# Implementation and Evaluation of Proxy Caching Mechanisms with Video Quality Adjustment

Masahiro Sasabe, Yoshiaki Taniguchi, Naoki Wakamiya, Masayuki Murata, and Hideo Miyahara
Graduate School of Engineering Science, Osaka University, Japan
Tel.: +81-6-6850-6588, Fax.: +81-6-6850-6589

**Abstract**: The proxy mechanism widely used in WWW systems offers low-delay data delivery by means of "proxy server". By applying the proxy mechanisms to the video streaming system, we expect that high-quality and low-delay video distribution can be accomplished without introducing extra load on the system. In addition, it is effective to adapt the quality of cached video data appropriately in the proxy if user requests are diverse due to heterogeneity in terms of the available bandwidth, end-system performance, and user's preferences on the perceived video quality. We have proposed proxy caching mechanisms to accomplish the high-quality and highly-interactive video streaming services. In our proposed system, a video stream is divided into blocks for efficient use of the cache buffer. The proxy server is assumed to be able to adjust the quality of a cached or retrieved video block to the request through video filters. In this paper, to verify the practicality of our mechanisms, we implemented them on a real system and conducted experiments. Through evaluations from several performance aspects, it was shown that our proposed mechanisms can provide users with a low-latency and high-quality video streaming service in a heterogeneous environment.

## 1. Introduction

With the growth of computing power and the proliferation of the Internet, video streaming services become widely deployed. A considerable amount of video traffic injected by the services causes serious congestion and, as a result, network cannot provide users with the real-time and interactive services.

The proxy mechanism widely used in WWW systems offers low-delay delivery of data by means of "proxy server". The proxy server caches multimedia data which have passed through it in its local buffer, called "cache buffer", then it provides the cached data to users on demand. By applying the proxy mechanisms to video streaming system, high-quality and low-delay video distribution can be accomplished without introducing extra load on the system [1, 2]. In addition, it is effective to adapt the quality of a cached video data appropriately in the proxy if user requests are different and diverse due to heterogeneity in terms of the available bandwidth, end-system performance, and user's preferences on the perceived video quality. Taking into account the heterogeneity among clients is indispensable when we want to provide users with a distributed multimedia service of a satisfactory level of quality.

Our research group has proposed proxy caching mechanisms to accomplish the high-quality and highly-interactive video streaming services in a heterogeneous environment [3]. In our proposed system, a video stream is divided into blocks for efficient use of the cache buffer. The proxy cache server is assumed to be able to adjust the quality of a cached or retrieved video block to the request through video filters or transcoders. We developed effective algorithms for determining the quality of a block to retrieve from a video server, replacing cached blocks with a newly retrieved block, and prefetching blocks in prior to requests. These algorithms are important when we want to suppress the possibility of cache misses, and decrease the block transfer delay introduced by retrieving a missing block from a distant video server.

Through simulation experiments, it was shown that our system with the above algorithms can provide users with low-latency, high-quality, and highly-interactive video streaming services. However, several assumptions were made to evaluate the ideal performance of our system. For example, we did not consider processing overhead introduced in quality adjustment. Degradation of video quality caused by packet losses and influence of quality variation on user's perception were also not taken into account.

In this paper, we implemented proposed mechanisms on a real system to verify their practicality and usefulness. We conducted several experiments and evaluated the traffic condition, the video quality variation, and the overhead of quality adjustment. Then, we confirmed that our implemented system can continuously provide users with a video distribution in accordance with the network condition.

The rest of the paper is organized as follows. In section 2, we introduce our proxy caching mechanisms with video quality adjustment. Next in section 3, we describe implementation of proposed mechanisms. In section 4, we conduct several experiments to evaluate our proposed mechanisms. Finally, we conclude this paper and introduce some future research works in section 5.

## 2. Proxy Caching Mechanisms with Video Quality Adjustment

In this section, we briefly introduce our mechanisms for video proxy caching. For further detailed descriptions, refer to our previous work [3].

Figure 1 illustrates the basic behavior of our mechanisms. In our system, considering the re-usability of the cached data, the video stream is divided into blocks. A client periodically requests proxy to send a block. The quality of the block is determined based on the available bandwidth, end-system performance, and user's prefer-

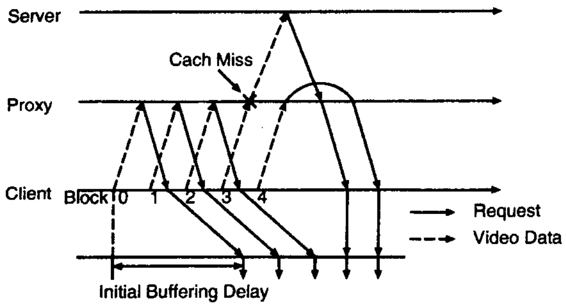



Figure 1. Basic Behavior of Our Mechanisms

ences. The proxy provides the client with the requested block of the desired quality by using the block retrieval mechanism, the cache replacement mechanism, and the block prefetching mechanism. Received blocks are first cached and play-back of blocks is deferred for duration of the pre-determined initial buffering delay.

On receiving a request for a video block from a client, a proxy cache server first examines its local cache buffer. When a video block of a higher quality than the request is stored in the cache buffer, the proxy only needs to adjust the quality of the cached block to the request and send the modified block to the requesting client. However, when the quality of the cached block cannot satisfy the request or there is no corresponding block in the buffer, the proxy must retrieve a missing block from an originating video server. The block retrieval algorithm determines the quality of the video block to retrieve from the server taking into account the client requests, the available bandwidth between the server and the proxy and that between the proxy and the client, and the re-usability of the cached block. For example, preparing for clients that will require the block in the near future, the quality would be set to higher than the request as far as the available bandwidth between the server and the proxy can afford.

Since the proxy is equipped with a cache buffer of the limited capacity, there might not be enough room for storing a newly retrieved video block in the buffer. In such cases, the proxy applies the block replacement algorithm to find, adjust, and discard less important video blocks in the buffer. Each block in the cache is given a priority in accordance with its location and client requests. Blocks that are considered not to be required in the near future are given lower priority and become victims. The proxy first tries to decrease the size of the least important victim by applying video quality adjustment mechanism. If the quality adjustment is not effective enough to make room for the newly retrieved block, then the proxy discards the victim and moves to the next victim. These degradation and discarding trials are repeated until the newly retrieved block is stored or all blocks given lower priorities are discarded.

When the cache hits the request, the bandwidth between the server and the proxy becomes available for the block prefetching. Even in the case of cache misses, there may be room for prefetching when there is much bandwidth between the server and the proxy. On receiving a request, the proxy examines the qualities of blocks succeeding the requested block. A block whose quality is lower than the current request is considered to cause a future cache miss and to be retrieved from the server using the residual bandwidth.

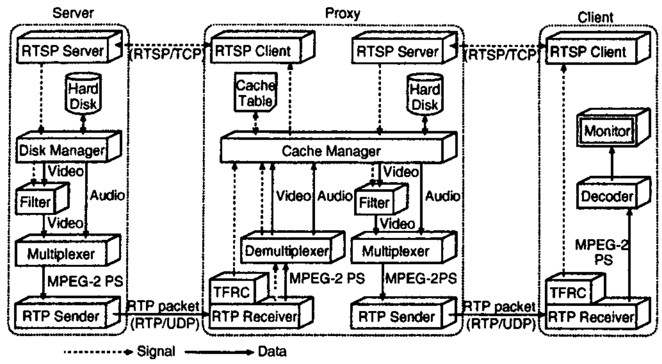



Figure 2. Modules Constituting System

## 3. Implementation of Proposed Mechanisms

Figure 2 illustrates modules constituting our video streaming system. The implemented system consists of a video server, a proxy cache server, and several clients. We employ well-known and widely-used protocols for inter-system communications. For example, the video streaming is controlled through RTSP (Real Time Streaming Protocol) [4] / TCP sessions. Video blocks are transferred over RTP (Realtime Transport Protocol) [5] / UDP sessions as being segmented into 1 K bytes-long RTP packets. The video stream is coded using the MPEG-2 video coding algorithm in the PS (Program Stream) format. The available bandwidth, that is taken into account in three mechanisms explained above, is determined by the underlying rate control mechanism, called TFRC (TCP Friendly Rate Control) [6]. The TFRC is a mechanism proposed for real-time multimedia applications to accomplish the fair share of network bandwidth with TCP data communications. The video quality adjustment is performed by a low-pass filter [7].

### 3.1 Demultiplexing MPEG-2 PS Blocks

MPEG-2 PS is one of formats for multiplexing video and audio streams. As the quality adjustment is applied only to video data, a block received through a proxy's *RTP Receiver* is divided into a pair of video and audio blocks by *Demultiplexer*. The divided blocks are stored in a local cache separately. In our implemented system, each block corresponds to a GoP (Group of Pictures) of MPEG-2, which consists of a series of frames.

The block to request and its quality are specified in the header of an RTSP PLAY message using the Range field and Bandwidth field, respectively. In a case of a cache hit, a proxy reads out both video and audio blocks from its cache, but it applies the quality adjustment only to the video block. Then, those blocks are multiplexed and transmitted to the requesting client.

### 3.2 Rate Control with TFRC

TFRC is a protocol that enables a non-TCP session to behave in a TCP-friendly fashion. TFRC sender estimates the throughput of a TCP session sharing the same path in accordance with network condition, expressed in terms of the packet loss probability and RTT. Those informations are obtained by exchanging RTCP (Real-Time Control Protocol) messages between a sender and a receiver. In our implemented system, we can use RTSP as a feedback mechanism where a client calculates the TCP-friendly rate and informs a proxy of the rate using the Bandwidth field of a RTSP PLAY message.

### 3.3 Video Quality Adjustment

We employ the low-pass filter as a quality adjustment mechanism. We compared several video filters such as the low-pass, re-quantization, and frame dropping [8]. Through experiments, it was shown that the low-pass filter is the most suitable as an MPEG-2 video filter for its flexibility in rate adaptation, faster processing, and video quality. The low-pass filter adjusts the video quality to the desired level by discarding some portion of less influential information in video blocks.

### 3.4 Block Retrieval Mechanism

A proxy maintains information about cached blocks as *Cache Table*. Each entry of the table contains the block identifier and its quality. On receiving a request, *Cache Manager* examines the table. When a cache miss occurs, it determines the quality of block to retrieve from a video server in accordance with the available bandwidth and requests. Then, it requests the server to send the block via an RTSP session established between them.

The server reads out a pair of a video block of the highest quality and a corresponding audio block through *Disk Manager*, adjusts the quality of the video block to the request using *Filter*, rebuilds a PS block by *Multiplexer*, and finally sends the block to the proxy via an RTP session in a TCP-friendly fashion.

*Cache Manager* obtains a pair of blocks through *RTP Receiver* and *Demultiplexer*. The block is sent to the client in a similar way to the block transfer from the video server to the proxy. At the same time, a pair of blocks is stored in *Hard Disk*. *TFRC* calculates TCP-friendly rate while receiving RTP packets from the server.

### 3.5 Block Prefetching Mechanism

*Cache Manager* is also responsible for prefetching blocks. If there exists any block whose quality is lower than the request in succeeding $P$ blocks, a prefetching mechanism is activated. The prefetch request is sent to a server in the same way as block retrieval, but the request has a lower priority than requests for retrieving cache-missed blocks at the server.

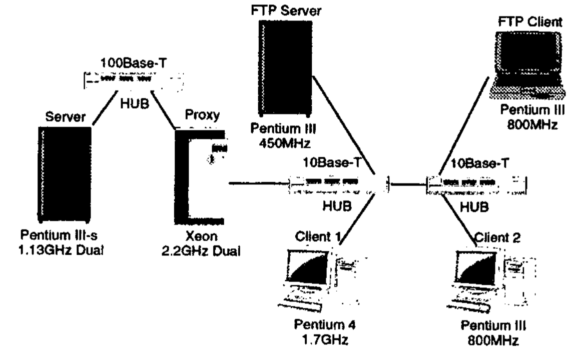



Figure 3. Configuration of Experimental System

### 3.6 Cache Replacement Mechanism

Replacing cached but less important blocks with a newly retrieved block is performed by *Cache Manager*. It determines, applies the quality adaptation to, and discards victims until a new block can be cached.

## 4. Evaluation

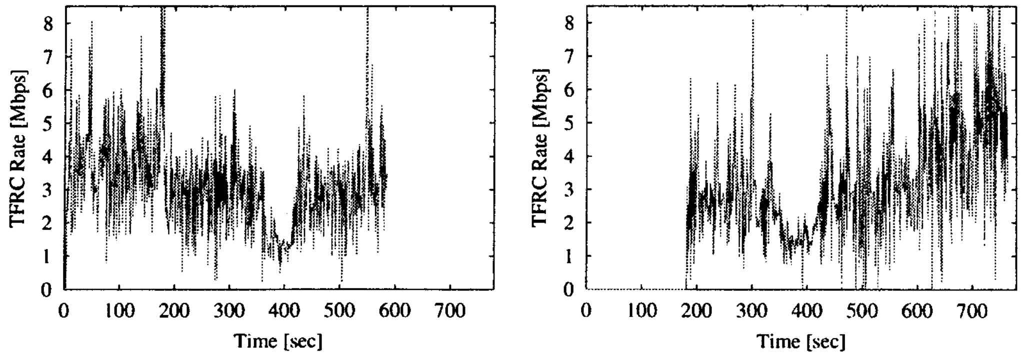
In this section, we conduct experiments to evaluate the rate variation, the video quality variation, and the overhead of quality adjustment.

Figure 3 illustrates a configuration of our experimental system. Two clients are connected to the proxy and watch the same video stream of 10 minutes from the beginning to the end without interactions such as rewinding, pausing and fast-forwarding. The video server has the whole video blocks of the highest quality of 8 Mbps. The proxy also has the whole video blocks, but the quality is 3 Mbps and a cache buffer capacity is limited to 450 MBytes. The prefetching window size $P$ is set to 5. There exists a TCP session for the file-transfer as a disturbance that competes with video sessions for the bandwidth.

Figure 4 illustrates variations in reception rates observed at the clients' *RTP Receiver*. At time 180 the client 2 begins a video session. From 360 to 420, the TCP session transfers a file. As shown in the figure, the video rate is regulated as the network condition changes, to avoid unexpected transfer delay and quality degradation that would be caused by congestions.

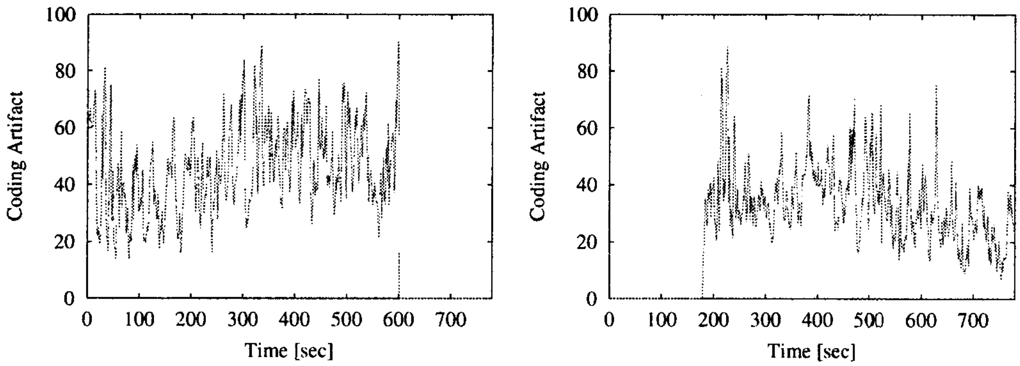
Figure 5 illustrates variations in the perceived video quality in terms of the coding artifact measured by VP2000A of KDD Media Will Corporation. A higher coding artifact value means that quality degradation is higher. This figure shows that the perceived video quality changes in accordance with the network condition. However, the fluctuation is smaller than that of the video rate (Fig. 4).

Through experiments, the maximum processing time of adjusting the quality of a video block was less than 0.5 sec. Since the proxy is required to process each block faster than one second, i.e., the interval between two consecutive requests in our experiments, more than 0.5 second can be devoted to the other tasks including retrieving a block from the video server. Thus, we can conclude that our mechanisms can accomplish a low-




(a) Client 1 (b) Client 2

Figure 4. Reception Rate Variation




(a) Client 1 (b) Client 2

Figure 5. Video Quality Variation

latency and high-quality video streaming service under a heterogeneous and dynamically changing environment.

## 5. Conclusion

In this paper, we implemented and evaluated proposed mechanisms in real system. Through the experiments, it is shown that our implemented system can continuously provide users with a high-quality and low-latency video distribution in accordance with the network condition.

As future research works, we should improve and extend our proposed mechanisms. For example, we should consider the case that several video streams are requested by clients in a larger network environment, mechanisms for interactive controls, and algorithms for further stable video quality.

**Acknowledgement**

This work is partly supported by Special Coordination Funds for promoting Science and Technology of the Ministry of Education, Culture, Sports, Science and Technology of Japan, and Telecommunication Advancement Organization of Japan.

**References**

[1] M. Hofmann, T. S. E. Ng, K. Guo, S. Paul, and H. Zhang, "Caching techniques for streaming multimedia over the Internet," *Technical Report BL011345-990409-04TM*, April 1999.

[2] K.-L. Wu, P. S. Yu, and J. L. Wolf, "Segment-based proxy caching of multimedia streams," in *Proceedings of the 10th International WWW Conference*, pp. 36–44, 2001.

[3] M. Sasabe, N. Wakamiya, M. Murata, and H. Miyahara, "Proxy caching mechanisms with video quality adjustment," in *Proceedings of ITCom 2001*, August 2001.

[4] H. Schulzrinne, A. Rao, and R. Lanphier, "Real Time Streaming Protocol (RTSP)," *Internet Request for Comments 2326*, Apr. 1998.

[5] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," *Internet Request for Comments 1889*, Jan. 1996.

[6] M. Handley, J. Padhye, S. Floyd, and J. Widmer, "TCP friendly rate control (TFRC): Protocol specification," *Internet draft draft-ietf-tsvwg-tfrc-04.txt*, Apr. 2002.

[7] 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.

[8] T. Yamada, N. Wakamiya, M. Murata, and H. Miyahara, "Implementation and evaluation of video-quality adjustment for heterogeneous video multicast," submitted to *APCC 2002*, April 2002.