PSC: A Priority Selected Cache Algorithm for Streaming Video over Internet

Shin-Hung Chang[+‡], Ray-I Chang[*], Jan-Ming Ho[+], and Yen-Jen Oyang[‡]
[+]Institute of Information Science, Academia Sinica, Taipei, Taiwan.
[‡] Dept. of Computer Science & Information Engineering, National Taiwan University, Taipei, Taiwan.
[*]Dept. of Information Management, National Central University, Chungli, Taiwan.
{viola@iis.sinica.edu.tw}

Abstract– **Proxy technology is commonly used at boundaries of ISPs (Internet Service Providers) to reduce the bandwidth requirement in the backbone WAN. By caching portions of a video in a video proxy closed to clients, the video playback quality can be dramatically improved and the problem of insufficient WAN bandwidth is eliminated. In the loss-less network environment, the *OC* (Optimal Cache) algorithm uses minimum cache storage in the video proxy and reduces the maximum bandwidth required in the backbone WAN. However, data packets may be lost to affect video playback quality while streaming video data through the Internet. Consider an MPEG video in which an I-frame is referenced by all other frames (B- or P-frames) in the same GOP (Group of Picture). Losing packets belonging to an I-frame makes it difficult to decode all of subsequent frames retrieved from the same GOP. The major goal of this paper is to select maximum video data from high-priority frames (I-frames) caching in the video proxy in order to defeat decoding error caused by packet loss and improve error recovery while serving QoS-guaranteed video playback. We propose a novel *PSC* (Priority Selected Cache) algorithm for solving this cache data selection problem. The *PSC* algorithm uses minimum cache storage in the video proxy and reduces maximum bandwidth requirement in the backbone WAN (as does the *OC* algorithm). Additionally, experiment results with several benchmark videos show that the *PSC* algorithm is 15% better than the conventional *OC* algorithm at caching I-frame data in a video proxy.**

I. Introduction

Due to advances in broadband technology, streaming services over the Internet have gained in popularity. Multimedia applications, such as digital library, video on demand and distance learning, require video streaming services to provide a more attractive and effective presentation. At present, various commercial products exist, including those from Microsoft Media, Real Media and Apple QuickTime technologies. The majority of these streaming products usually provide on-demand streaming services of poor-quality video content. However, with the rapid growth of streaming services, customers are becoming more and more sensitive to video playback quality. Poor-quality video content (with low bit rate) and small screen display have left current customers unsatisfied.

Because of high bandwidth requirement, a high-quality video is usually stored and streamed in a compressed format. Therefore, a compressed video naturally has the variable bit rate (VBR) property and its peak bit rate is generally much larger than its average bit rate, as shown in Table 2. Due to the burst nature of compressed video, it is a challenge to provide QoS-guaranteed video streaming services. Moreover, this problem is more complicated while streaming video data through the Internet.

Currently, Internet architecture is generally heterogeneous and consists of many Internet service providers (ISPs) as shown in Fig. 1. These ISPs interconnect through a backbone WAN owned by the third party. Each client accesses the Internet through an ISP via the access network. Typical examples of access networks include HFC, XDSL, ISDN, or LAN. Because the backbone WAN is shared by a large number of clients, network transmission quality is difficult to guarantee. Hence it is generally more costly to deliver data across the backbone WAN than across the access network.

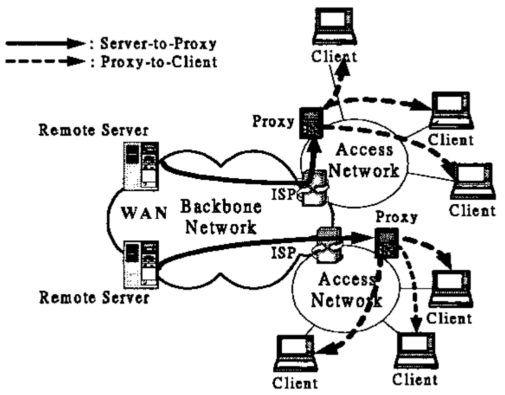


Fig. 1. Illustration of heterogeneous Internet with proxies installed.

To reduce the required external WAN bandwidth in video streaming applications, previous researchers have proposed two major technologies: video smoothing and video proxy.

1) *Video smoothing*. This technique flattens the bit rate fluctuation of the inter-frame by utilizing a client buffer, called smoothing buffer. By averaging the transmission rate of consecutive video frames, the end-to-end peak bandwidth (from server to client) can be dramatically reduced. Of course, the bandwidth requirement in the backbone WAN is also reduced. This issue has been well studied by previous researchers [5][6][7][8][9][10]. Using the optimal smoothing algorithm, one can obtain a minimum smoothing rate for streaming video across networks. However, if the external WAN bandwidth is small (less than minimum smoothing rate), the quality of the video transmission still cannot be guaranteed.

2) *Video proxy*. Proxy technology is widely used for a variety of services. Many proxies for handling video content were designed by several groups of researchers for different

purposes [1][2][3][4]. Because the amount of video data is usually huge, it is impossible to cache the entire video data in the video proxy. Among these different mechanisms, video staging, first proposed by Z. L. Zhang et al., caches only a pre-selected portion of the remote video data by using the cache storage in the video proxy [2]. By storing portions of a video data in a video proxy closed to clients, the impact of insufficient backbone WAN bandwidth is eliminated. They also proposed a linear time algorithm to handle the video staging problem. In this paper, we refer to this algorithm as the *CC* (Cut-off Cache) algorithm.

In a video streaming system, a large amount of video data is archived in the video server for on-demand services. The total amount of video data in the video server usually attains to a high Terabytes level. However, the *CC* algorithm usually caches too much video data and a large amount of storage space needs to be allocated in the video proxy. Therefore, we propose an *OC* (Optimal Cache) algorithm with linear time complexity [1] (*O(n)*, where *n* is the total frame number) to handle video staging. Subject to resource constraints (startup latency, client buffer size and allocated WAN bandwidth), the *OC* algorithm caches minimum video data in the video proxy and provides QoS-guaranteed video playback. In contrast, if the same size of storage to cache the video is allocated in the video proxy, the streaming schedule of video server (computed by the *OC* algorithm) reduces the maximum external WAN bandwidth requirement.

If network delivery over the backbone WAN is loss-less, the *OC* algorithm is the most cost-effective design to provide video streaming services for QoS-guaranteed video playback. However, data packets may be lost to affect the video playback quality while streaming video data through the Internet. Furthermore, the importance of each frame in a compressed video (e.g., MPEG video) differs. Consider an MPEG video in which an I-frame is referenced by all other frames in the same GOP (Group of Picture). Losing packets belonging to an I-frame makes it difficult to decode all of subsequent frames (B- or P-frames) retrieved from the same GOP. Therefore, video playback quality will degrade even faster due to heavy packet loss.

In this paper, we establish that I-frames are more important than other kinds of frames in a compressed video and must be given the highest-priority for caching. We propose a novel *PSC (Priority Selected Cache)* algorithm to solve this priority data selection problem. This algorithm uses the minimum amount of cache storage in the video proxy and reduces maximum bandwidth requirement in the backbone WAN (as does the *OC* algorithm). Furthermore, it determines a video pre-caching subset such that the total amount of I-frame cache data is maximal, subject to the minimum cache storage requirement. Experiment results with testing on several benchmark videos show that our proposed *PSC* algorithm improves the ratio of I-frame data cached in the video proxy by over 15% more than the conventional *OC* algorithm, a significant improvement.

The rest of this paper is organized as follows. Related works and problem formulations are described in section II. Our proposed algorithm is presented in section III. Analysis and simulation results are presented in section IV. Finally, we state the conclusion in section V.

II. Problem Formulations and Related Works

For clarification of the problem and the proposed algorithm, we state the following definitions. Video content V consists of a sequence of video frames $\{f_i > 0 \mid 0 \le i < n, f_{-1} = 0\}$, where f_i is the size of the $i-th$ video frame, and n is the total number of video frames. When this video V is requested, each video frame f_i is sequentially streamed to the client for playback. Additionally, the size of video V is denoted by $|V| = \sum_{i=-1}^{i=n-1} f_i$ and the amount of I-frame data in this video is denoted by $|V|_I = \sum_{i=-1}^{i=n-1} (f_i \times u_i)$, where $u_i = 1$ indicates that frame i is an I-frame; otherwise $u_i = 0$. At client side, the time period between receiving and playing the video is called startup latency, denoted by L. In this paper, we formulate the problem on the basis of a discrete time model. Let T_i represent the time period between playback of consecutive frames (f_{i-1} and f_i), where $0 \le i < n-1$. Without loss of generality, T_i is set as *1/frame rate* and the initialized value $T_0 = L$. The time instance of the $i-th$ frame playback is defined by $t_i = t_{i-1} + T_i$, where $0 \le i < n$ and $t_{-1} = 0$.

Let $S = \{r_i \mid 0 \le i < n\}$ represent a video streaming schedule of the remote video server, where r_i indicates the rate streaming the video data from the video server between the time instance t_{i-1} and t_i. r_{WAN} represents the allocated WAN bandwidth (the maximum rate used to deliver video content across the WAN, where $r_i \le r_{WAN}$). To simplify network resource management, we assume that network delivery service with minimum delay is used to stream video data across networks. Additionally, the available network bandwidth under the access network is assumed to be ample.

A sequence of the cached video data is represented by $C = \{c_i \ge 0 \mid -1 \le i < n, c_{-1} = 0\}$, where c_i indicates the cache size of the video frame i retrieved from the video proxy at the time instance t_i. The amount of cache size is denoted by $|C| = \sum_{i=-1}^{i=n-1} c_i$ and the amount of I-frame data cached in the video proxy is represented by $|C|_I = \sum_{i=-1}^{i=n-1} (c_i \times u_i)$.

In the Internet delivery service, the loss rate is much higher in the backbone WAN (from server to proxy) than in the local access network (from proxy to client). Therefore, we assume that the data loss rate in the access network is zero and the data loss rate in the backbone WAN is defined by $0 < e(r_{WAN}) < 1$. In this paper, we rough measure the playback quality by using the equation $Q = [1 - e(r_{WAN})] \times (\frac{|V|_I}{|C|} - \frac{|C|_I}{|C|}) + \frac{|C|_I}{|C|}$. Increasing the value of $\frac{|C|_I}{|C|}$ (the ratio of I-frame data cached in the video proxy) decreases the probability of a video decoding error and thus increases video playback quality. It is not difficult to extend this rough measurement function to consider different error control schemes. Finally, we define this problem on the *priority cache data selection* of a video as follows:

Problem: *Given a video, to determine a pre-caching subset C, such that the cache I-frame data $|C|_I$ is maximum, subject to the minimum cache data $|C|$, while startup latency, client buffer size, and allocated WAN bandwidth remain constant.*

A. Cut-off Cache (CC) Algorithm

The *CC* algorithm sequentially compares each video frame with the given cut-off rate (the allocated WAN bandwidth, r_{WAN}) [2]. If an entire frame cannot be transmitted by this cut-off rate in a frame period (the duration of each frame playback), the *CC* algorithm cuts the frame and stores the excessive portion of this frame in the video proxy, as shown in Fig. 2(a). The peak WAN bandwidth requirement is reduced from $max\{f_i/T_i\}$ to r_{WAN}, because part of a video is accessed from the nearby video proxy.

B. Optimal Cache (OC) Algorithm

The main idea behind the *OC* algorithm is to use unutilized WAN bandwidth from the *CC* algorithm to pre-fetch the subsequent video data, as shown in Fig. 2 (b). The detail elaboration of the *OC* algorithm is stated in [1].

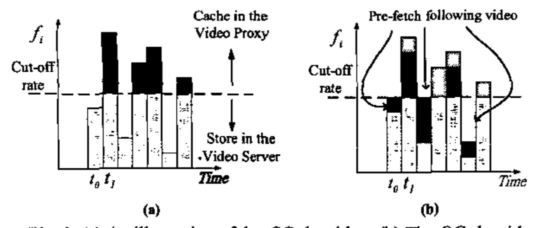


Fig. 2. (a) An illustration of the *CC* algorithm. (b) The *OC* algorithm uses unutilized WAN bandwidth to pre-fetch the subsequent video data.

III. Proposed Algorithms

A. Priority Selected Cache (PSC) Algorithm

In the *OC* algorithm, portion of a video frame is cached without consideration of the frame's priority. Therefore, the *OC* algorithm may store a large number of low-priority frame data (B- or P-frames) in the video proxy. Once packet loss occurs in I-frames, low-priority data cached in the video proxy will not be correctly decoded. We propose a *PSC* algorithm to solve this shortcoming in the *OC* algorithm. The *PSC* algorithm selects the maximum amount of the I-frame video data cached in the video proxy, subject to minimum cache storage (same as the *OC* algorithm).

The main function of the *PSC* algorithm is to exchange the cached data from a low-priority frame with video data from a high-priority frame (I-frame). A simple example is presented in Fig. 3 and Fig. 4 to illustrate this process of exchanging cached data.

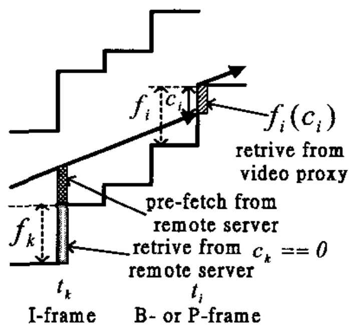


Fig. 3. With the buffer underflow occurring at time instance t_i, we should cache the video data, $f_i(c_i)$, in the video proxy for QoS-guaranteed video playback.

When the buffer underflow occurs at the time instance t_i, we should cache the video data ($f_i(c_i)$, with the size c_i) in the video proxy for QoS-guaranteed video playback, as shown in Fig. 3. Because frame *i* is not an I-frame, the *PSC* algorithm backtracks to search for an I-frame (frame *k*) from previous frames and caches the video data, $f_k(c_i)$, in the video proxy instead of the original video data, $f_i(c_i)$, as shown in Fig. 4.

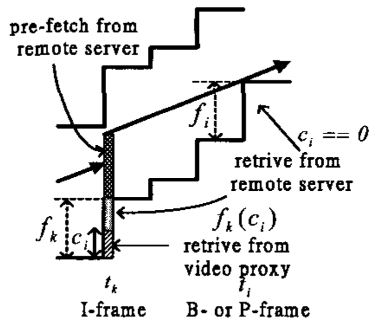


Fig. 4. Because the video data, $f_i(c_i)$, belongs to B- or P-frame, the *PSC* algorithm caches the video data $f_k(c_i)$ belonging to an I-frame *k*.

This exchanging process is not trivial and two tricky situations need to be considered. (1) Is the un-cached data of selected I-frame (in frame *k*) large enough for cache data exchanging (in frame *i*)? (2) Does buffer overflow occur while exchanging cache data?

Let $\Delta_k = f_k - c_k$ represent the size of the remaining video data in frame *k* to be cached in the video proxy. Without occurring buffer overflow, the available client buffer used for retrieving cached data from the video proxy at time instance t_k is denoted by $\nabla_k = \{B - b_j \mid k \le j < i\}$. The amount of data in frame *k* that can be exchanged is determined by $x = min\ \{\Delta_k,$

∇_k, $c_i\}$, where c_i is the size of original cached data in the frame i that is computed by the *PSC* algorithm, as shown in Fig. 5. This selection procedure is iteratively applied to other previous I-frames until either $\nabla_k = 0$ or $c_i = 0$. The detail of the *PSC* algorithm is presented as follows:

```
Algorithm: PSC (Priority Selected Cache) Algorithm
//Given a video $V = \{f_i > 0 \mid -1 \le i < n, f_{-1} = 0\}$
// $b_i$ is the client buffer occupancy at the time instance $t_i$.
//B indicates the size of client buffer;
//Given the allocated WAN bandwidth $r_{WAN}$;
(1) $i = -1$;  $b_i = 0$;
(2) repeat
(3) { $i = i + 1$;
(4)   $r_i = r_{WAN}$;
(5)   $b_i = min\{B, b_{i-1} + (r_i \times T_i) - f_{i-1}\}$;
(6)   if ($f_i \le b_i$) /*buffer is overflow*/
(7)   {$c_i = 0$; if ($b_i == B$) {$r_i = (B - b_{i-1} + f_{i-1})/T_i$;}}
(8)   else  /* buffer is underflow*/
(9)   { $c_i = f_i - b_i$; $k = i + 1$;
(10)    repeat /* select the most I-frames */
(11)    { $k--$;
(12)      if ((frame $k$ is an I-frame) && ($k != i$))
(13)      { /*available size for caching */
(14)        $x = min\{\Delta_k, \nabla_k, c_i\}$;
(15)        $c_k = c_k + x$; $c_i = c_i - x$; $\nabla_k = \nabla_k - x$; $\Delta_k = \Delta_k - x$;
(16)        Cache $f_k(x)$ in the video proxy;}
(17)    }until (($\nabla_k == 0$)||($c_i == 0$)&&($k != i$));} /*end of else*/
(18)  $b_i = f_i$; Cache $f_i(c_i)$ in the video proxy;
(19) }until (i > (n-1));
```

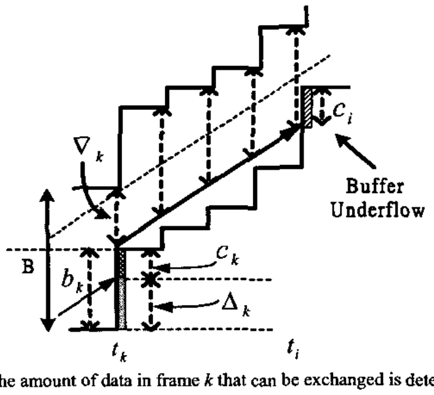


Fig. 5. The amount of data in frame k that can be exchanged is determined by $x = min\{\Delta_k, \nabla_k, c_i\}$.

B. A Fast Priority Selected Cache (F-PSC) Algorithm

The drawback of *PSC* algorithm is frequently backtracking to find the closest I-frame and compute the smallest available client buffer. This will cause the *PSC* algorithm requiring $O(n^2)$ computing complexity to finish the cache data selection, where n is the number of video frames. Therefore, we proposed a *F-PSC (Fast Priority Selected Cache)* algorithm to speed up the efficiency of the *PSC* algorithm by using a linked list, called *I-list*.

$I - list = \{(\Delta_w, \phi_w) \mid 0 \le w < n\}$ constantly keeps the location of each I-frame appearing in a video content while running the *F-PSC* algorithm. $\phi_w = min\ \{\nabla_j \mid w \le j < w^+\}$ records the available client buffer space for retrieving cached data in the video proxy from time instance t_w to t_{w^+-1}, where w^+ denotes the successor I-frame of frame w in the *I-list*, as shown in Fig. 6. If buffer underflow occurs in frame i at time instance t_i, then the *F-PSC* algorithm seeks the closest I-frame w by using the *I-list*. Meanwhile, we use ∇_{min} to keep track of the smallest available client buffer from frame w to frame i (the initialized value of ∇_{min} is the maximum number, denoted by ∞).

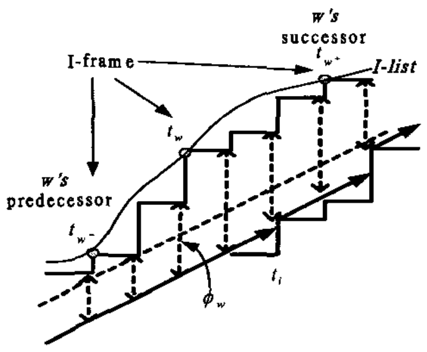


Fig. 6. A linked list, called *I-list*, is proposed to keep the location of each I-frame appearing in a video program.

The amount of data in frame w that can be exchanged is determined by $x = min\{\Delta_w, \nabla_{min}, c_i\}$. This exchanging process is sequentially applied to the next I-frame in the *I-list* until $c_i = 0$, $\nabla_{min} = 0$, or *I-list* is null. The *F-PSC* algorithm is constructed by rewriting step 5 and modifying steps 9~17 in the *PSC* algorithm. The detail of the *F-PSC* algorithm is presented as follows:

```
Algorithm: F-PSC (Fast Priority Selected Cache) Algorithm
Step 5 of the PSC algorithm is modified as follows:
/*create and maintain I-list */
(5.a) $b_i = min\{B, b_{i-1} + (r_i \times T_i) - f_{i-1}\}$;
(5.b) $\nabla_{min} = min\ \{B - b_i, \nabla_{min}\}$;
(5.c) if (frame $i$ is an I-frame)
(5.d) {$\Delta_i = f_i$; $\phi_{i^-} = \nabla_{min}$; $\nabla_{min} = \infty$;
(5.e) Insert frame $i$ in the I-list;}

Steps 9~17 of the PSC algorithm are modified as follows:
(9)  $c_i = f_i - b_i$;
```

```
(10) repeat /* select the most I-frames */
(11) { Delete the next I-frame w from I-list;
(12)   $\nabla_{min} = min\{\nabla_{min}, \phi_w\}; x = min\{\Delta_w, \nabla_{min}, c_i\};$
(13)   $c_w = c_w + x; c_i = c_i - x; \nabla_{min} = \nabla_{min} - x; \Delta_w = \Delta_w - x;$
(14)   Cache $f_w(x)$ in the video proxy;
(15) } until ($c_i == 0 \| \nabla_{min} == 0 \|$ I-list==null);
(16) if ($\Delta_w > 0$)
(17) {$\phi_w = \nabla_{min}$; Insert the frame w in the I-list;}
```

IV. Simulation Results

Section IV presents the simulation results of tests used to evaluate the effectiveness of our proposed *PSC* algorithm and compares its performance with previous methods. We test the *PSC* algorithm, conventional *OC* and *CC* algorithms by using several benchmark videos [11]. Encoding parameters of benchmark videos and parameters used in our experiments are described in Table 1. Statistics of these four video streams used in our experiments are presented in Table 2. Experiment results are evaluated according to four performance indices as follows:

(1) The proxy cache storage requirement
$=(|C|/|V|)\times 100\%$

(2) The WAN bandwidth utilization
$=(\sum_{i=-1}^{i=n-1}[(f_i - c_i)/T_i]/[r_{WAN}\times\sum_{i=-1}^{i=n-1}T_i])\times 100\%$

(3) The WAN bandwidth requirements
$=r_{WAN}/(|C|/|V|\times 100\%)$

(4) The percentage of cached I-frame video
$=(|C|_I/|C|)\times 100\%$

TABLE 1. Parameters used in our experiments.

Parameters	Values	Parameters	Values
Encoder Inputs	384x288	Frame Rate	24
Quantizer	I=10, P=14, B=18	Startup Latency	1sec
Encoding Patten	IBBPBBPBBPBB	Client Buffer	200kB

TABLE 2. Statistics of video streams used in our experiments.

Video Stream	Video Size (MB)	AVG Bit Rate (kbps)	Frame Size (kB)		
			MAX	AVG	STD
Star Wars	44.4088	218.278	15.24	1.14	1.58
Jurassic Park	62.36151	306.519	14.6	1.59	1.8
News	73.23109	359.945	23.18	1.87	2.38
James Bond	115.91179	596.73	29.86	2.97	3.14

A. Proxy Cache Storage Requirements

To increase system scalability in the installation of video proxies, the cache storage allocated for serving each video must be precisely controlled. The *PSC* algorithm caches minimum video data in the video proxy (as does the *OC* algorithm). Experiments on these benchmark videos show that our *PSC* algorithm on average reduces the cache storage requirement in the video proxy by over 30% more than the *CC* algorithm, if these benchmark videos are streamed using their average bit rate, as shown in Fig. 7.

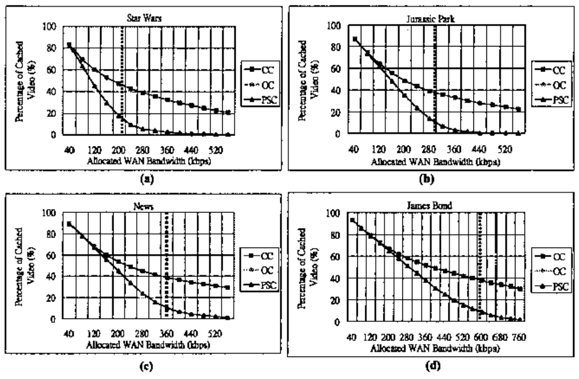


Fig. 7. Proxy cache storage requirement.
(a) Star Wars (b) Jurassic Park (c) News (d) James Bond.

B. Allocated WAN Bandwidth Utilization

In a distributed video streaming system, high bandwidth utilization implies that more user requests can be served simultaneously. Experiment results are presented in Fig. 8. By using the *PSC* algorithm, experiment results show that allocated WAN bandwidth utilization increases more than 30% on average (as does the *OC* algorithm), if these benchmark videos are streamed using their average bit rate.

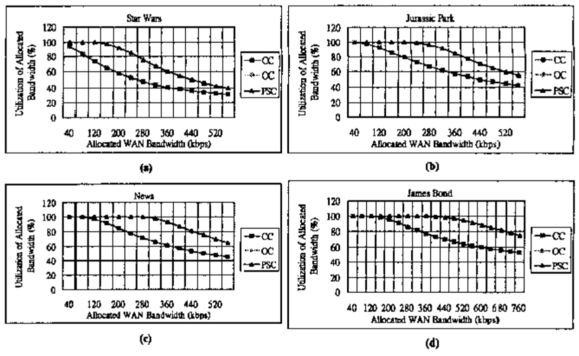
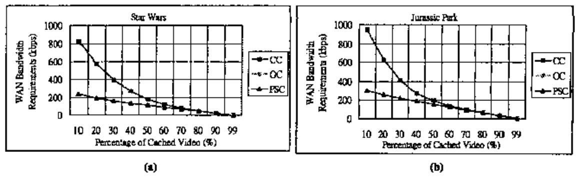


Fig. 8. Utilization of allocated WAN bandwidth.
(a) Star Wars (b) Jurassic Park (c) News (d) James Bond.

C. WAN Bandwidth Requirements

In Fig. 9, we present the WAN bandwidth requirement computed by the *PSC, OC and CC* algorithms when the cache storage space increases in the video proxy.



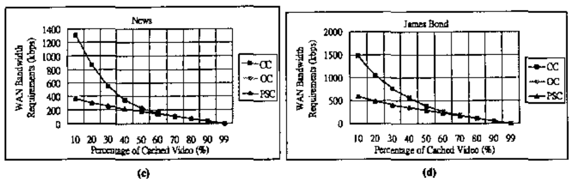


Fig. 9. WAN bandwidth requirements.
(a) Star Wars (b) Jurassic Park (c) News (d) James Bond.

Simulation results show that the *PSC* algorithm reduces the allocated WAN bandwidth requirement by more than 50% on average (as does the *OC* algorithm), when the cached data of a video is less than 20%.

D. *Percentage of Cached I-frame Video*

The cache priority of I-frames is set higher than other frames, the *PSC* algorithm can efficiently select maximum video data belonging to high-priority frames and cache them in the video proxy so as to minimize packet loss and improve error recovery. Because the *CC* algorithm requires too much cache storage, we only test the cached I-frame video computed by the *PSC* and *OC* algorithms. Experiments on these benchmark videos show that the *PSC* algorithm on average improves the ratio of I-frame video cached in video proxy by over 15% more than the conventional *OC* algorithm, as shown in Fig. 10.

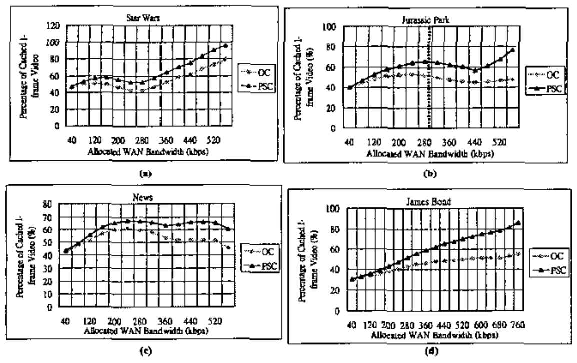


Fig. 10. Percentage of cached I-frame video.
(a) Star Wars (b) Jurassic Park (c) News (d) James Bond.

V. Conclusions

With advances in broadband technology, video streaming services over the Internet have gained in popularity. The majority of current commercial products in the streaming category usually provide either poor-quality video streaming over Internet or high-quality video streaming over LAN. Because of the insufficient bandwidth, it is difficult for service providers to stream high-quality videos across the Internet. By caching portions of a video in a video proxy closed to clients, the video content can be streamed across the WAN with CBR (constant-bit-rate) services. The *CC* algorithm is a good design at selecting the cached data of a video content. However, the *CC* algorithm usually caches too much video data in the video proxy. This requires a large amount of storage space in the video proxy. On the other hand, the *OC* algorithm caches the minimum amount of video data in the video proxy for QoS-guaranteed video playback. In the loss-less network environment, the *OC* algorithm is the most cost-effective approach to minimizing cache storage in the video proxy and bandwidth requirement in the backbone WAN. However, data packets may be lost to affect video playback quality while delivering video data through the Internet. When consider a compressed video, losing packets from a high-priority frame makes it difficult to decode all of subsequent frames. Therefore, in this paper, we propose a novel approach, called the *PSC* algorithm, to select the maximum video data from high-priority frames caching in the video proxy to defeat decoding error caused by packet loss while serving QoS-guaranteed video playback. From the experiment results, we show that the *PSC* algorithm uses minimum cache storage in the video proxy and reduces maximum bandwidth required in the backbone WAN (as does the *OC* algorithm). Additionally, the *PSC* algorithm also improves the ratio of I-frame data cached in video proxy by over 15% more than the conventional *OC* algorithm, a significant improvement.

References

[1] Shin-Hung Chang, Ray-I Chang, Jam-Ming Ho and Yen-Jen Oyang, *"OC: An Optimal Cache Algorithm for Video Staging,"* in proc. of IEEE International Conference on Networking 2002, (ICN'02).

[2] Zhi-Li Zhang, Yuewei Wang, David H. C. Du, and Dongli Su, *"Video Staging: A Proxy-Server-Based Approach to End-to-End Video Delivery over Wide-Area-Networks,"* in proc. of IEEE/ACM Transaction on Networking 2000.

[3] Wei-Hsiu Ma, and David H. C. Du, *"Reducing Bandwidth Requirement for Delivering Video Over Wide Area Network With Proxy Server,"* in proc. of IEEE International Conference on Multimedia and Expo 2000, (ICME'00).

[4] Zhourong Miao and Antonio Ortega, *"Proxy Caching for Efficient Services over the Internet,"* in proc. of *9th* International Packet Video Workshop 1999, (PVW'99).

[5] Subhabrata Sen, Jennifer Rexford, and Don Towsley, *"Proxy Prefix Caching for Multimedia Streams,"* in proc. of IEEE INFOCOM 1999.

[6] Ray-I Chang, M. Chen, Jan-Ming Ho, and M.T. Ko, *"An Effective and Efficient Traffic-Smoothing Scheme for Delivery of Online VBR Media Streams,"* in proc. of IEEE INFOCOM'99.

[7] Ray-I Chang, M. Chen, M. T. Ko, and Jan-Ming Ho, *"Designing the On-off CBR Transmission Schedule for Jitter-free VBR Media Playback in Real-time Networks,"* in proc. of IEEE RTCSA'97.

[8] Jennifer Rexford, Subhabrata Sen, and Don Towsley, *"Online Smoothing for Live, Variable-bit-rate Video,"* in proc. of NOSSDAV'97.

[9] Wu-Chi Feng and Jennifer Rexford, *"A Comparison of Bandwidth Smoothing Techniques for the Transmission of Prerecorded Compressed Video,"* in proc. of IEEE INFOCOM'97.

[10] J. Salehi, Zhi-Li Zhang, J. Kurose, and Don. Towsley, *"Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements through Optimal Smoothing,"* in proc. of ACM SIGMETRICS'96.

[11] http://www-info3.informatik.uni-wuerzburg.de/MPEG/traces/.