEE Trans. Circuits Syst. Video Technol.*, vol. , no. 3, pp. 257–269, Mar. 2003.

[27] J. Liu, B. Li, H.-R. Shao, W. Zhu, and Y.-Q. Zhang, "A proxy-assisted adaptation framework for object video multicasting," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 3, pp. 402–411, Mar. 2005.

[28] P. de Cuetos, M. Reisslein, and K. W. Ross, Evaluating the Streaming of FGS-Encoded Video With Rate-Distortion Traces Institut Eurecom, Tech. Rep. RR-03-078, Jun. 2003.

Jiangchuan Liu (S'01-M'03) received the B.Eng degree (cum laude) from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree from The Hong Kong University of Science and Technology in 2003, both in computer science.

He is currently an Assistant Professor in the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada. From 2003 to 2004, he was an Assistant Professor at The Chinese University of Hong Kong. His research interests include Internet architecture and protocols, media streaming, wireless *ad hoc* networks, and service overlay networks. He is a co-inventor of one European patent and two U.S. patents and won first-class honors in several regional and national programming contests.

Dr. Liu serves as a Technical Program Committee Member for various networking conferences, including IEEE INFOCOM, IEEE MASS, and IWQoS. He was TPC Co-Chair for The First IEEE International Workshop on Multimedia Systems and Networking (WMSN'05), Information System Co-Chair for IEEE INFOCOM'04, and a Guest Editor for the ACM/Kluwer *Journal of Mobile Networks and Applications* (MONET), Special Issue on Energy Constraints and Lifetime Performance in Wireless Sensor Networks. He is an Editor of *IEEE Communications Surveys and Tutorials*. He was a recipient of the Microsoft research fellowship (2000) and the *Hong Kong Young Scientist Award* (2003). He is a member of the IEEE Communications Society and is an elected member of Sigma Xi.

Jianliang Xu (S'02–M'03) received the B.Eng. degree in computer science and engineering from Zhejiang University, Hangzhou, China, in 1998, and the Ph.D. degree in computer science from Hong Kong University of Science and Technology in 2002.

He is currently an Assistant Professor in the Department of Computer Science, Hong Kong Baptist University. His research interests include mobile and pervasive computing, wireless sensor networks, and distributed systems. He has published over 40 technical papers in these areas, many in prestigious journals and conferences, including ACM SIGMOD, MobiSys, IEEE ICDE, INFOCOM, TKDE, TPDS, and VLDBJ. He is a co-editor of a book entitled *Web Content Delivery* (New York: Springer, 2005).

Dr. Xu has served as a Session Chair and Program Committee Member for many international conferences, including IEEE INFOCOM.

Xiaowen Chu (M'03) received the B.E. degree from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree from the Hong Kong University of Science and Technology, Hong Kong, China, in 2003, both in computer science.

He is currently an Assistant Professor in the Department of Computer Science, Hong Kong Baptist University. His main research interests include optical WDM Networks, multimedia networking, and network security.