World Wide Web: Internet and Web Information Systems, 7, 281–296, 2004

A QoS-Based Joint Scheduling and Caching Algorithm for Multimedia Objects

JIANGCHUAN LIU ljc@cse.cuhk.edu.hk
Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, China

BO LI bli@cs.ust.hk
Department of Computer Science, The Hong Kong University of Science and Technology, Clearwater Bay, Kowloon, Hong Kong, China

Abstract

With the development of the broadband Internet, multimedia services have been widely deployed and contributed to a significant amount of today's Internet traffic. Like normal web objects (e.g., HTML pages and images), media objects can benefit from proxy caching; yet their unique features such as huge size and high bandwidth demand imply that conventional proxy caching strategies have to be substantially revised. Moreover, in the current Internet, clients are highly heterogeneous; it is necessary to differentiate their Quality-of-Service (QoS) requirements in streaming. However, the presence of an intermediate proxy in a streaming system poses great challenges to designers. This paper proposes a novel QoS-based algorithm for media streaming with proxy caching. We employ layered coding and transmission, and jointly consider the problems of caching and scheduling to improve the QoS for the clients. We derive general and effective solutions to the problems and evaluate their performance under various configurations. The results demonstrate that the proposed algorithm can accommodate diverse QoS demands from the clients, and yet satisfy stringent resource limits.

Keywords: media objects, proxy caching, scheduling, Quality-of-Service

1. Introduction

With the development of the broadband Internet, streaming media are getting increasingly popular among users and have contributed to a significant amount of today's Internet traffic [27]. To reduce client-perceived access latencies as well as server/network loads, an effective means is to cache frequently used data at proxies close to clients [13,24]. Given the static nature in contents and the highly localized accesses of media objects, we expect that streaming media can also benefit significant performance improvement from proxy caching. There are, however, some important and unique features of streaming media to be taken into account. In particular, a media object general has a huge data volume due to the high data rate and long playback time. As such, a significant amount of network bandwidth and storage space are to be consumed by media streaming.

To address this problem, a solution is to cache only portions of an object, and a playback is thus accomplished by a joint delivery involving both the proxy and the origin server. There have been many partial caching algorithms proposed in the literature

[1,2,4,6,7,10,12,14,19,21–23,26,28]. However, most of them target homogeneous clients of uniform Quality-of-Service (QoS) demands. This is actually not realistic in the current Internet; it is necessary to differentiate the QoS requirements of clients, e.g., in terms of streaming rates. In this paper, we propose a novel QoS-based algorithm for proxy-assisted media streaming. We employ layered encoding and transmission for media objects to accommodate heterogeneous client requests, and jointly consider two important management problems, namely, the schedule of the layers to be delivered to each client and the selection of the layer segments to be cached at a proxy.

We formulate the above problems with the objective of optimizing client perceived QoS for given resource constraints, specifically, the cache space and the transmission cost for each media object. The formulations are general in the sense they can accommodate various QoS and cost measures. We derive an optimal QoS-based scheduling algorithm for general caching schemes, and then investigate two typical scenarios in detail, namely, streaming with sequential accesses and streaming with non-sequential accesses due to such client interactivities as VCR-like operations. We prove sufficient conditions to achieve the optimal caching for sequential accesses, and provide an iterative algorithm for the non-sequential case. The performances of the algorithms are evaluated through extensive numerical simulations, which demonstrate that they can offer satisfactory quality-of-service to heterogeneous clients with stringent resource constraints. The results also identify the key factors that affect the system performance, including QoS measures, transmission costs, as well as client interactivities.

The rest of this paper is organized as follows. Section 2 describes the proxy-assisted media streaming system. In Section 3, the problem of QoS-based layer scheduling is formulated and solved for general caching schemes. The caching algorithms for both sequential and non-sequential accesses are discussed in Section 4. Section 5 evaluates their performance under various settings. The related works are listed in Section 6. Finally, we conclude the paper in Section 7.

2. System architecture and operations

2.1. System architecture

The Internet Engineering Task Force has developed the RTP/RTCP/RTSP protocol suite dedicated to media streaming over the Internet. The basic functionalities for media transferring are provided by the Real-Time Transport Protocol (RTP), including payload identification, sequence numbering for loss detection, and time stamping for playback control [17]. The Real-Time Streaming Protocol (RTSP) coordinates the delivery of media objects and enables a rich set of control, such as the SETUP method that negotiates transport parameters and the PLAY method that initiates a media streaming [18].

Given these protocols, Figure 1 illustrates a generic architecture for proxy-assisted streaming, which consists of a server that stores a repository of media objects, and a set of proxies at the edge of the network. Each video stored in the server is either entirely or partially cached in a proxy. In this paper, we do not consider cooperation among proxies,

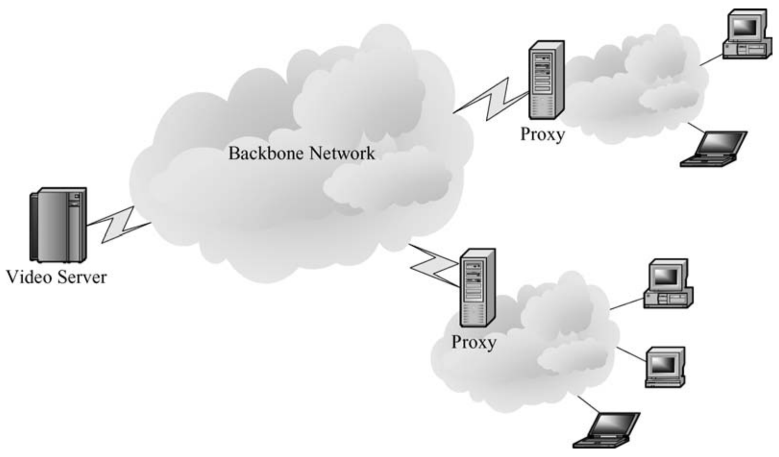


Figure 1. The proxy-assisted video streaming system.

and hence focus on the operations in a single proxy. A client's RTSP PLAY request is first intercepted by the proxy, which attempts to locate the media object of request from its cache; if fails, the request will be forwarded to the origin server, which, as the authoritative source of the object, will streaming it to the proxy through an RTP channel. The proxy then relays the object to the client, and saves a copy in the cache if necessary. If the object is partially cached, the proxy has to fetch the uncached portion from the remote server. Such fetching can be achieved through an RTSP Range request specifying the playback points, and the proxy will then schedule the local transmission to ensure continuously playback.

The problem is particularly complicated considering a heterogeneous enterprise network behind the proxy. As shown in Figure 1, diverse hardware configurations and access interfaces can be used in current enterprise networks, and hence the clients may have quite different QoS demands. Running on top of UDP, RTP itself does not guarantee or differentiate the delivered QoS to clients, but relies on application-layer adaptation. To this end, we employ the cumulatively layered coding for media objects [27], i.e., an object is encoded into several layers, where the first layer (base layer) has a low rate with low quality, and other layers (enhancement layers) can progressively refine the reconstructed quality. Through delivering different number of layers, the system can accommodate clients of diverse QoS demands. Obviously, there are two important problems to be addressed in this system:

1. *Scheduling strategy*: which layer is to be delivered to a client?
2. *Caching strategy*: which segments of which layers are to be cached in the proxy?

For both problems, the objective is to maximized the perceived QoS of the client and yet to satisfy certain resource constraints, including the capacity of the clients, the expected transmission cost, and the cache space. While these two problems are often considered orthogonal with each other, we will demonstrate that they are closely related in the system, and hence a joint optimization is necessary.

Table 1. List of system parameters.

Parameter	Description
T	Length of the media stream
L	Total number of layers
b_i	Bandwidth of layer i
N	Total capacity levels
c_i	Capacity of a level i client
r_i	Streaming rate to a level i client
p_i	Probability of a client at capacity level i
$qos_i(\cdot)$	QoS measurement function of a client of capacity level i
$v_{s\to p}$	Server to proxy transmission cost (per unit bandwidth)
$v_{p\to c}$	Proxy to client transmission cost (per unit bandwidth)
S	Cache size limit of the object
W	Expected cost limit of each request to the object
M_i	Total number of the cached segments for layer i
$(s_{i,j}, t_{i,j})$	Starting and ending positions of cached segment j of layer i

2.2. *System parameters*

In the proposed system, the stream of a media object is partitioned into L cumulative layers, where layer 1 is the base layer, layer L is the least important enhancement layer, and the rate of layer i is b_i, $i = 1, 2, \ldots, L$. There are N different levels of the client capacities, and the probability that a client has capacity level i is p_i. The capacity of a client imposes an upper bound of the total bandwidth of the layers that the client can subscribe to, reflecting the heterogeneous QoS demands from the clients. The aforementioned notations as well as those to be used are summarized in Table 1. We assume that the parameters are estimated online, and the caching and scheduling algorithm is executed periodically in a dynamic network.

3. QoS-based layer scheduling for general caching schemes

We first consider the problem of QoS-based layer scheduling for the proxy-assisted streaming system. Given a constraint of the expected transmission cost for a media object, the proxy should schedule the cumulative layers for each client request such that the expected QoS is optimized. We do not restricted our study to a specific caching strategy, but consider a general caching strategy represented by M_i, $i = 1, 2, \ldots, L$, and $(s_{i,j}, t_{i,j})$, $j = 1, 2, \ldots, M_i$, where M_i is the total number of the cached segments for layer i, and $(s_{i,j}, t_{i,j})$ is the pair of starting and ending positions of segment j of layer i. We also assume that the streaming rate to the clients of the same capacity level should be identical to ensure intra-level fairness, and the QoS of a level i client is a non-decreasing function of streaming rate r_i, denoted as $qos_i(r_i)$. The layer scheduling problem thus can be formulated as follows:

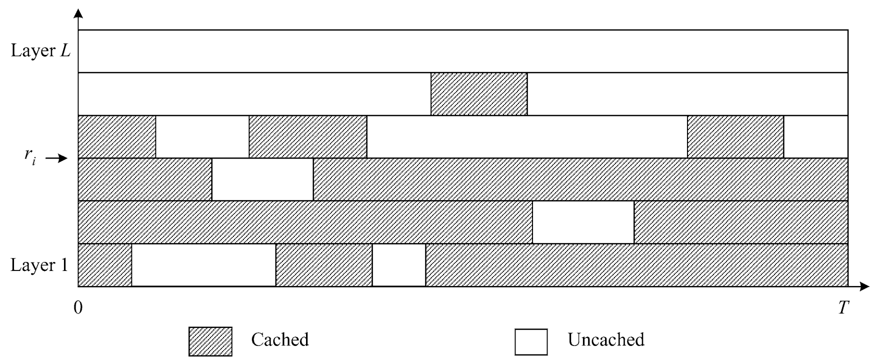


Figure 2. An illustration of the cached/uncached segments of a layered stream.

$$
\begin{aligned}
&\text{maximize } Q = \sum_{i=1}^{N} p_i \cdot qos_i(r_i),\\
&\text{subject to} \quad r_i \in \left\{ \sum_{j=1}^{l} b_j,\ l = 0, 1, 2, \ldots, L \right\},\\
&\qquad 0 \leqslant r_i \leqslant c_i, \quad i = 1, 2, \ldots, N, \\
&\qquad \sum_{i=1}^{L} \sum_{j=1}^{M_i} b_i (t_{i,j} - s_{i,j}) \leqslant S,\\
&\qquad \sum_{i=1}^{N} p_i \big[v_{s\to p} b_{s\to p}(r_i) + v_{p\to c} b_{p\to c}(r_i) \big] \leqslant W,
\end{aligned}
\tag{1}
$$

where $b_{s\to p}(r_i)$ and $b_{p\to c}(r_i)$ respectively represent the average bandwidth demand of the server-to-proxy transmission and the proxy-to-client transmission for a request from a level i client. $v_{s\to p}$ and $v_{p\to c}$ are the respective costs per unit bandwidth. The first constraint follows the layered transmission paradigm; the second ensures that each client will receive a streaming rate not exceeding its capacity; the third ensures that the caching strategy occupies a disk space not exceeding S; and the last constraint places the expected cost limit for each client request.

As an illustration, for a playback with sequential accesses (no VCR-like operations), the values of $b_{s\to p}(r_i)$ and $b_{p\to c}(r_i)$ can be calculated as follows (see Figure 2):

$$
\begin{aligned}
b_{s\to p}(r_i) &= \sum_{i=1}^{\mu(r_i)} \left(1 - \frac{\sum_{j=1}^{M_i} (t_{i,j} - s_{i,j})}{T} \right) b_i,\\
b_{p\to c}(r_i) &= r_i,
\end{aligned}
\tag{2}
$$

where $\mu(r_i)$ satisfies $\sum_{j=1}^{\mu(r_i)} b_j = r_i$.

Since the streaming rates are restricted to a discrete set of cardinality L for layered transmission, the scheduling problem can be solved by an exhaustive search on r_i for the N client capacity levels. The time complexity of this brute-force algorithm is $O(L^N)$, which can be prohibitively high given a large number of client types or a fine-grained layering.

We now show a more efficient dynamic programming algorithm by discretizing the cost. Let $Q_{i,w}$ denote the expected QoS of clients in capacity levels 1 through i with an expected transmission cost of w. The solution to scheduling problem is thus given by $Q_{N,W}$. As an initial case, we have $U_{0,w} = 0$ for $0 \leqslant w \leqslant W$. We then have a recurrence relation:

$$\begin{aligned} Q_{i,w} = & \max_{0 \leqslant h \leqslant L, \sum_{i=1}^{h} b_i \leqslant c_i} \left(\sum_{j=1}^{i} p_j \right)^{-1} \\ & \times \left(\sum_{j=1}^{i-1} p_j \cdot Q_{i-1,[w \cdot \sum_{j=1}^{i} p_j - p_i \cdot V_i(\sum_{j=1}^{h} b_j)]/\sum_{j=1}^{i-1} p_j} + p_i \cdot qos_i \left(\sum_{i=1}^{h} b_i \right) \right) \end{aligned} \tag{3}$$

where $V_i(r)$ is the expect cost of a request from a capacity level i client with streaming rate r, and $V_i(r) = v_{s\to p} b_{s\to p}(r) + v_{p\to c} b_{p\to c}(r)$. For layered transmission, the possible outcome of $V_i(r)$, $i = 1, 2, \ldots, N$ can be pre-computed and stored in $O(NL)$ space, and then extracted through a simple table search. Assume a transmission cost is always rounded to a multiple of a cost unit, g, the above recurrence relation can solved in $O(NL(W/g))$ time.

4. Caching with sequential and non-sequential accesses

Note that the method for optimal layer scheduling is applicable to any caching strategies. If the streaming rate is specified, the bandwidth demand and hence the cost of proxy-to-client transmission is independent of which portion is cached, as illustrated in Equation (2). However, the bandwidth demand for server-to-proxy transmission, $b_{s\to p}(r_i)$, does depend on the caching strategy, given the different segments to be fetched from the server and the skewed populations of the clients at different capacity levels. As such, it is necessary to find a caching strategy that leads to the minimum backbone bandwidth demand and hence the minimum transmission cost.

4.1. Caching strategy with sequential accesses

We first consider the optimal caching strategy for streaming with sequential accesses, that is, a client always starts playback from the beginning of a media, and finishes watching or listening without premature terminations. Without loss of generality, we assume the cache size for a media object is limited, i.e., $S \leqslant T \sum_{i=1}^{L} b_i$. In this case, we have the following observation:

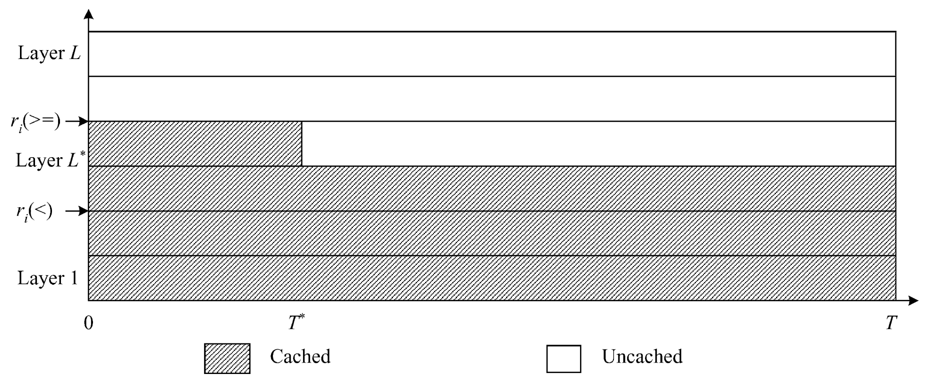


Figure 3. An illustration of the optimal caching strategy for sequential accesses.

Proposition 1.

$$\begin{cases} M_i = 1, (s_{i,1}, t_{i,1}) = (0, T), & \text{if } 1 \leqslant i < L^*, \\ M_i = 1, (s_{i,1}, t_{i,1}) = (0, T^*), & \text{if } i = L^*, \\ M_i = 0, & \text{if } i > L^* \end{cases}$$

is an optimal caching strategy in the sense that it minimizes the transmission cost under any valid layer scheduling. Here, $L^* = \min_l\{S \leqslant T\sum_{i=1}^{l} b_i\}$ and $T^* = (S - T\sum_{i=1}^{L^*-1} b_i)/b_{L^*}$.

Proof: It is sufficient to show this strategy yields the minimum sever-to-client bandwidth demand. Divide the streaming rates to the clients into two cases: $r_i < \sum_{j=1}^{L^*} b_j$ and $r_i \geqslant \sum_{j=1}^{L^*} b_j$. As illustrated in Figure 3, in the first case, the server-to-proxy bandwidth demand is zero, as all the requested portions are cached; in the second case, the data to be fetched from the server is $r_i T - S$. In other words, the data that are saved from server-to-proxy transmission is S, which yields the maximum saving and hence the minimum bandwidth demand. □

Given this optimal caching strategy, the bandwidth demand of server-to-proxy transmission for a capacity level i client can be simplified as $b_{s \to p}(r_i) = \max\{0, r_i - S/T\}$, and the demand of proxy-to-client transmission $v_{s \to p}(r_i)$ remains r_i, as mentioned previously.

4.2. *Caching strategy with non-sequential accesses*

Next, we consider the caching strategy with non-sequential accesses, i.e., with such VCR-like operations as rewind, fast-forward, random-seek, and early termination. In general, it is difficult to model the VCR-like operations, for they largely depend on human behaviors. Here we consider a simple scenario in which the access probabilities to different portions (partitioned along the axis of playback time) of a media stream are known in advance

```
for $i = 1$ to $L$ do
  for $j = 1$ to $K$ do
    $\pi_{i,j} \leftarrow m_i \cdot a_j$;               /* assume that $m$ and $a$ are independent */
$s \leftarrow 0$;                                         /* cache space used so far */
for $i = 1$ to $L$ do $M_i \leftarrow 0$;
while $s < S$ do
  $(i, j) \leftarrow \arg_{(i',j')} \max\{\pi_{i',j'}\}$;   /* find the segment of the maximum access
                                                            rate; in case of multiple maxima, pick the one
                                                            with the smallest $i'$, $j'$ */
  $M_i \leftarrow M_i + 1$; $(s_{i,M_i}, t_{i,M_i}) \leftarrow (x_j, y_j)$;
  cache segment $(s_{i,M_i}, t_{i,M_i})$;               /* in this scenario, $t_{i,M_i} - s_{i,M_i} = T/K$ */
  $s \leftarrow s + b_i \cdot T/K$;
  if $s > S$
    then $M_i \rightarrow M_i - 1$;
    else remove $\pi_{i,j}$ from the candidate list;
end while
```

Figure 4. The subroutine to select the segments of the highest access rates for caching.

or online estimated; the skewness of the access probabilities thus reflects the presence of VCR-like operations, e.g., most client would skip a stuffy portion of a movie through fast-forward, while watch the climax several times through rewind. The streaming with sequential accesses can be viewed as a special case of this model, where the access probabilities are identical for each portion. Since the skewness generally depends on the content of the media object while not the specific streaming rate for a client, the portion access probabilities are identical for the clients of different capacity levels.

Assume the media object is divided into K portions of equal length T/K; the starting and ending points of portion i are $x_i (= (i-1)T/K)$ and $y_i (= x_i + T/K)$, $i = 1, 2, \ldots, K$, respectively. Given the estimated access probabilities of the portions, a_i, $i = 1, 2, \ldots, K$, the access rate of a segment in a specific layer can be calculated as the product of the access rate of the layer and that of the corresponding portion, and the proxy can then cache the segments of the highest access rates. To this end, the layers to be delivered to each client should be known in advance. However, the layer scheduling algorithm proposed in Section 3 is applicable only if the caching strategy is given. In other words, for the case of non-sequential accesses, the layer scheduling and the caching strategies have to be optimized iteratively.

Figure 4 shows a subroutine for locating the segment to cache in the proxy given the access probability of the layers, m_i, $i = 1, 2, \ldots, L$, and that of the portions, a_i, $i = 1, 2, \ldots, K$.

Given this subroutine, we can start from an arbitrary caching strategy and then iterative invoke the optimal layer scheduling algorithm and the above subroutine, until the total number of iterations reaches some threshold or the QoS improvement is smaller than some threshold. For each iteration, the values of $b_{s\rightarrow p}(r_i)$ and $b_{p\rightarrow c}(r_i)$ can be calculated as follows:

$$
\begin{aligned}
b_{s\to p}(r_i) &= \sum_{i=1}^{\mu(r_i)} \left(1 - \sum_{j=1}^{M_i} a_{j'}\right) b_i, \\
b_{p\to c}(r_i) &= r_i,
\end{aligned}
\tag{4}
$$

where $\mu(r_i)$ satisfies $\sum_{j=1}^{\mu(r_i)} b_j = r_i$, and j' is the index of the portion starting at $s_{i,j}$.

5. Numerical results

In this section, we present numerical results for our QoS-based caching and scheduling algorithm. Unless otherwise specified, the following default parameters of a media object will be used in the experiments: $T = 60$ minutes, $L = 5$ layers, and $b_i = 64 \cdot 2^{i-1}$ Kbps, $i = 1, 2, \ldots, 5$. The default cache size S is equal to 15% of the total volume of the object. We assume that there are 200 clients randomly distributed in 20 capacity levels; the client capacity of each level is randomly generated in between 64 Kbps and 2 Mbps. The results presented in this section are averages of 20 runs using the above settings. Note that the optimization algorithms depend only on the normalized access probability of each layer or its segments; thus our conclusions are also valid for systems of other client populations.

5.1. *Impact of caching strategies for sequential accesses*

In the first set of experiment, we investigate the impact of caching strategies for streaming with sequential accesses. To demonstrate the superiority of our optimal caching strategy, we compare its performance with that of two greedy strategies. The first is a *random selection*, which tries to cache all the layers and, if there is no enough space, picks up several victim segments of randomly generated starting and ending points to release. The second is a simple prefix caching, which stores all the layers from the beginning until the cache is filled up.

We adopt a linear client QoS function in this set of experiments, of the form

$$
qos_i(r_i) = \begin{cases} 0.2 + 0.8\dfrac{r_i - 64}{c_i - 64}, & \text{if } 64 \leqslant r_i \leqslant c_i, \\ 0, & \text{otherwise.} \end{cases}
$$

Here we assume that the minimum streaming rate is 64 (Kbps) to ensure an acceptable playback quality. If the streaming rate is in between 64 Kbps and c_i, then the perceived QoS has a linear value normalized in between 0.2 and 1; otherwise, the perceived QoS is 0.

Given a QoS level, all the clients try to subscribe to as many layers as possible to achieve this level. We also assume that the transmission costs for the local (proxy-to-client) network and the backbone (server-to-proxy) network are identical. Figure 5 shows the transmission cost as a function of the client QoS level for the three caching strategies. The cost is normalized by that with no caching. It is clear that the optimal strategy outperforms the two greedy algorithms. In particular, when the specified QoS level is low, the optimal

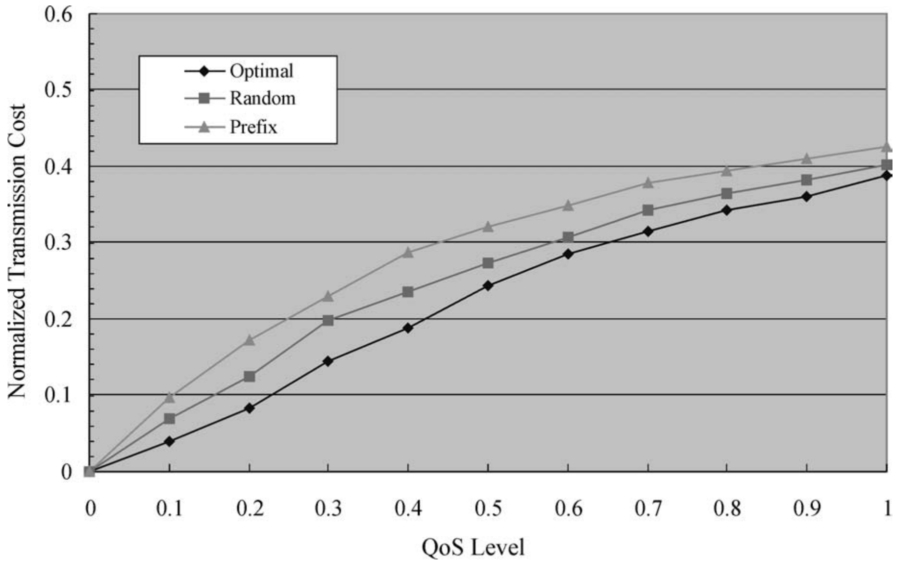


Figure 5. Normalized transmission cost as a function of client QoS level for different caching strategies.

strategy performs much better than the prefix caching, for clients of narrow capacities cannot subscribe to higher enhancement layers, even though the prefixes of these layers are cached.

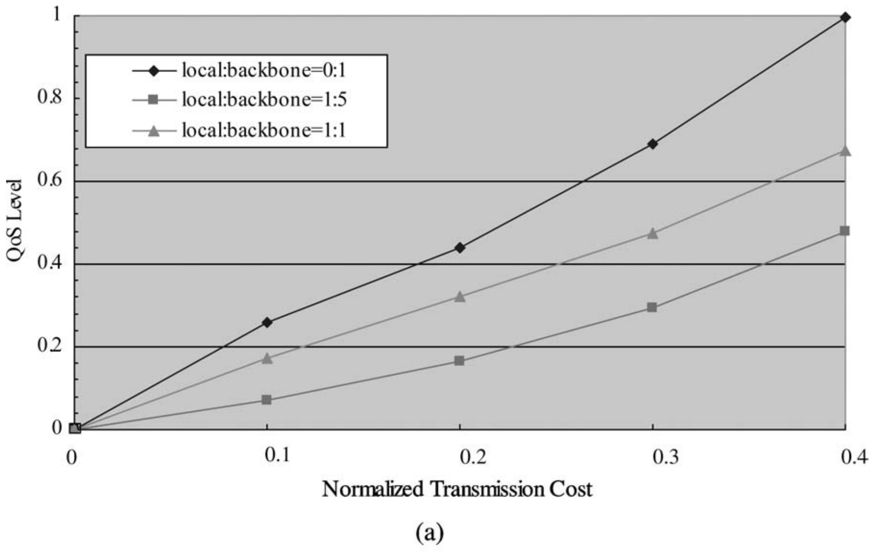
5.2. *Impact of transmission costs*

In the second set of experiments, we examine the relation between the client perceived QoS and the transmission costs. We employ three different combinations of transmission costs (local : backbone), namely, 0 : 1, 1 : 1, 1 : 5. In the first combination, the costs of local transmissions are neglected, which is a common assumption in existing studies. In the other two cases, the costs of local transmissions are taken into account with different weights. We also consider the nonlinearity in the perceived QoS of a client. Specifically, we use a logarithmic function to characterize the nonlinear relation between streaming rate and client perceived QoS, as follows:

$$qos_i(r_i) = \begin{cases} 0.2 + 0.8\dfrac{\log(r_i - 63)}{\log(c_i - 63)}, & \text{if } 64 \leqslant r_i \leqslant c_i, \\ 0, & \text{otherwise.} \end{cases}$$

The rationale of using this concave function is that the perceived QoS for media objects often saturates with high streaming rates.

Figure 6 shows the expected client QoS as a function of transmission costs when the optimal caching and scheduling strategy is adopted. It is clear that the client QoS is improved with increasing transmission costs. Note that the transmission costs are normalized by that with no caching. Figure 6 thus reveals that, even if the cost is quite limited,



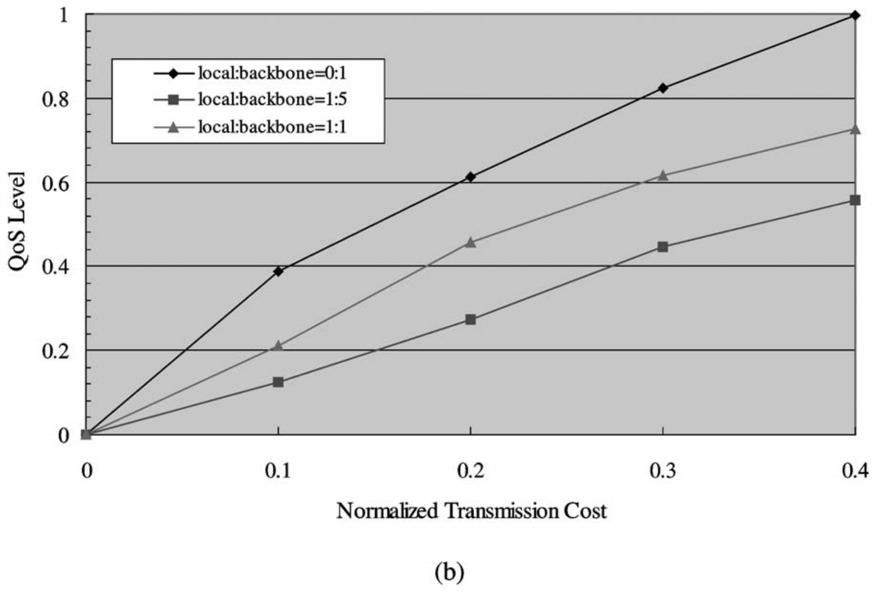


Figure 6. Average client QoS level as a function of the normalized transmission cost. (a) Linear QoS function; (b) nonlinear QoS function.

satisfactory client QoS can be still achieved with proxy caching. For the nonlinear QoS function, the rewards of having more transmission costs are not as significant as the linear measure, simply because the marginal utility quickly diminishes for the nonlinear (log) utility function, as mentioned previously. Among the combinations of transmission costs, (local : backbone $= 0 : 1$) benefits much more from caching than the other two, implying that caching is an effective means to reduce the backbone transmission costs; its effectiveness, however, diminishes if the costs of local transmission are comparable with that of backbone transmission.

5.3. Impact of layering granularity

In this set of experiments, we vary the number of layers to investigate the impact of layering granularity. Clearly, the less the number of layers is, the coarser the adaptation granularity will be. An extreme case is $L = 1$, which degenerates to the single-rate streaming. In Figure 7, we show the expected client QoS as a function of the number of layers. It can be seen that the use of layered stream can significant improves the client utility, as compared to the single-rate case. For the nonlinear QoS measure, as narrowband clients will have more flexible choices with increasing the number of layers, the improvement is more remarkable. Nevertheless, the improvements become marginal with increasing the number of layers. As an example, in Figure 7, the QoS level of 3 layers is about 0.6, which is noticeably higher than that of only one layer (about 0.3); when the number of layers is 5, the QoS has reached 0.7, already close to that of 9 layers (about 0.8). Obviously, when the number of layers is greater than 5, the cost limit becomes the main bottleneck, while not the layering granularity. Considering this as well as the computation costs, we believe that 5-layer is a reasonably good choice, which is used in other simulations.

5.4. Effect of nonuniform access rates

In the last experiments, we consider the non-uniform access rates to different portions of the stream. As mentioned in the previous section, such non-uniform access rates partially reflect effects of using VCR-like operations. To model the non-umiform access rates, given K portion of the stream, we assign each portion a unique ID randomly selected from 1 through K; we then assign an access rate of $(1/i)^\theta / \sum_{j=1}^{K}(1/j)^\theta$ to the portion of ID i. This assignment follows a Zip-like distribution of skew factor θ [30]. When θ is zero, this reduces to a uniform distribution; with the increase of θ, the distribution becomes more and more skewed, i.e., non-uniform.

Figure 8 depicts the expected client QoS as a function of the skew factor when applying the iterative scheduling and caching algorithm with 5 iterations. We set K to 20 in the experiments. It can be seen that the client QoS increases with increasing the skew factor. Intuitively, the skewer the access rates are, the stronger the locality of accesses is. As such, if some good heuristics are employed to cache the segments of high access rates, streaming with VCR-like operations would benefit more from caching than that with sequential accesses only, although the latter is assumed in most existing studies for streaming caching. In practice, the system can identify the skewness according to the access histogram and then decide which caching strategy is to be used.

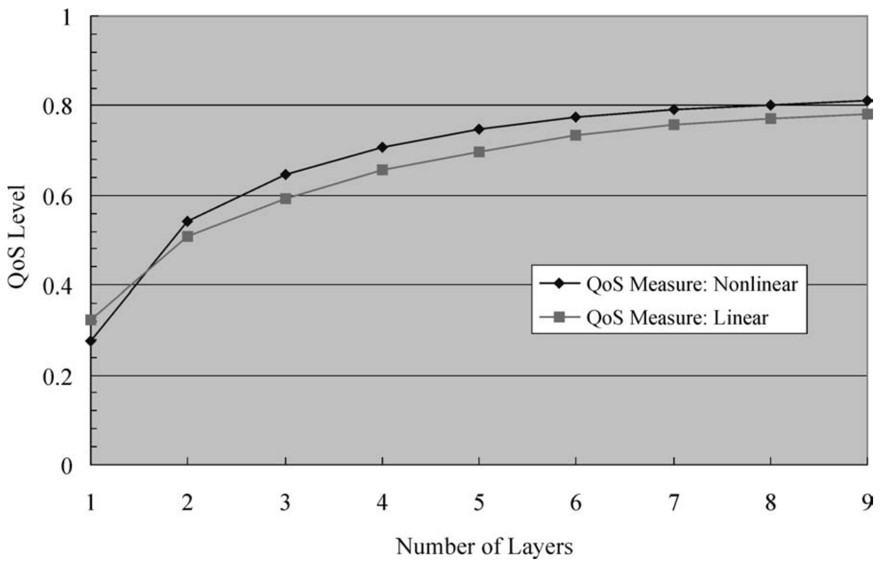


Figure 7. Average client QoS level as a function of the number of layers for linear and nonlinear QoS measures.

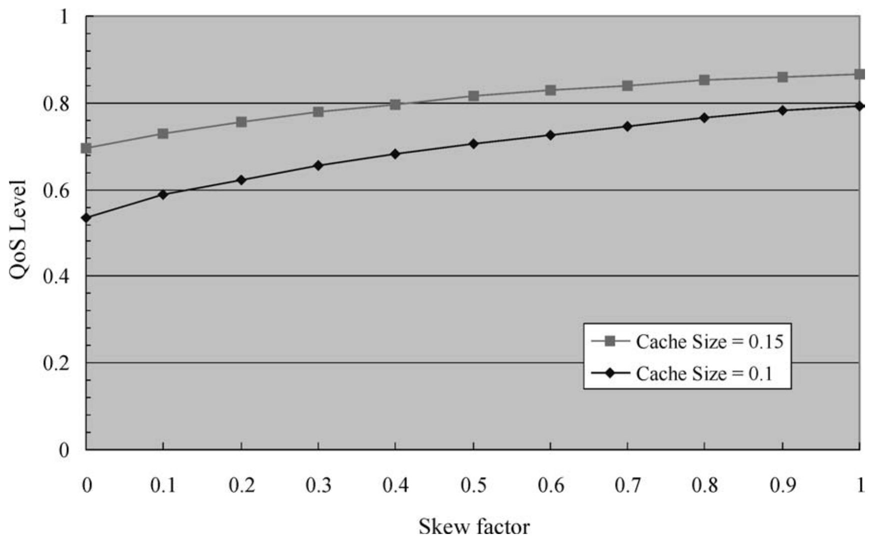


Figure 8. Average client QoS level as a function of the skew factor for non-uniform access rates. The cache size is normalized by the total volume of the object.

6. Previous work

Taking advantage of the temporal and spatial localities of client requests, proxy caching noticeably reduces access latencies and transmission costs by storing recently accessed web objects close to the clients. It thus becomes one of the vital components in existing web systems, and numerous cache management algorithms for web proxies have been proposed in the past decade [13,24]. Streaming media caching, however, has many distinct focuses from conventional web caching beyond the obvious protocol considerations. On one hand, since the content of a media object is rarely updated, issues like cache consistency and coherence are less critical in media caching. On the other hand, given the high resource requirements of media objects, effective management of proxy cache resources (disk space, disk I/O, and network I/O) becomes more challenging.

There has been plenty of works devoted to streaming media caching [1,2,4,6–8,10,12, 14,19,21–23,26,28]. Most of them target homogeneous clients, which have identical or similar configurations and capabilities behind a proxy. As such, a single encoded version of an object would match the bandwidth and format demands of all requests to the object. According to the selection of the portions to cache, we can classify existing algorithms into four categories: *sliding-interval caching*[2,6,21], *prefix caching* [10,14,19], *segment caching* [4,8,28], and *rate-split caching* [26]. Our scheme is particularly related to the rate-split caching in the sense that we also partition the stream along the rate axis. However, in rate-split caching, the primary objective is to reduce rate variability while not to match the heterogeneous QoS demands from clients. Our work is also motivated by the cache allocation scheme proposed in [25]; yet we consider a more general framework with joint caching and scheduling. Our algorithms can also be extended to the multi-object case by employing the resource allocation algorithms in [21,25].

Owing to diverse network models and device configurations, clients behind the same proxy often have quite different requirements on the same object, in terms of streaming rates or encoding formats. To accommodate such heterogeneity, a straightforward solution is to produce replicated streams of different rates or formats, each targeting a subset of clients [9]. Though being widely used in commercial streaming systems, the storage and bandwidth demands of this approach can be prohibitively high. An alternative is to transcode a media from one form to another with a lower rate or a different format in an on-demand fashion [20]. The intensive computation overhead of transcoding, however, prevents a proxy from supporting a large population of clients.

Yet a more efficient approach to this problem is the use of layered encoding and transmission, as advocated in this paper. For layered caching, Kangasharju et al. [11] assume that the cached portions are semi-static and only complete layers are cached. They develop effective heuristics based on an analytical stochastic knapsack model to determine which objects and which layers of the objects should be cached in order to maximize the total revenue. In their model, the client population and the distribution of their capacities are known *a priori*. For layered streaming under dynamic conditions, Rejaiet et al. [15,16] study segment-based cache replacement and prefecting policies to achieve efficient utilization of the cache space and available bandwidth. The main objective is to deal with the congestion problem for individual clients. To this end, the proxy keeps track of popularities of each object on a per layer basis. When the quality of the cached layers is lower than the maximum deliverable quality to an interested client, the proxy sends requests to the server for missing segments within a sliding prefetching window. On cache replacement, a victim layer is identified based on popularities, and its cached segments are flushed from the tail until sufficient space is obtained. Yu et al. [29] studies the caching policies using a fine-grained scalable coding mechanism. They focus on the optimal replacement policies to minimize end-to-end quality distortion for mixed-media streaming. Our work differs from them in that we jointly consider the caching and scheduling problems with QoS constraints as well as VCR-like operations.

7. Conclusions

In this paper, we addressed the problem of QoS-based proxy caching for media streaming over the Internet. We employed a layered streaming framework, and jointly considered two important proxy management issues, namely, *caching*, which is to select segments of the layers cache such that the transmission costs are minimized, and *scheduling*, which is to select the layers to be delivered to clients such that their perceived QoS is optimized and yet the resource constraints are satisfied. We presented effective and general solutions to these two problems both with sequential accesses and with VCR-like operations. Their performance was evaluated through extensive numerical simulations, which showed that the proposed framework not only achieves significant transmission cost reduction but also offers satisfactory quality-of-service to heterogeneous clients.

Acknowledgements

J. Liu's work was supported in part by a grant from Research Grants Council under contract CUHK C001/2050312. B. Li's work was supported in part by grants from RGC under contracts HKUST6196/02E and HKUST6402/03E, a NSFC/RGC joint grant under contract N_HKUST605/02, and a grant from Microsoft Research under the contract MCCL02/03.EG01. The authors thank Profs. Jianliang Xu and Xiaowen Chu at Hong Kong Baptist University for valuable discussions, and thank the anonymous reviewers for their valuable comments.

References

[1] S. Acharya and B. C. Smith, "Middleman: A video caching proxy server," in *Proc. NOSSDAV'00*, June 2000.
[2] J. M. Almeida, D. L. Eager, and M. K. Vernon, "A hybrid caching strategy for streaming media files," in *Proc. Multimedia Computing and Networking*, San Jose, CA, January 2001.
[3] J. M. Almeida, J. Krueger, D. L. Eager, and M. K. Vernon, "Analysis of educational media server workloads," in *Proc. NOSDAV'01*, Port Jefferson, NY, June 2001.
[4] S. Chen, B. Shen, S. Wee, and X. Zhang, "Designs of high quality streaming proxy systems," in *Proc. IEEE INFOCOM'04*, Hong Kong, March 2004.
[5] M. Chesire, A. Wolman, G. Voelker, and H. Levy, "Measurement and analysis of a streaming media workload," in *Proc. 3rd USENIX Symposium on Internet Technologies and Systems*, San Francisco, CA, March 2001.
[6] A. Dan and D. Sitaram, "Multimedia caching strategies for heterogeneous application and server environments," *Multimedia Tools and Applications* 4, May 1997, 279–312.

[7] D. L. Eager, M. C. Ferris, and M. K. Vernon, "Optimized caching in systems with heterogeneous client populations," *Performance Evaluation* 42(2–3), Special Issue on Internet Performance Modeling, 2000, 163–185.

[8] H. Fahmi, M. Latif, S. Sedigh-Ali, A. Ghafoor, P. Liu, and L. Hsu, "Proxy servers for scalable interactive video support," *IEEE Computer* 43(9), 2001, 54–60.

[9] F. Hartanto, J. Kangasharju, M. Reisslein, and K. W. Ross, "Caching video objects: layers vs versions?" in *Proc. IEEE Internat. Conf. on Multimedia and Expo*, Lausanne, Switzerland, August 2002.

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

[11] J. Kangasharju, F. Hartanto, M. Reisslein, and K. W. Ross, "Distributing layered encoded video through caches," *IEEE Trans. on Computers* 51(6), 2002, 622–636.

[12] Z. Miao and A. Ortega, "Scalable proxy caching of video under storage constraints," *IEEE Journal on Selected Areas in Communications* 20(7), Special Issue on Internet Proxy Services, 2002, 1315–1327.

[13] M. Rabinovich and O. Spatscheck, *Web Caching and Replication*, Addison-Wesley, 2002.

[14] S. Ramesh, I. Rhee, and K. Guo, "Multicast with cache (mcache): An adaptive zero-delay video-on-demand service," in *Proc. IEEE INFOCOM'01*, Anchorage, AK, April 2001.

[15] R. Rejaie and J. Kangasharju, "Mocha: A quality adaptive multimedia proxy cache for Internet streaming," in *Proc. NOSSDAV'01*, Port Jefferson, NY, June 2001.

[16] 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, March 2000.

[17] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," IETF RFC 1889, January 1996.

[18] H. Schulzrinne, A. Rao, and R. Lanphier, "Real Time Streaming Protocol (RTSP)," IETF RFC 2326, April 1998.

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

[20] X. Tang, F. Zhang, and S. T. Chanson, "Streaming media caching algorithm for transcoding proxies," in *Proc. 31st Internat. Conf. on Parallel Processing* (*ICPP'02*), August 2002.

[21] R. Tewari, H. M. Vin, A. Dan, and D. Sitaram, "Resource-based caching for Web servers," in *Proc. SPIE/ACM Conf. on Multimedia Computing and Networking* (*MMCN'98*), San Jose, CA, January 1998.

[22] C. Venkatramani, O. Verscheure, P. Frossard, and K.-W. Lee, "Optimal proxy management for multimedia streaming in content distribution networks," in *Proc. NOSSDAV'02*, Miami, FL, May 2002.

[23] O. Verscheure, C. Venkatramani, P. Frossard, and L. Amini, "Joint server scheduling and proxy caching for video delivery," in *Proc. 6th Internat. Content Caching and Distribution Workshops*, Boston, MA, June 2001.

[24] J. Wang, "A survey of Web caching schemes for the Internet," *ACM Computer Communication Review* 29(5), 1999, 36–46.

[25] 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, NY, June 2002.

[26] 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*, April 1998.

[27] D. Wu, T. Hou, and Y.-Q. Zhang, "Transporting real-time video over the Internet: Challenges and approaches," *Proceedings of the IEEE* 88(12), 2000, 1855–1875.

[28] K. L. Wu, P. S. Yu, and J. L. Wolf, "Segment-based proxy caching of multimedia streams," in *Proc. WWW10*, Hong Kong, May 2001.

[29] F. Yu, Q. Zhang, W. Zhu, and Y.-Q. Zhang, "QoS-adaptive proxy caching for multimedia streaming over the Internet," *IEEE Trans. on Circuits and Systems for Video Technology*, 2003, to appear.

[30] G. Zipf, *Human Behavior and the Principle of Least Effort*, Addison-Wesley, Reading, MA, 1949.