Proxy Caching Mechanisms with Video Quality Adjustment

Masahiro Sasabe, Naoki Wakamiya, Masayuki Murata, and Hideo Miyahara

Graduate School of Engineering Science, Osaka University, 1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan

ABSTRACT

The proxy mechanisms widely used in WWW systems offer low-delay data delivery by a means of "proxy server". By applying the proxy mechanism to the video transfer, we expect a real-time and interactive video streaming without introducing extra load on the system. In addition, if the proxy appropriately adjusts the quality of cached video data to the user's demand, video streams can be delivered to users considering their heterogeneous QoS requirements. In this paper, we propose proxy caching mechanisms that can achieve a high-quality video transfer considering the user's demand and the available bandwidth. In our system, a video stream is divided into pieces. The proxy caches them in local buffer, adjusts their quality if necessary, transmits them to users, replaces them with cached data, and retrieves them from the video server, considering user's requirement. We evaluate the proposed video caching mechanisms and compare their performance in terms of the required buffer size, the play-out delay and the video quality. Consequently, the validity of the video quality adjustment in the proxy is confirmed.

Keywords: proxy cache, video streaming, QoS, quality adjustment

1. INTRODUCTION

With the growth of computing power and the proliferation of the Internet, many distributed multimedia applications have been introduced to users. However, those applications inject considerable multimedia traffic into the Internet and cause serious congestions. As a result of congestion, the performance of conventional data applications deteriorates due to the congestion control mechanism such as TCP's. In addition, multimedia applications also suffer from increased transfer delay and frequent packet loss that affect timeliness, interactivity and media quality.

The proxy mechanisms widely used in WWW (World Wide Web) systems offer low-delay data delivery by a means of "proxy server". By applying the proxy mechanisms to the video transfer, we expect that a real-time and interactive video streaming can be performed without introducing extra load on the system. However, a current proxy system cannot be employed because it only handles multimedia files whereas a single video stream sometimes amounts to several tens of giga-bytes. Furthermore, in the case of video delivery service, the client's request on the video quality considerably differs due to heterogeneity in the available bandwidth, end-system performance and user's preference on the perceived video quality. Considering those facts, there have been several research works on video proxy caching mechanisms.[1–4] For example, a multimedia proxy caching mechanism is proposed by Rejaie et al.[1] It uses layered video data to satisfy heterogeneous quality requests . The proxy retrieves the video data layer by layer and segment by segment, considering the availability of bandwidth among the server, the proxy and the clients. They claim the effectiveness of their algorithm in term of the "completeness", which stands for the percentage of cached data, and the "continuity", which represents the smoothness of cached data. However, both of these metrics are only related to the efficiency of cached data and user's impression on the video quality is not considered.

In this paper, we propose proxy caching mechanisms which accomplish a high-quality and low-delay video streaming service while meeting user's demands and available bandwidth. The proxy in our system can adjust cached or retrieved video data to the user's request by means of QoS filtering techniques such as frame dropping, low-pass and re-quantization filters.[5] The video data are divided into segments, each of which corresponds to a set of pictures, intending an efficient use of the cache buffer. Cached data are helpful to reduce response time and accelerate the video playout. However, careful considerations on several issues related to video caching are indispensable to avoid

Further author information: (Send correspondence to m-sasabe@ics.es.osaka-u.ac.jp)
E-mail: {m-sasabe,wakamiya,murata,miyahara}@ics.es.osaka-u.ac.jp
URL: http://www.nal.ics.es.osaka-u.ac.jp

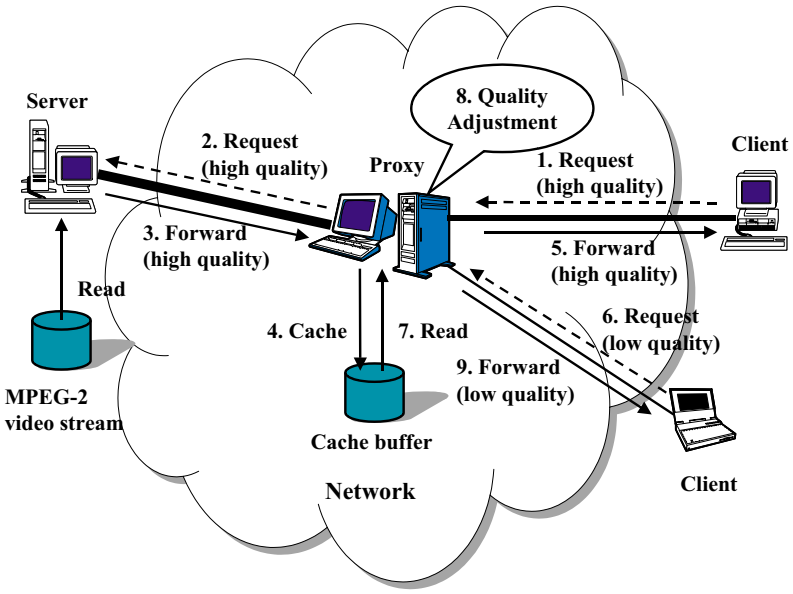


Figure 1. Video Streaming System

a waste of buffer capacity and bandwidth. Those include the data retrieval; how high the quality of video data to retrieve from the server must be in order to satisfy the current and future demands, and the buffer management; which cached data should be shrunk or thrown away to store the newly retrieved data. In addition, we should also consider the effective use of network bandwidth and the heterogeneity of clients. Coping with those problems, we propose several caching mechanisms for video proxy with quality adaptation capability, evaluate and compare them through simulation experiments in terms of the required buffer size, the play-out delay and video quality.

The rest of the paper is organized as follows. In section 2, we describe our assumptions on a video streaming system and propose proxy caching mechanisms with video quality adjustment. Next in section 3, we conduct simulation experiments to evaluate our proposed mechanisms. Finally, we conclude this paper and introduce some future research works.

2. VIDEO STREAMING SYSTEM WITH PROXY CACHE

Figure 1 illustrates our video streaming system. Each client is connected to the proxy cache server. The client begins with establishing a connection to the proxy for the video streaming service, then requests the video data in a unit of a set of pictures. The request on the quality of the video data reflects the user's preference, the system constraints and the available bandwidth. For example, a user with a high performance computer would prefer high resolution and smooth video play out. However, such a demand cannot be satisfied if he is connected to the proxy via, e.g., 33.6 Kbps telephone line. In our evaluations, available bandwidth dominates the video quality assuming that users are tolerant of the quality distortion and the relatively high performance system are used as clients' end-systems.

The available bandwidth for the video streaming corresponds to the allocated bandwidth among server, proxy and client if the network offers a bandwidth reservation mechanism. Another possible case is that the underlying transport protocol controls the sending rate. In most of the cases, multimedia applications prefer UDP than TCP as a transport protocol to avoid the unacceptable delay introduced by the congestion control and the retransmission mechanism of TCP. However, it has been pointed out that greedy and non-cooperative UDP traffic would dominate network bandwidth and the performance of TCP connections considerably deteriorates. For multimedia video transfer, several rate control algorithms such as MPEG-TFRCP (TCP-Friendly Rate Control Protocol for MPEG video transfer)[6,7] or TFRC (TCP Friendly Rate Control)[8] have been proposed aiming at fairly sharing the network bandwidth among conventional data applications and real-time multimedia applications. Considering the current Internet environment, the underlying transport protocol must be "TCP-friendly" and regulate the sending rate according to the network condition.

Table 1. cached data table

GoP	size	quality	quality of receiving data
1	a	A	B
2	b	C	0
3	0	0	D
⋮	⋮	⋮	⋮

2.1. Proxy Caching Mechanisms with Video Quality Adjustment

In our system, considering the data structure of MPEG-2 video and the re-usability of the cached data, the video stream is divided into GoPs (Group of Pictures), each of which corresponds to a set of pictures. A client periodically requests proxy to send a GoP. The quality of the GoP is determined based on the available bandwidth specified by an underlying protocol, i.e, TFRC. However, TFRC determines sending rate independently of upper-layer applications. Since it is not preferable to directly reflect the TFRC rate to the video quality, the TFRC rate observed at the moment when the client issues a request is regarded as available bandwidth on a path between the proxy and the client. Based on the available bandwidth, the quality of GoP which can be transferred within one GoP-time is determined. The GoP-time is given by dividing the number of pictures in a GoP by the frame rate. In our evaluations, the GoP has 30 pictures is played out at 30 frames per second. Thus, one GoP-time is equal to one second.

The proxy maintains tables for each video stream and possesses informations on cached data (Table 1). Those include the GoP number, the size and the quality of the cached GoP and the quality of GoP under transmission. On receiving the request, the proxy compares it to a corresponding entry of the table. If the quality of cached data can satisfy the request (cache hit), the proxy reads out and adjusts the cached data to the request and transmits it to the client. Video quality adjustment is performed by means of the re-quantization filter[5] which controls the quantizer scale, that is, the degree of the quantization. An appropriate quantizer scale is easily derived using relationship among quantizer scale, average rate and average video quality.[9]

Otherwise, the proxy retrieves a video data of suitable quality from the server. The data retrieval is performed on a session established between the server and the proxy for the client whose available bandwidth is also determined by the underlying protocol i.e., TFRC in our case. In determining the quality of the GoP, the proxy takes into account the client's request, the availability of bandwidth and re-usability of cached data. Then, the newly obtained GoP is stored in the cache. If there is not enough room, one or more cached GoPs are replaced with the new GoP. Cached data are useful to reduce the data transfer delay, but further reduction can be expected if the proxy aggressively retrieves GoPs in advance using the residual bandwidth. In the following subsections, we propose several algorithms for quality determination, prefetching and replacement.

2.2. Data Retrieval with Consideration on Client's Request

When the cache cannot supply the client with the GoP of the requested quality, the proxy should retrieve the data whose quality reflects the re-usability of cached data, client's request and available bandwidth. Since the available bandwidth for the data retrieval from the server is determined regardless of that between the proxy and client, the proxy cannot always satisfy the client's demand even if it retrieves video data from the server. We introduce a parameter α, which is given as the ratio of the quality that the proxy can provide to the request.

$$\alpha = \frac{\max(Q_{sp}(i,j), Q_{cache}(i,j))}{Q_{pc}(i,j)} \tag{1}$$

where j is the GoP number that the client i requires. $Q_{sp}(i,j)$ stands for the quality of GoP_j that can be transfered from the server to the proxy within a GoP time and is given as a reciprocal of a quantizer scale determined from the available bandwidth between the server and proxy. The quality affordable on the path between the proxy and the client is expressed as $Q_{pc}(i,j)$ and is regarded as the client_i's request on GoP_j. If exists, the quality of a cached GoP_j, $Q_{cache}(i,j)$, is obtained from a corresponding entry of the cached data table. Otherwise, $Q_{cache}(i,j)$ is zero. In this subsection, since we consider a case of the cache miss, $Q_{cache}(i,j) < Q_{pc}(i,j)$ holds.

When $\alpha \geq 1$, that is, the proxy can provide the client with the GoP of the desired quality, we have three alternatives of determining the quality of GoP to request, $Q_{req}(i,j)$.

method1: A possible greedy way is to request the server to send GoP_j of as high quality as possible. This strategy seems reasonable because cached GoPs can satisfy the most of the future requests and probability of cache misses becomes small. Then, the request $Q_{req}(i,j)$ becomes

$$Q_{req}(i,j) = Q_{sp}(i,j) \tag{2}$$

method2: When the available bandwidth between the server and proxy is extremely larger than that between the proxy and client, method 1 cannot accomplish at effective use of bandwidth and cache buffer. Thus, we propose an alternative which determines the quality $Q_{req}(i,j)$ based on prediction of demands on GoP_j.

$$Q_{req}(i,j) = \min(\max_{k \in S, 0 \leq l \leq j}(Q_{pc}(k,l), Q_{sp}(i,j))) \tag{3}$$

where S is a set of clients which are going to require GoP_j in the future. The client_i is also in S.

method3: To accomplish a further efficient use of the cache, it is possible to request GoP_j of the same quality that the client requests.

$$Q_{req}(i,j) = Q_{pc}(i,j) \tag{4}$$

With this strategy, the number of cached GoPs increases and the probability of cache misses is expected to be suppressed as far as the future requests can be satisfied with them.

In some cases, both cached and retrieved GoP cannot meet the demand ($\alpha < 1$). One way is to request the server to send GoP of the desired quality, but it may cause undesirable delay. The other is to be tolerant the quality degradation and accept the GoP whose quality is lower than the request. We introduce another parameter β to tackle the problem. β is defined as the ratio of the acceptable quality to the demand and expresses the client's insistence on the video quality. Clients with β close to one want to receive the video data in accordance with the request at the risk of unacceptable transfer delay. On the other hand, those who emphasis timeliness and interactivity of applications will choose β close to zero.

First we consider the case that the quality of a cached GoP can satisfy the client, but is still lower than the request ($\beta \leq \frac{Q_{cache}(i,j)}{Q_{pc}(i,j)} \leq \alpha < 1$). In such a case, in order to effectively reuse the cached data, the proxy only sends the cached GoP to the client regardless of the quality of the GoP the server can provide, $Q_{sp}(i,j)$. When the quality of the cached GoP is not high enough ($\frac{Q_{cache}(i,j)}{Q_{pc}(i,j)} < \beta \leq \frac{Q_{sp}(i,j)}{Q_{pc}(i,j)} = \alpha < 1$), the proxy requests the server to send GoP_j whose quality is equal to $Q_{sp}(i,j)$.

$$Q_{req}(i,j) = Q_{sp}(i,j) \tag{5}$$

Finally, if the proxy cannot provide the client with a GoP of satisfactory quality, that is, $\alpha < \beta$, it requests the server to send the GoP of the minimum quality which is expected not to cause a cache miss.

$$Q_{req}(i,j) = \beta \cdot Q_{pc}(i,j) \tag{6}$$

2.3. Prefetching Mechanism

To reduce the possibility of cache misses and avoid the delay in obtaining missing data from the server, the proxy prefetches data that clients are going to require in the future. On receiving a request on GoP_j from the client_i, after checking the cache table for GoP_j, the proxy first compares the minimum requirement $\beta \cdot Q_{pc}(i,j)$ to the quality of cached GoPs $Q_{cache}(i,k)$ and that of receiving GoPs $Q_{rec}(i,k)$ ($j+1 \leq k \leq j+P$). Here, P is the size of a sliding window called a prefetching window, which determines the range of examination for prefetching. If there exists any GoP whose quality is lower than the minimum, a data retrieval mechanism is triggered. The mechanism is the same as one explained in subsection 2.2 except that the available bandwidth to prefetching is the remainder of bandwidth between the server and the proxy.

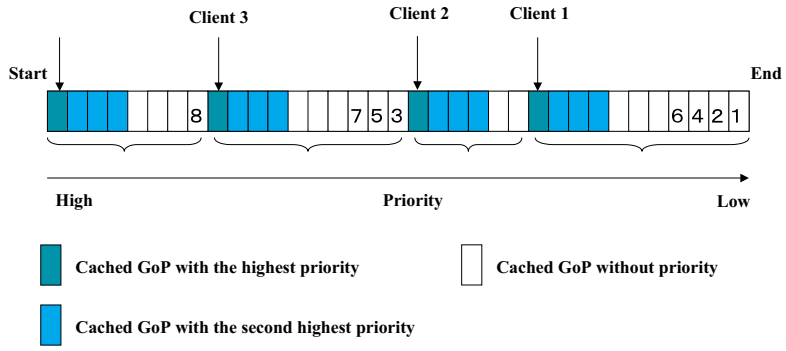


Figure 2. Priorities of cached data

2.4. Replacement Algorithm

When the cache has a limited capacity, a replacement of cached data should be considered to accomplish the effective use of storage. In storing a newly arrived GoP into the cache whose residual capacity is not sufficient, some cached GoPs must be thrown away. We propose a replacement algorithm of cached data with consideration of size, quality and re-usability of data.

First, the proxy assigns priority to cached data. GoPs requested by clients at the moment of the replacement have the highest priority and are never be thrown away from the cache. The GoP resides at the beginning of the stream is also assigned the highest priority to provide potential clients with a low-latency service. The second important GoPs are those in the prefetching windows following the most important GoPs. The other GoPs are with no priority.

The candidate GoPs for replacement are chosen one by one until the sufficient capacity becomes available. In Fig. 2, we show an example of candidate selection. A cached GoP, which locates at the end of longest succession of un-prioritized GoPs, is regarded as the least important and becomes the first candidate as indicated as "1" in the figure. Among successions of the same length, one closer to the end of the stream has a lower priority. The proxy first tries the quality adjustment to decrease the size of the candidate if it is larger than the retrieved GoP. Since it is meaningless to hold GoP whose quality is smaller than $Q_{req}(i,j)$ determined by Eq.(3), no further adjustment is performed and the GoP is thrown away from the cache. If it is insufficient, the proxy chooses the next candidate and applies the same techniques. Finally, the capacity for the newly arrived GoP is sufficient and the proxy caches it in local buffer.

3. SIMULATION EXPERIMENTS

In this section, we conduct simulation experiments to evaluate performance of the proposed caching mechanisms in terms of the required buffer size, the playout delay and the video quality. For comparison purpose, we also consider a traditional caching mechanism without quality adjustment capability.

Figure 3 illustrates our simulation system model. The simulation runs for 29,000 seconds in simulation time unit. Video stream is two hours long (7,200 seconds). Ten clients are connected to the proxy on the same path and watch the same video stream from the beginning to the end without interactions such as rewinding, pausing and fastforwarding. The inter-arrival time between two successive client participations follows the exponential distribution whose average is 1,800 seconds. The size of entire video stream ranges from 8.6 Gbits to 194.5 Gbits according to the applied quantizer scale. The propagation delay between the server and the proxy is 200 msec and that between the proxy and the client is 50 msec. An enlarged view of the available bandwidth between the server and the proxy is shown in Fig. 4(a) and that between the proxy and the client is shown in Fig. 4(b). They are obtained from simulation results on TFRC with ns-2.[10]

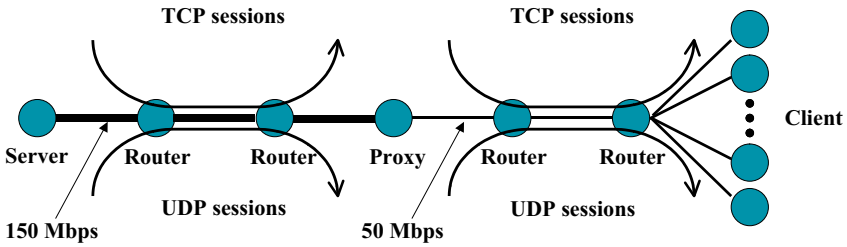
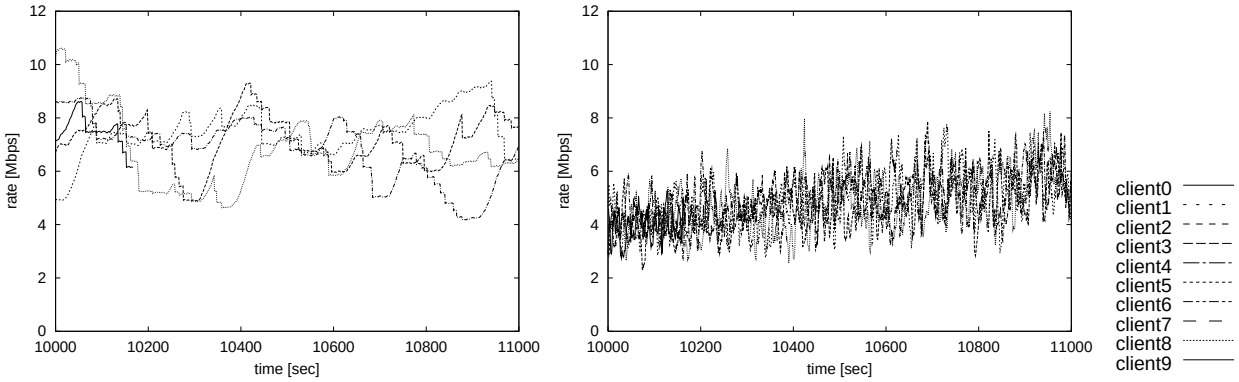


Figure 3. Simulation system model



(a) between server and proxy

(b) between proxy and client

Figure 4. Available bandwidth

Although requests are sent to the proxy at the regular interval of a GoP-time, inter arrival times of GoPs at the client fluctuate due to cache hit, cache miss and available bandwidth. In any types of streaming services, it is necessary for a client to defer the playout and buffer some amount of video data preparing for expected delay jitter. We define the time that the client$_i$ waits and buffers received video data at the beginning of the session in order to ensure regularity and smoothness of video playout as the playout delay $W(i)$. $W(i)$ is derived as:

$$W(i) = \max_{1 \le j \le GoP_{end}} (T(i,j) - I(i,j)) \tag{7}$$

where j stands for the GoP number and GoP_{end} is the number of GoPs in the stream. The arrival time of GoP$_j$ at client$_i$ is denoted as $T(i,j)$. $I(i,j)$ corresponds to the ideal arrival time of GoP$_j$ and those conditions hold that $I(i,j) - I(i,j-1) = 1$ GoP-time and $I(i,1) = T(i,1)$.

Next, we define the degree of user's satisfaction with video quality as

$$S(i) = \frac{1}{GoP_{end}} \sum_{j=1}^{GoP_{end}} \frac{Q_{act}(i,j)}{Q_{pc}(i,j)} \tag{8}$$

where $Q_{act}(i,j)$ is the quality of GoP$_j$ provided to client$_i$ whose request on the GoP is $Q_{pc}(i,j)$.

In Figs. 5 through 7, we summarize simulation results on the playout delay $W(i)$, the amount of cached data and the degree of satisfaction in the system with infinite cache. Those results labeled "traditional" correspond to the case that the proxy does not have capability of neither quality adjustment or prefetching. The traditional proxy requires the server to send video data of the same quality that the client requests if there is not an identical GoP in the cache, and it caches all received GoPs. Even if clients insist on the quality ($\beta = 1$), the playout delay is

suppressed by introducing the quality adjustment and the prefetching mechanism as shown in Fig. 5. In addition, the required buffer size is down to one forth of the traditional method while providing clients with the video data as requested. In the case of method3, the prefetching mechanism is not so effective compared with the others because the proxy retrieves and caches GoPs of the minimum quality.

Next, we show simulation results for the case where the proxy is equipped with the cache of only 20 Gbits, which is smaller than the half of that required (see Fig. 6). If the buffer capacity is limited, we must apply the replacement algorithm proposed in subsection 2.4. Since we cannot expect an efficient use of cached data with obstinate clients, we assume that they are tolerant of quality degradation ($\beta = 0.6$). As shown Figs. 8 through 10, regardless of methods, the playout delay is small enough while the amount of cached data is strictly limited to 20 Gbits (Fig. 9). Due to a limited cache buffer, the degree of satisfaction $S(i)$ slightly decreases but is still higher than 0.6 as shown in Fig. 10.

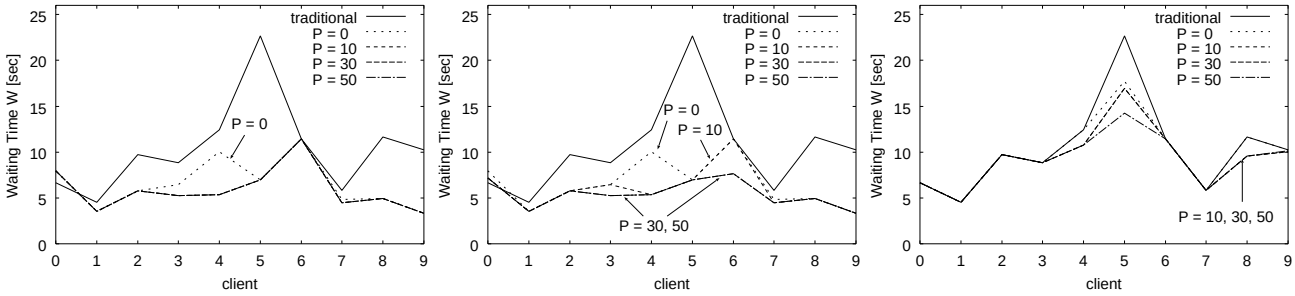
4. CONCLUSION

In this paper, we proposed several caching mechanisms for the video streaming system with the proxy server capable of video quality adjustment. Simulation results show that our system is effective enough in suppressing the playout delay and reducing the required cache size while providing users with a video stream of the desired quality. Especially for the limited buffer, it is shown effective for clients to be tolerant in order to accomplish the low delay and efficient video distribution service.

As future research works, we should examine further effective algorithms, which accomplish the comfortable service with low delay and high quality, even if the cache buffer is limited, the available bandwidth between the server and the proxy is smaller than that between the proxy and the clients, and the users insist on the quality. Moreover, we should consider interactions such as rewinding, pausing and fastforwarding. Furthermore, it is necessary to take into consideration the fairness between streams which shares cache buffer in the case where the proxy treats two or more video stream.

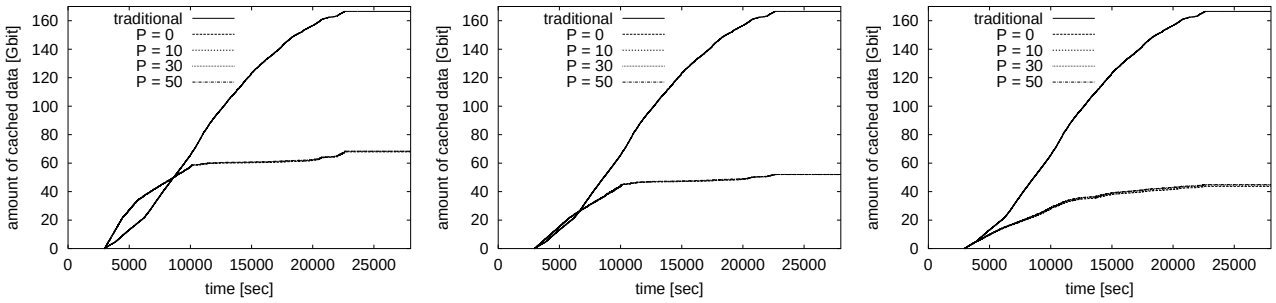
REFERENCES

1. R. Rejaie, H. Yu, M. Handley, and D. Estrin, "Multimedia proxy caching mechanism for quality adaptive streaming applications in the internet," in *Proceedings of IEEE INFOCOM 2000*, March 2000.
2. M. Hofmann, T. S. E. Ng, K. Guo, S. Paul, and H. Zhang, "Caching techiniques for streaming multimedia over the Internet," *Technical Report BL011345-990409-04TM*, April 1999.
3. M. Andrews and K. Munagala, "Online algorithms for caching multimedia streams," in *Proceedings of European Symposium on Algorithms*, pp. 64–75, 2000.
4. M. R. F. Hartanto and K. W. Ross, "Interactive video streaming with proxy servers," in *Proceedings of First International Workshop on Intelligent Multimedia Computing and Networking*, vol. 2, pp. 588–591, February 2000.
5. N. Yeadon, F. García, D. Hutchinson, and D. Shepherd, "Filters: QoS support mechanisms for multipeer communications," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 1245–1262, September 1996.
6. N. Wakamiya, M. Murata, and H. Miyahara, "TCP-friendly video transfer," in *Proceedings of SPIE International Symposium on Information Technologies 2000*, November 2000.
7. M. Miyabayashi, N. Wakamiya, M. Murata, and H. Miyahara, "MPEG-TFRCP: Video transfer with TCP-friendly rate control protocol," to be presented at *IEEE International Conference on Communications 2001 (ICC2001)*, June 2001.
8. S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications: the extended version," *International Computer Science Institute technical report TR-00-03*, March 2000.
9. K. Fukuda, N. Wakamiya, M. Murata, and H. Miyahara, "QoS mapping between user's preference and bandwidth control for video transport," in *Proceedings of IFIP IWQoS '97*, pp. 291–302, May 1997.
10. "UCB/LBNL/VINT Network Simulator - ns (version 2)." available at `http://www-mash.cs.berkeley.edu/ns/`.



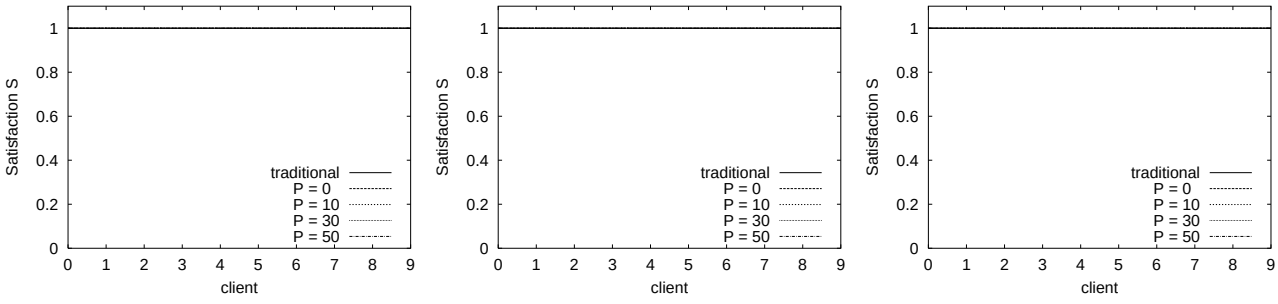
(a) method1 (b) method2 (c) method3

Figure 5. Playout delay with infinite cache ($\beta = 1$)



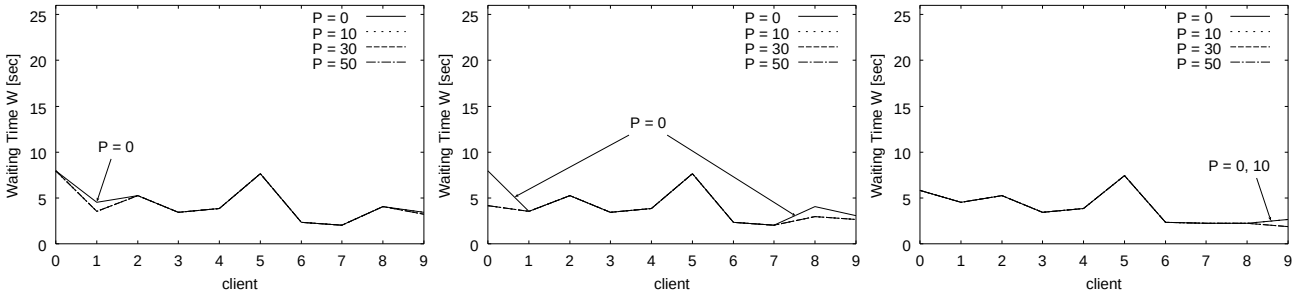
(a) method1 (b) method2 (c) method3

Figure 6. Amount of cached data with infinite cache ($\beta = 1$)



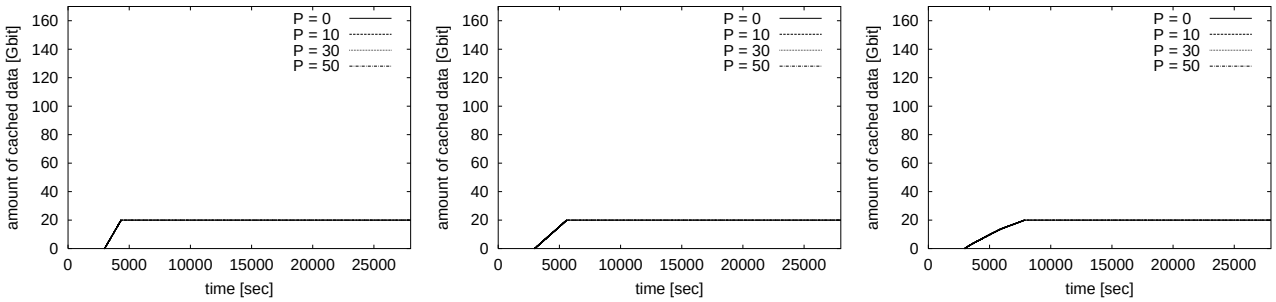
(a) method1 (b) method2 (c) method3

Figure 7. Degree of satisfaction on delivered video with infinite cache ($\beta = 1$)



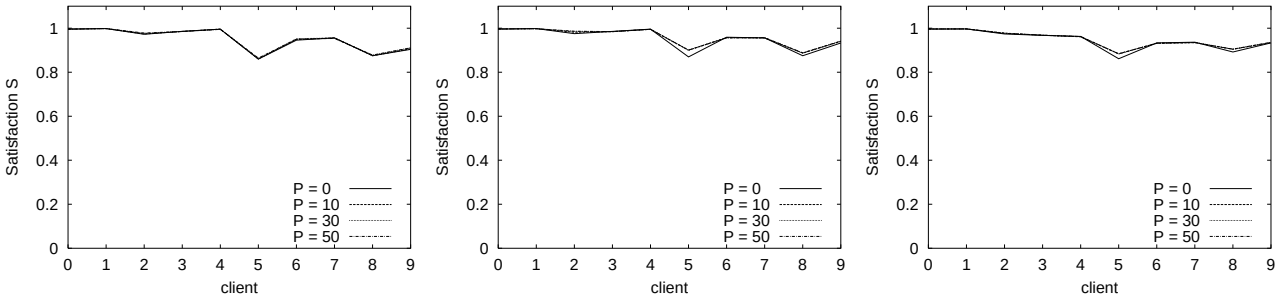
(a) method1 (b) method2 (c) method3

Figure 8. Playout delay with 20 Gbits cache ($\beta = 0.6$)



(a) method1 (b) method2 (c) method3

Figure 9. Amount of cached data with 20 Gbits cache ($\beta = 0.6$)



(a) method1 (b) method2 (c) method3

Figure 10. Degree of satisfaction on delivered video with 20 Gbits cache ($\beta = 0.6$)