A Priority Selected Cache Algorithm for Video Relay in Streaming Applications

Shin-Hung Chang, Ray-I Chang, Jan-Ming Ho, and Yen-Jen Oyang

***Abstract*—With the popularization of Internet, many users can obtain various multimedia services through heterogeneous Internet. Among these services, video streaming application is the most challenging. Because of time constraint and variable bit rate (VBR) property of a video, the problem of streaming high quality video is insufficient external WAN bandwidth. Therefore, it is difficult to expand high quality video streaming services. To solve this problem, we proposed a novel video cache algorithm, called the Optimal Cache (OC) algorithm, for a relay video proxy. By caching portions of a video in a relay video proxy closed to clients, the video playback quality can be guaranteed and the problem of insufficient WAN bandwidth across Internet is eliminated. However, data packets are often lost while streaming video data across Internet, which downgrades video playback quality and even halts the video playback. In this paper, we refine the OC algorithm and propose a novel Priority Selected Cache (PSC) algorithm to select maximum high priority video data for caching in a relay video proxy. The PSC algorithm reduces the decoding errors caused by packet loss, improves error recovery, and provides QoS-guaranteed video playback. On the basis of experiment results with testing several benchmark videos, we show that the PSC algorithm caches at least 15% more high priority video data in a relay video proxy than conventional OC algorithm. Additionally, the PSC algorithm uses minimum storage in a relay video proxy and reduces the maximum bandwidth requirement in the WAN (as does the OC algorithm) subject to QoS-guaranteed video playback.**

***Index Terms*—Heterogeneous internet, relay video proxy, variable bit rate, video cache, video staging, video streaming.**

I. Introduction

DUE TO advances in broadband technology, media streaming services over the Internet have gained in popularity in recent years. Multimedia applications, such as digital libraries, video-on-demand and distance learning, require video streaming services to provide a more attractive and effective presentation than conventional text- or image-based services. Although there are various commercial products for video streaming, including those from Microsoft Media, Real Media and Apple QuickTime technologies, the majority of them 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. Consequently, they are increasingly dissatisfied with poor quality video content (with low bit rates) and small screen displays (computer screen).

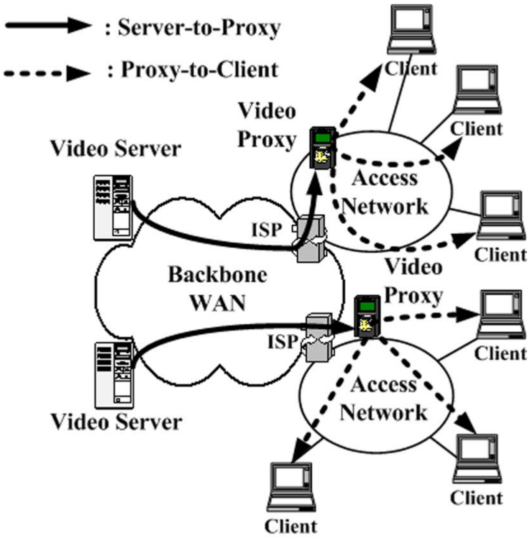

Fig. 1. Video streaming services with relay video proxies installed.

Because of high bandwidth requirement, a high-quality video is usually stored and streamed in a compressed format. A compressed video naturally has the variable bit rate (VBR) property and its peak bit rate is generally much higher than its average bit rate, as shown in Table II. The bursty nature of a compressed video causes it difficult to provide QoS-guaranteed video streaming services because external WAN bandwidth is insufficient [1]–[5], [7], [13], [16], [17], [29]. The problem becomes more complicated when streaming high quality video data through heterogeneous 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. To reduce the required external WAN bandwidth in video streaming applications, prior researchers have proposed two major techniques: video smoothing and web proxy.

Manuscript received August 5, 2003; revised October 17, 2006.

S.-H. Chang and J.-M. Ho are with the Institute of Information Science, Academia Sinica, Taipei, 10617 Taiwan (e-mail: viola@iis.sinica.edu.tw; hoho@iis.sinica.edu.tw).

R.-I. Chang is with the Dept. of Engineering Science, National Taiwan University, Taipei, 10617 Taiwan.

Y.-J. Oyang is with the Dept. of Computer Science and Information Engineering, National Taiwan University, Taipei, 10617 Taiwan (e-mail: yjoyang@csie.ntu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBC.2006.887170

1) Video smoothing. This technique flattens the bit rate fluctuation of the inter-frame by utilizing a client buffer, called a smoothing buffer. By averaging the transmission rate of consecutive video frames, the end-to-end peak bandwidth (from the video server to the client) can be reduced dramatically. Certainly, the external WAN bandwidth requirement is naturally reduced. This issue has been well researched already [6], [9]–[12], [14], [15], [19], [21]–[23], [28], [30]–[32]. Using the optimal video smoothing algorithm, one can obtain a minimum smoothing rate for streaming video across networks. However, if the external WAN bandwidth is too small (i.e., less than the minimum smoothing rate), the quality of video transmission still cannot be guaranteed.
2) Web proxy. Proxy technology has been widely used for improving the service quality and distributing contents, as shown in Fig. 1. For example, web content (e.g. hypertext, image data, and even small video) is entirely cached in a web proxy close to clients and end-users can retrieve this web content from the web proxy via the ample LAN access link without using external WAN bandwidth. By reducing content retrievals from the remote web server, the WAN bandwidth requirement decreases and the remote web server traffic is off-loaded [8], [18], [20], [24]. However, compared with the small size of web content, a high quality video is extremely huge. Caching an entire video in a web proxy to eliminate the WAN bandwidth requirement is unrealistic. Hence, it is impractical to apply the web proxy to directly handle video caching services.

For guaranteeing video playback quality and avoiding insufficient WAN bandwidth, the best way to stream high-quality video is to replicate a video server closed to clients at the access network. However, in a video streaming system, there are generally a large number of videos stored in the video server for on-demand services. The total amount of these video contents usually tends to a high Terabytes level. From the cost-benefit analysis, replicating video servers in all access networks is uneconomical and impractical. Therefore, a video proxy is proposed to install in the local access network for caching partial video contents. The main objective is to reduce the bandwidth requirement in the backbone WAN.

Many proxies with handling video contents were designed by several groups of researchers [25]–[27], [33]–[35], among which, the Video Staging mechanism, first proposed by Zhang *et al.*, caches only a pre-selected portion of a video data into a relay video proxy close to clients. Also, an algorithm to handle the Video Staging mechanism was presented in [35]. We refer to this algorithm as the CC (cut-off cache) algorithm in this paper. This CC algorithm is a one-pass algorithm and mainly concerned with comparing each frame sequentially in a video with a given cut-off rate (the external WAN bandwidth). 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 excessive portion of this video frame and stores it in the relay video proxy. However, the compressed video usually has a large size variation between video frames. Therefore, Zhang *et al.* proposed an enhanced CAS (cut-off after smoothing) algorithm to handle the Video Staging Mechanism. It is a two-pass algorithm that combines the CC algorithm and video smoothing technologies to further reduce the proxy storage and WAN bandwidth requirements. The CAS algorithm is the most effective algorithm in [35]. However, it is too complicated to handle the server streaming schedule and the client buffer control computed by the CAS algorithm.

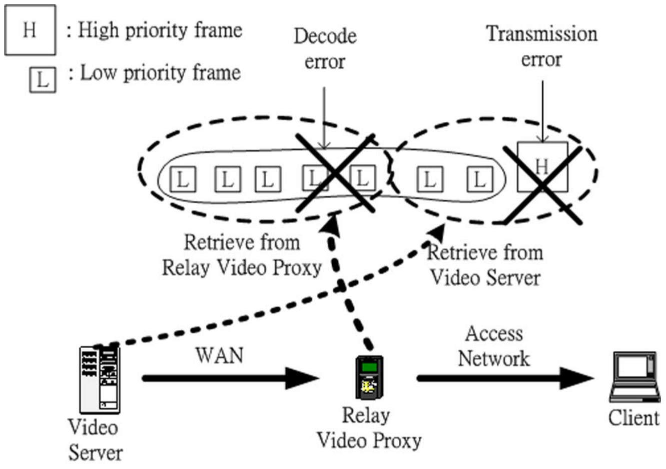

Fig. 2. The loss of packets belonging to a high priority frame (I-frame), denoted by $\boxed{\mathrm{H}}$, makes it incorrect to decode all subsequent low priority frames (B- or P-frames), denoted by $\boxed{\mathrm{L}}$, in the same GOP in a MPEG video.

To solve problems mentioned above, a one-pass algorithm, called Optimal Cache (OC) algorithm, is proposed to compute the subset of a video caching in a relay video proxy with linear time complexity ($O(n)$, where n is the number of frames), same as the CC algorithm. The proxy cache storage computed by the OC algorithm is minimal. This indicates that the OC algorithm caches the less video but reduces same WAN bandwidth requirement. If the equal amount of cache storage is given, the OC algorithm requires less WAN bandwidth than that computed by the CC or CAS algorithm to provide QoS-guaranteed video streaming services. We present that our OC algorithm is the most effective than previous conventional algorithms in the video proxy cache handling [26].

If network delivery is lossless, the OC algorithm is the most cost-effective design for video relay in terms of QoS-guaranteed video playback. However, data packets are transmitted across Internet and often lost, which downgrades video playback quality and even halts video playback. Furthermore, the importance of each frame in a compressed video (e.g., a MPEG video) is different. Consider a MPEG video in which an I-frame is referenced by all other frames in the same GOP (Group of Pictures) [2]. The loss of packets belonging to a high priority frame (I-frame), denoted by $\boxed{\mathrm{H}}$, makes it incorrect to decode all subsequent low priority frames (B- or P-frames), denoted by $\boxed{\mathrm{L}}$, in the same GOP, as shown in Fig. 2. Consequently, video playback quality is degraded even faster due to heavy packet loss.

In this paper, we set that I-frames are more important than other kinds of frames in a compressed video and must be given the highest priority for caching. Therefore, we propose a novel Priority Selected Cache (PSC) algorithm to solve this priority

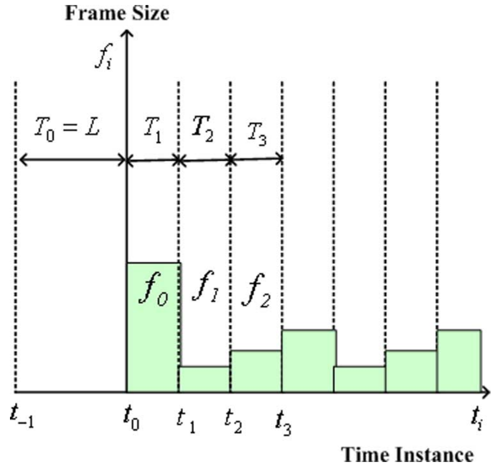

Fig. 3. A sequence of video frames $\{f_i \geq 0| -1 \leq i < n, f_{-1} = 0\}$, where f_i is the size of the i^{th} video frame.

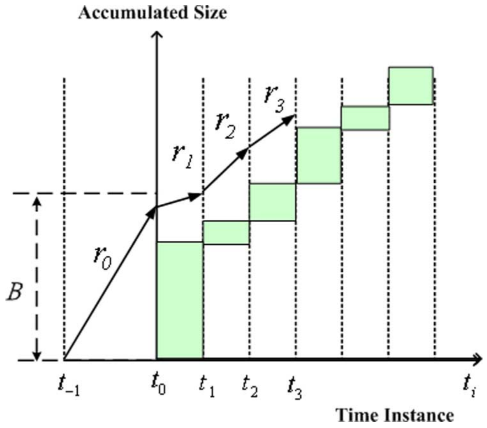

Fig. 4. The process of the video streaming is an accumulative transmission, as the cumulative amount of video data that received at time instance t_i. $S = \{r_i | 0 \leq i < n\}$ represent a video streaming schedule of a remote video server, where r_i indicates the rate of streaming the video data from the video server between time instance t_{i-1} and t_i.

data selection problem. The PSC algorithm calculates the minimum amount of cached video in the relay video proxy and reduces the maximum external WAN bandwidth requirement (as does the OC algorithm). Furthermore, it determines a video caching subset such that the total amount of high priority cached data is maximal, subject to the minimum cache storage requirement. Experiments with several benchmark videos show that our proposed PSC algorithm improves the ratio of high priority data cached in a relay video proxy by at least 15% more than the conventional OC algorithm.

The remainder of this paper is organized as follows. We discuss related works and problem formulations in Section II and present our proposed algorithm in Section III. Experiment results are presented and analysed in Section IV. Finally, in Section V, we present our conclusions.

II. Problem Formulation and Related Works

To clarify the problem and the proposed algorithm, we state the following definitions. Video content V consists of a sequence of video frames $\{f_i \geq 0| -1 \leq i < n, f_{-1} = 0\}$, where f_i is the size of the i^{th} video frame, and n is the total number of video frames, as shown in Fig. 3. When the video V is requested, each video frame f_i is sequentially streamed to the client for playback. Additionally, the size of a 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 $u_i = 0$. On the client side, the time period between receiving and playing the video is called "startup latency," denoted by L, as shown in Fig. 3. The size of client buffer (stores received video for playback in a client) is denoted by B, as shown in Fig. 4. In this paper, we formulate the problem on the basis of a discrete time model. Let $T_i$represent the time period between the playback of consecutive frames (f_{i-1} and f_i), where $0 \leq i < n - 1$. 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 $0 \leq i < n$ and $t_{-1} = 0$.

While a video request $V = \{f_i \geq 0| -1 \leq i < n, f_{-1} = 0\}$ is admitted for serving, a video server should sequentially retrieves video data from storage devices and sends them to the client buffer by a proper bandwidth. Therefore, the client can continuously display video frames received and starts to playback at time instance t_0. The time interval between adjacent frames is T_i. The process of the video streaming is an accumulative transmission, as the cumulative amount of video data that received at time instance t_i. Let $S = \{r_i | 0 \leq i < n\}$ represent a video streaming schedule of a remote video server, where r_i indicates the rate of streaming the video data from the video server between time instance t_{i-1} and t_i, as shown in Fig. 4. r_{WAN} represents the allocated external WAN bandwidth. To simplify network resource management, we assume that a network delivery service with minimum delay is used to stream video data across networks.

A sequence of the cached video data is represented by $C = \{c_i \geq 0| -1 \leq i < n, c_{-1} = 0\}$, where c_i indicates the cache size of video frame i retrieved from the relay video proxy at time instance t_i. The total size of cached video is denoted by $|C| = \sum_{i=-1}^{i=n-1} c_i$ and the amount of I-frame data cached in the relay video proxy is represented by $|C|_I = \sum_{i=-1}^{i=n-1} (c_i \times u_i)$. Increasing the value of $|C|_I/|C|$ (the ratio of I-frame data cached in the relay video proxy) decreases the probability of a video decoding error and thus increases video playback quality. We now define the problem of "priority cache selection" of a video.

Problem: Given a video, we compute a pre-cached subset C, such that the cached I-frame data $|C|_I$ is maximal, subject to the minimum amount of cached data $|C|$, while the startup latency, client buffer size, and external WAN bandwidth remain constant.

A. Cut-Off Cache (CC) Algorithm

Zhang *et al.* proposed a CC algorithm to handle Video Staging. The CC algorithm sequentially compares each video frame with a given cut-off rate [35]. If an entire frame cannot

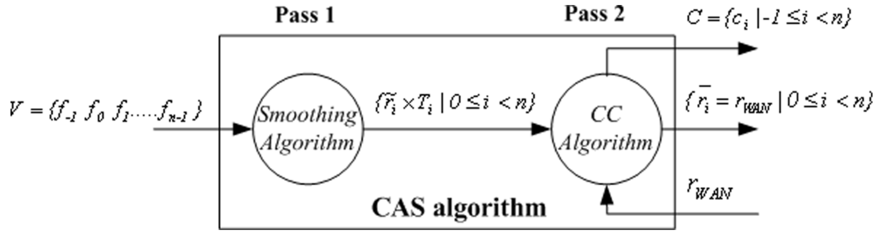
Pass 1
Pass 2
$C = \{c_i \mid -1 \le i < n\}$
$V = \{f_{-1}\ f_0\ f_1 \ldots f_{n-1}\}$
Smoothing Algorithm
$\{\widetilde{r}_i \times T_i \mid 0 \le i < n\}$
CC Algorithm
$\{\overline{r}_i = r_{WAN} \mid 0 \le i < n\}$
r_{WAN}
CAS algorithm

Fig. 5. Data flow diagram of operations in the two-pass CAS algorithm.

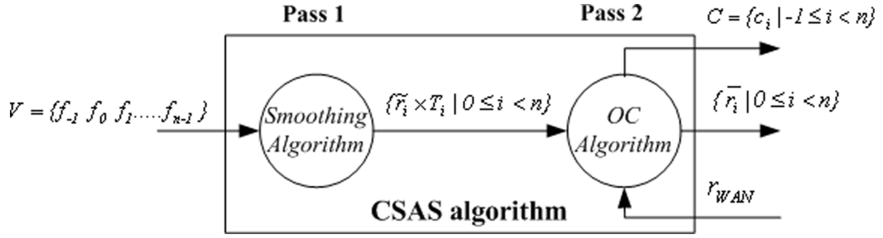
Pass 1
Pass 2
$C = \{c_i \mid -1 \le i < n\}$
$V = \{f_{-1}\ f_0\ f_1 \ldots f_{n-1}\}$
Smoothing Algorithm
$\{\widetilde{r}_i \times T_i \mid 0 \le i < n\}$
OC Algorithm
$\{\overline{r}_i \mid 0 \le i < n\}$
r_{WAN}
CSAS algorithm

Fig. 6. Data flow diagram of operations in the two-pass CSAS algorithm.

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 excess portion in the video proxy. The peak bandwidth requirement is reduced from $\max\{f_i/T_i\}$ to r_{WAN}, because part of the video is accessed from the nearby video proxy. The CC algorithm is a good design in system implementation. However, the compressed video usually has a large size variation between video frames. One frame size may be very small and it will result in the external WAN bandwidth not to be fully utilized. If this unutilized WAN bandwidth can be used to pre-fetch subsequent video data, the cached storage requirement in the video proxy will be reduced even more.

B. Cut After Smoothing Algorithm (CAS)

To solve the problem of unutilized WAN bandwidth, Zhang *et al.* proposed an enhanced Cut-off after Smoothing (CAS) algorithm to handle the Video Staging Mechanism. This is the most effective algorithm in [35]. It is a two-pass algorithm that combines the CC algorithm and video smoothing technologies to further reduce the requirement for video proxy storage and WAN bandwidth. In the two-pass process, the smoothing algorithm is run first, followed by the CC algorithm. In Fig. 5, we illustrate the data flow diagram of operations in the CAS algorithm. According to the experiment results in [35], the CAS algorithm is more effective than the CC algorithm. However, it is more complicated to handle the server streaming schedule and the client buffer control computed with the CAS algorithm than with the CC algorithm. The CAS algorithm is proposed by integrating the CC algorithm with the video smoothing technique: First, given a video content $V = \{f_{-1}\ f_0\ f_1\ \ldots\ f_{n-1}\}$, the video smoothing technique is applied and a smooth streaming schedule, $\widetilde{S} = \{\widetilde{r}_i | 0 \le i < n\}$, of the remote video server is computed. Second, the CC algorithm is applied and c_i is designed as $(\widetilde{r}_i - r_{WAN}) \times T_i$. If $c_i \le 0$, then c_i will be set to zero (none of the video data is retrieved from the video proxy at time index t_i). Additionally, the finial streaming schedule of the remote server in the CAS algorithm is $\{\overline{r}_i = r_{WAN} | 0 \le i < n\}$.

C. Optimal Cache (OC) Algorithm

A one-pass algorithm, called Optimal Cache (OC) algorithm, is proposed to compute the subset of a video caching in the video proxy with linear complexity ($O(n)$, where n is the number of frames), same as the CC algorithm. The key concept of the OC algorithm is to use the unutilized WAN bandwidth to pre-fetch the subsequent video data. Given a video content and specific resources, including the external WAN bandwidth, client buffer, and startup latency, the proxy cache storage computed by our OC algorithm is minimal. This indicates that the OC algorithm caches the less video but reduces same WAN bandwidth requirement. If the equal amount of cache storage is given, the OC algorithm requires less WAN bandwidth than that computed by the CC or CAS algorithm to provide QoS-guaranteed video streaming services. We present that our OC algorithm is the most effective than previous conventional algorithms in the video proxy cache handling [26].

D. Cache Selected After Smoothing (CSAS) Algorithm

In order to understand the effectiveness of adding smoothing algorithm, we also combine the smoothing algorithm [22] with the OC algorithm [26] to propose a Cache Selected after Smoothing (CSAS) algorithm in [27]. The main idea behind the CSAS algorithm is designed by integrating two processes, the video smoothing process and the OC process. In this algorithm, the client buffer consists of two parts, the smoothing buffer and the staging buffer. The smoothing buffer is designed to use in the video smoothing process and the staging buffer is designed to use in the OC process. In Fig. 6, we illustrate the data flow diagram of operations in the two-pass CSAS algorithm. There are five steps to run the CSAS algorithm. Detail descriptions are presented as follows. First, a video $V = \{f_{-1}\ f_0\ f_1\ \ldots\ f_{n-1}\}$ is fed into the smoothing process of the CSAS algorithm. Through the smoothing process, an end-to-end basis (from server to client) streaming schedule $\widetilde{S} = \{\widetilde{r}_i | 0 \le i < n\}$ (smoothed) is computed. The peak bandwidth of this streaming schedule is computed, denoted

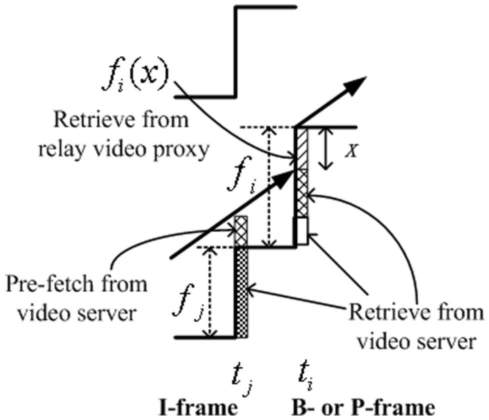
$f_i(x)$
Retrieve from relay video proxy
f_i
x
Pre-fetch from video server
f_j
Retrieve from video server
t_j
t_i
I-frame
B- or P-frame

Fig. 7. Frame f_i (B- or P-frame) is high priority video data and frame f_j (I-frame) is low priority video data. The buffer underflow occurs in frame f_i. On the basis of the OC algorithm, partial video data, $f_i(x)$, of frame f_i with size x will be selected to cache.

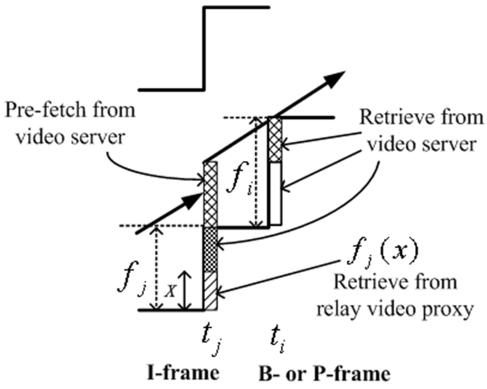
Pre-fetch from video server
Retrieve from video server
f_i
$f_j(x)$
Retrieve from relay video proxy
f_j
x
t_j
t_i
I-frame
B- or P-frame

Fig. 8. The PSC algorithm exchanges the cached data from a low priority frame with video data from a high priority frame. Therefore, the video data, $f_j(x)$, of frame f_j will retrieved from relay video proxy.

by $\widetilde{S}^*$. After the smoothing process, the regulation video data $\{\widetilde{r}_i \times T_i | 0 \leq i < n\}$ at the client buffer is formulated. Given the allocated WAN bandwidth, r_{WAN}, this regulation video data is fed into the OC process, where $r_{WAN} \leq \widetilde{S}^*$. The final streaming schedule $\{\overline{r}_i | 0 \leq i < n\}$ of the remote server is computed. The cached data set $C = \{c_i | -1 \leq i < n\}$ is computed after the OC process. From the experiment results in [27], we find that the effectiveness of the CSAS algorithm is very close to the OC algorithm based on different performance indices. The influence of adding smoothing algorithm is not conspicuous. Therefore, smoothing process only causes that the streaming schedule computed by the CSAS algorithm is difficult to control and implement.

III. Proposed Priority Selected Cache Algorithm

Because of video compression technology, there is the intraframe relation between the consecutive frames in a video. Generally, I-frame is more important than P- or B-frame in a MPEG video, because P- and B-frame will be decoded correctly with depending on I-frame. Therefore, the cache priority setting of each frame in a video should be different. In conventional algorithms, such as CC, CAS, OC and CSAS algorithms, a portion of a video frame is cached without considering the frame's priority. Therefore, a large amount of low priority frame data (B- or P-frames) will be stored in the relay video proxy. Once I-frame data needs to be transmitted from a video server across Internet and packet loss occurs in this I-frame data, however, the subsequent low priority P- or B-frame data retrieved from a video server or a relay video proxy will not be correctly decoded.

A. Priority Selected Cache (PSC) Algorithm

We refine the OC algorithm and propose a novel PSC algorithm to overcome this shortcoming. The PSC algorithm selects the maximum amount of high priority frame data (I-frame) cached in the relay video proxy server, subject to minimum cache storage (same as the OC algorithm).

The main idea of the PSC algorithm is to exchange the cached data from a low priority frame (B- or P-frame) with video data from a high priority frame (I-frame). A simple example presented in Fig. 7 and Fig. 8 illustrates this exchange process. In Fig. 7 and Fig. 8, we present a exchange example between two consecutive frames, f_i and f_j. Frame f_i (B- or P-frame) is low priority video data and frame f_j (I-frame) is high priority video data. The buffer underflow occurs in frame f_i at time instance t_i. On the basis of the OC algorithm, partial video data, $f_i(x)$, of frame f_i with size x will be selected to cache in the video proxy, as shown in Fig. 7. Originally, frame f_j is retrieved from the remote video server. However, the PSC algorithm will exchange the cached data from a low priority frame with video data from a high priority frame. Therefore, the video data, $f_j(x)$, of frame f_j will be retrieved from video proxy after processing exchange process, as shown in Fig. 8.

However, the exchange process is rarely applied between two consecutive frames. Generally, there are several frames in the middle of two frames which are needed to apply video data exchange. In Fig. 9 and Fig. 10, we present an example to illustrate the exchange process between two distant frames. When the buffer underflow occurs at time instance t_i, we originally cache the video data, $f_i(c_i)$, where $c_i = \alpha$, in the video proxy server to provide QoS-guaranteed video playback, as shown in Fig. 9. Because frame i is not an I-frame, the PSC algorithm backtracks to search for an I-frame (frame k) among previous frames and caches the video data, $f_k(c_k)$, where $c_k = \alpha$, instead of the original video data, $f_i(c_i)$, in the video proxy server, as shown in Fig. 10. After processing the exchange, the size of c_i is changed from α to zero and the size of c_k is modified from zero to α.

However, this exchange process is not trivial and two tricky situations need to be considered. (1) Is the uncached video data of the selected I-frame (in frame k) large enough for exchanging cached data of frame i? (2) Does buffer overflow occur while processing cached data exchange?

Let $\Delta_k = f_k - c_k$ represent the size of the remaining video data in frame k to be cached in the relay video proxy. Without incurring buffer overflow, the available client buffer that will retrieve cached data from the relay video proxy at time instance t_k is denoted by $\nabla_k = min\{B - b_x | k \leq x < i\}$, where B is client buffer size and b_x is the client buffer occupancy at time instance t_x. The amount of data in frame k that can be exchanged is determined by $\alpha = \min\{\Delta_k, \nabla_k, c_i\}$, where c_i is

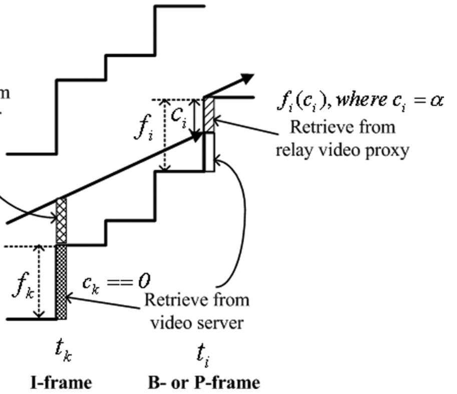
Pre-fetch from video server
$f_i(c_i)$, where $c_i = \alpha$
Retrieve from relay video proxy
f_i
c_i
f_k
$c_k == 0$
Retrieve from video server
t_k
t_i
I-frame
B- or P-frame

Fig. 9. With the buffer underflow occurring at time instance t_i, we should cache the video data, $f_i(c_i)$,where $c_i = \alpha$, in the video proxy server for QoS-guaranteed video playback.

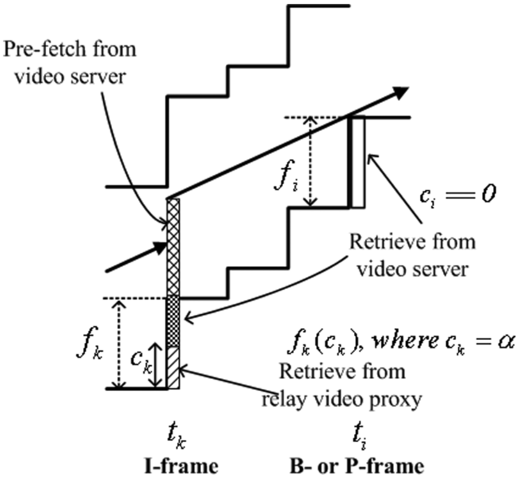
Pre-fetch from video server
f_i
$c_i = 0$
Retrieve from video server
f_k
c_k
$f_k(c_k)$, where $c_k = \alpha$
Retrieve from relay video proxy
t_k
t_i
I-frame
B- or P-frame

Fig. 10. Because the video data, $f_i(c_i)$, belongs to a B- or P-frame, the PSC algorithm caches the video data, $f_k(c_k)$, where $c_k = \alpha$, belonging to an I-frame k.

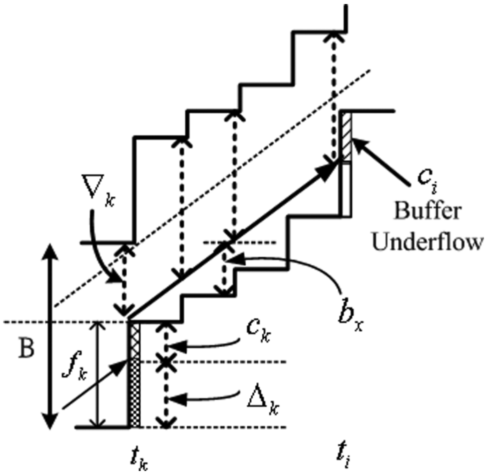

Fig. 11. The amount of data in frame k that can be exchanged is determined by $\alpha = min\{\Delta_k, \nabla_k, c_i\}$.

the size of the original cached data in the frame i computed by the PSC algorithm, as shown in Fig. 11.

This selection procedure is iteratively applied to other previous I-frames until either $\nabla_k == 0$ or $c_i == 0$. The details of the PSC algorithm are as follows:

Algorithm: Priority Selected Cache (PSC) Algorithm

```
//Given a video $V = \{f_i \geq 0| - 1 \leq i < n, f_{-1} = 0\}$;
//$b_i$ is the client buffer occupancy at time instance $t_i$;
//$B$ indicates the size of the client buffer;
//Given the allocated external WAN bandwidth $r_{WAN}$;
$i = -1; b_i = 0;$
repeat
  {$i = i + 1;$
  $r_i = r_{WAN};$
  $b_i = min\{B, b_{i-1} + (r_i \times T_i) - f_{i-1}\}$
  if $(f_i \leq b_i)$ /*buffer is overflow*/
    { $c_i = 0;$if $(b_i == B)\{r_i = (B - b_{i-1} + f_{i-1})/T_i;\}\}$ /*end of if*/
  else   /* buffer is underflow*/
    {  $c_i = f_i - b_i; k = i + 1;$
    repeat /* backtrack to select I-frames */
      {  $k - -;$
      if ( (frame $k$ is an I-frame) and $(k! = i)$)
        { /* determine the caching size $\alpha$ */
        $\alpha = min\{\Delta_k, \nabla_k, c_i\};$
        $c_k = c_k + \alpha; c_i = c_i - \alpha; \nabla_k = \nabla_k - \alpha; \Delta_k = \Delta_k - \alpha;$
        cache $f_k(\alpha)$ in the relay video proxy;} /*end of if*/
      }until $((\nabla_k == 0)$ or $(c_i == 0)$ and $(k! = i))$;
    } /*end of else*/
  $b_i = f_i$; cache $f_i(c_i)$ in the relay video proxy;
  } until (i > (n − 1)); /*end of repeat*/
```

B. A Fast Priority Selected Cache (FPSC) Algorithm

The drawback of the PSC algorithm is the frequent backtracking to find the closest I-frame and compute the smallest available client buffer. As a result, the PSC algorithm requires $O(n^2)$ computing complexity to finish the selection of high priority video data for caching, where n is the number of total video frames in the video program. We therefore propose a FPSC (Fast Priority Selected Cache) algorithm to speed up the PSC algorithm by using a stack data structure, called $I - linkstack$.

For clarify the relationship of I-frames in the $I - linkstack$, we use x^{-1} to denote the predecessor I-frame of frame x, as shown in Fig. 12. On the analogy of this rule, x^{-2} denotes the predecessor I-frame of the frame x^{-1} in the $I - linkstack$ and x denotes the predecessor I-frame of the frame x^{+1} in the

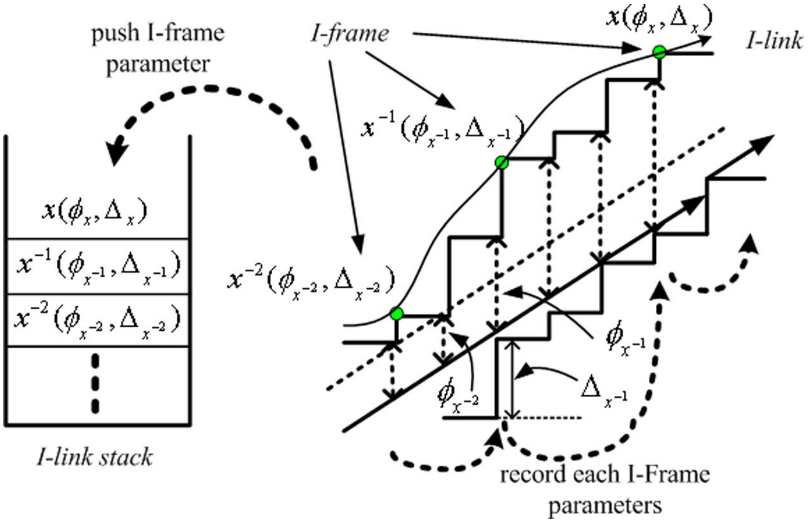
push I-frame parameter
I-frame
$x(\phi_x, \Delta_x)$
I-link
$x^{-1}(\phi_{x^{-1}}, \Delta_{x^{-1}})$
$x(\phi_x, \Delta_x)$
$x^{-1}(\phi_{x^{-1}}, \Delta_{x^{-1}})$
$x^{-2}(\phi_{x^{-2}}, \Delta_{x^{-2}})$
$x^{-2}(\phi_{x^{-2}}, \Delta_{x^{-2}})$
$\phi_{x^{-1}}$
$\Delta_{x^{-1}}$
$\phi_{x^{-2}}$
I-link stack
record each I-Frame parameters

Fig. 12. A stack, called the $I - link\ stack$, is proposed to track the location of each I-frame appearing in a video program.

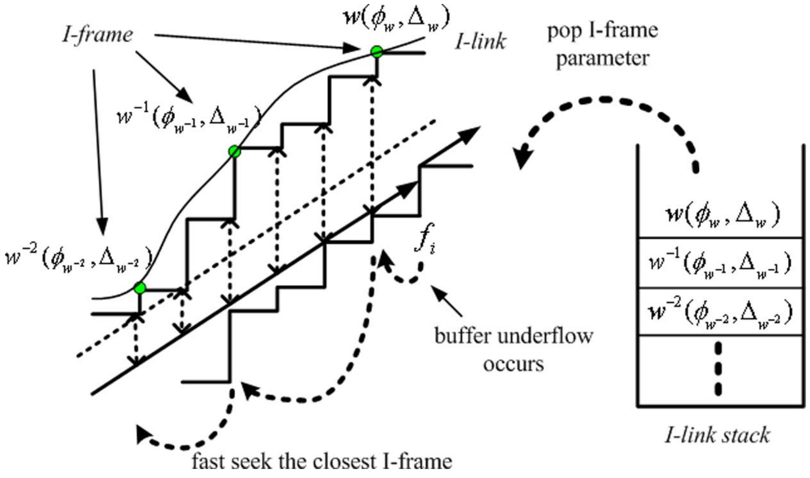
I-frame
$w(\phi_w, \Delta_w)$
I-link
pop I-frame parameter
$w^{-1}(\phi_{w^{-1}}, \Delta_{w^{-1}})$
$w^{-2}(\phi_{w^{-2}}, \Delta_{w^{-2}})$
f_i
buffer underflow occurs
$w(\phi_w, \Delta_w)$
$w^{-1}(\phi_{w^{-1}}, \Delta_{w^{-1}})$
$w^{-2}(\phi_{w^{-2}}, \Delta_{w^{-2}})$
I-link stack
fast seek the closest I-frame

Fig. 13. An illustration of fast priority selection cache with using the $I - link\ stack$.

$I - linkstack$. At each time instance t_x, $I - link\ stack$= $\{x(\Delta_x, \phi_x)|0 \le x < n, f_x\ is\ I - frame\}$ will constantly keeps the location of each I-frame x in a video program and records two corresponding parameters, Δ_x and ϕ_x, while running FPSC algorithm, as shown in Fig. 12. Meanwhile, $\phi_{x^{-1}}$= $min\{\nabla_y|x^{-1} \le y < x, f_{x^{-1}}\ and\ f_x\ are\ I - frame\}$ records the available client buffer space for retrieving cached data from the relay video proxy from time instance $t_{x^{-1}}$ to t_{x-1}. Additionally, Δ_x represents the size of the remaining video data in the I-frame x to be cached in a relay video proxy, mentioned in the last section.

In Fig. 13, we illustrate an example of fast I-frame selection in FPSC algorithm by using the $I - linkstack$. If buffer underflow occurs in frame i at time instance t_i, the FPSC algorithm will seek the closest I-frame fast, w, by using the $I - link\ stack$. In the FPSC algorithm, we use variable $\Phi = min\{\nabla_y|w \le y < i, f_w\ is\ I - frame\}$ to track smallest available client buffer from frame w to frame i and the initialized value of Φ is the maximum number, denoted by ∞. The amount of data in I-frame w that can be exchanged is determined by $\alpha = min\{\Delta_w, \Phi, c_i\}$, when buffer underflow occurs at frame i. If the amount α is not enough for exchanging, the FPSC algorithm will continuously exchange cached video data from the predecessor I-frame w^{-1}. This exchange process is sequentially applied to the next I-frame $(w^{-1}, w^{-2}, w^{-3} \ldots)$ in the $I - link\ stack$ until c_i is equal to zero, Φ is equal to zero, or the $I - link\ stack$ becomes empty. The FPSC algorithm is constructed by modifying Step1, rewriting Step 5 and modifying Steps 9∼18 in the PSC algorithm. The detail modifications are shown as follows:

```
Algorithm: Fast Priority Selected Cache (FPSC) Algorithm

Step 1 of the PSC algorithm is modified as follows:
(1) $i = -1$; $b_i = 0$; $\Phi = \infty$; push an empty I-frame parameter
into $I - link\ stack$;

Step 5 of the PSC algorithm is modified as follows:
/*create and maintain I-link */
(5.a) $b_i = min\{B, b_{i-1} + (r_i \times T_i) - f_{i-1}\}$;
(5.b) $\Phi = min\{B - b_i, \Phi\}$;
```

TABLE I
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

```
(5.c) if (frame $i$ is an I-frame)
(5.d) { $\Delta_i = f_i$; $\phi_{i-1} = \Phi$; $\Phi = \infty$;
(5.e)  push parameter of I-frame $i$ into the $I - link\ stack$;}
```

Steps 9∼18 of the PSC algorithm are modified as follows:

```
(9) $c_i = f_i - b_i$;
(10) repeat /* select the most I-frame data */
(11) { pop the parameter of I-frame $w$ from the $I - linkstack$;
(12)  if ($I - link! = $ null)
(13)  { $\Phi = min\{\Phi, \phi_w\}$; $\alpha = min\{\Delta_w, \ \Phi, c_i\}$;
(14)   $c_w = c_w + \alpha$; $c_i = c_i - \alpha$; $\Phi = \Phi - \alpha$; $\Delta_w = \Delta_w - \alpha$;
(15)   cache $f_w(\alpha)$ in the relay video proxy;}
(16) }until($c_i == 0, \Phi == 0,$ or $I - link\ stack ==$ null);
(17) if ($\Phi ! = 0$)
(18) { if ($\Delta_w > 0$)
(19)  { $\phi_w = \Phi$; /*update the parameter $\phi_w$ of I-frame $w$; */
(20)    push the parameter of I-frame $w$ into the
        $I - link\ stack$;}}
(21) else
(22) { clear up $I - link\ stack$; $I - link =$ null;}
```

IV. EXPERIMENT RESULTS AND ANALYSIS

In this section, we present the simulation results of tests used to evaluate the effectiveness of our proposed PSC algorithm and compare its performance with previous algorithms. We test the PSC, the OC, the CC, the CAS, and the CSAS algorithms on several benchmark videos [36]. The encoding parameters of the benchmark videos and the parameters used in our experiments are described in Table I, while the statistics of the four video streams used in our experiments are presented in Table II. The experiment results are evaluated according to the following four performance indices:

1) The cache storage requirement in the $\mathrm{relay\ video\ proxy} = (|C|/|V|) \times 100\%$
2) The external WAN bandwidth utilization= $(\sum_{i=0}^{i=n-1}(r_i/r_{WAN})/n) \times 100\%$
3) $\mathrm{The\ external\ WAN\ bandwidth\ requirements} = (|V| - |C|)/\sum_{i=0}^{i=n-1} T_i$
4) The percentage of cached I-frame video in the $\mathrm{relay\ video\ proxy} = (|C|_I/|C|) \times 100\%$

A. Cache Storage Requirements in the Relay Video Proxy

A video server for providing on-demand services usually stores a huge number of videos. The total amount of these video contents usually tends to a high Terabytes level. From the cost-benefit analysis, it is uneconomical and impractical to replicate video servers in all access networks for QoS-guaranteed video playback. Therefore, a relay video proxy is proposed to install in the local access network for caching partial video contents. Furthermore, if minimum cache data of each video is allocated in a relay video proxy, the total cache storage required to build a relay video proxy will be dramatically reduced. The relay video proxy will be easy-to-install and the scalability of streaming video service will be extensively distributed.

In this section, on each benchmark video, the different storage requirement computed by the CC, CAS, CSAS, OC and PSC algorithms are presented. In Fig. 14, we will show which algorithm caches the smallest portion of a video in the relay video proxy by experiment subject to QoS-guaranteed video playback. Given the same resources, we present the relationship between the cached percentage of each benchmark video and the variation of the external WAN bandwidth. Based on the average bit rate of each benchmark video, we present the percentage variation of cached video in the ±200 kbps variation of external WAN bandwidth.

When the external WAN bandwidth increases, the storage requirement of each benchmark video computed by these algorithms decreases. In Fig. 14, take video "Star War" for example, we show that the PSC algorithm caches 17% of this video data in the relay video proxy and the CC algorithm caches 48% of the video data in the relay video proxy, when we stream this video with using its average bit rate 218 kbps. On average, the cached storage requirement of each benchmark video is reduced by more than 31%, while using the PSC algorithm. In experiment results on the four benchmark videos, we show the experimental curve of the PSC algorithm is very close to the curves of the OC algorithm. Therefore, in the cache storage requirement, the PSC algorithm approaches optimal. Additionally, the decreasing slope of experimental curve computed by the PSC algorithm is sharper than those computed by the CC and CAS algorithms, when we stream these benchmark videos with using more external WAN bandwidth than their average bit rates. The larger the frame variation, the later the experiment curves will meet. Hence, the PSC algorithm reduces the cache storage even further when the external WAN bandwidth is more sufficient, particularly for a video with large frame size variations.

B. External WAN Bandwidth Utilization

Currently, the cost of using external WAN bandwidth is high compared to that of using bandwidth in access network. In a distributed video-streaming system, high bandwidth utilization implies that more video requests can be served simultaneously. A good system design for streaming services should utilize the external WAN bandwidth as far as possible at all times so as to save cost. In this section, we present the relationship between the percentage of WAN bandwidth utilization and the allocated WAN bandwidth. Through simulation, we stream the four benchmark videos with their distinct streaming schedules computed by CC,

TABLE II
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

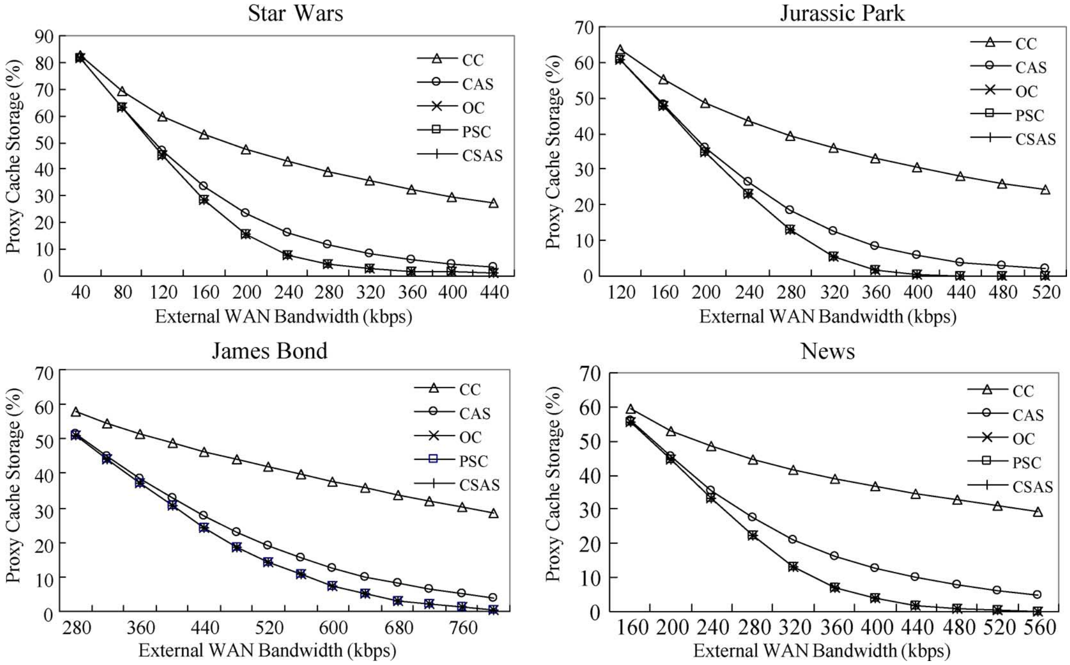

Fig. 14. Cache storage requirements in a relay video proxy: (a) Star Wars; (b) Jurassic Park; (c) James Bond; (d) News.

CAS, OC, CSAS, and PSC algorithms. In Fig. 15, the experimental results are presented to show the utilization of the external WAN bandwidth for streaming four benchmark videos distinctly. We mainly observe the variation of bandwidth utilization in the ±200 kbps of their specific average bit rate.

According to the experiment results, the WAN bandwidth utilization percentage decreases when the external WAN bandwidth increases. This indicates that the higher the external WAN bandwidth, the more waste of bandwidth there is. Because of frame size variation, the bandwidth utilization of the CC algorithm without considering client buffer control decreases rapidly. Additionally, because the buffer is limited and causes overflow in streaming a video data, WAN bandwidth utilization cannot reach 100% all the time by using the CAS, OC, CSAS, and PSC algorithms. From the experiment result of video "Star War," we find that the PSC algorithm utilizes 85% WAN bandwidth, the CAS algorithm utilizes 76% WAN bandwidth, and the CC algorithm only utilizes 51% WAN bandwidth, while streaming this video with its average bit rate. On average, the bandwidth utilization of the PSC algorithm increases more 34% than that of the CC algorithm and more 25% than that of the CAS algorithm, if each benchmark video is streamed with its average bit rate. Additionally, the results of the PSC algorithm are close to that of the OC algorithm in four benchmark videos. We conclude that the PSC algorithm is effective in avoiding the waste of external WAN bandwidth.

C. External WAN Bandwidth Requirements

If the cache storage in the relay video proxy is limited, how much bandwidth of WAN bandwidth is needed to provide QoS-guaranteed video streaming services with relay video proxy install. In this section, we present the relationship between the WAN bandwidth requirement and the percentage of video cached. It is obviously that the external WAN bandwidth requirement decreases as the video cache percentage increases. In Fig. 16, the experimental curve is sharper when the percentage of cached video is low. This indicates that the smaller the cache, the more effective the bandwidth reduction.

In Fig. 16, we present the WAN bandwidth requirement computed by the CC, CAS, OC, CSAS, and PSC algorithms when the video proxy storage increases. Compared with the CC algorithm, the PSC algorithm, on average, reduces the external WAN bandwidth requirement by more than 50% when the cached data of a video is less than 25%. Additionally, compared with the CAS algorithm, the PSC algorithm can, on average, reduce the allocated WAN bandwidth by more than 15% when the cached data of a video is less than 15%.

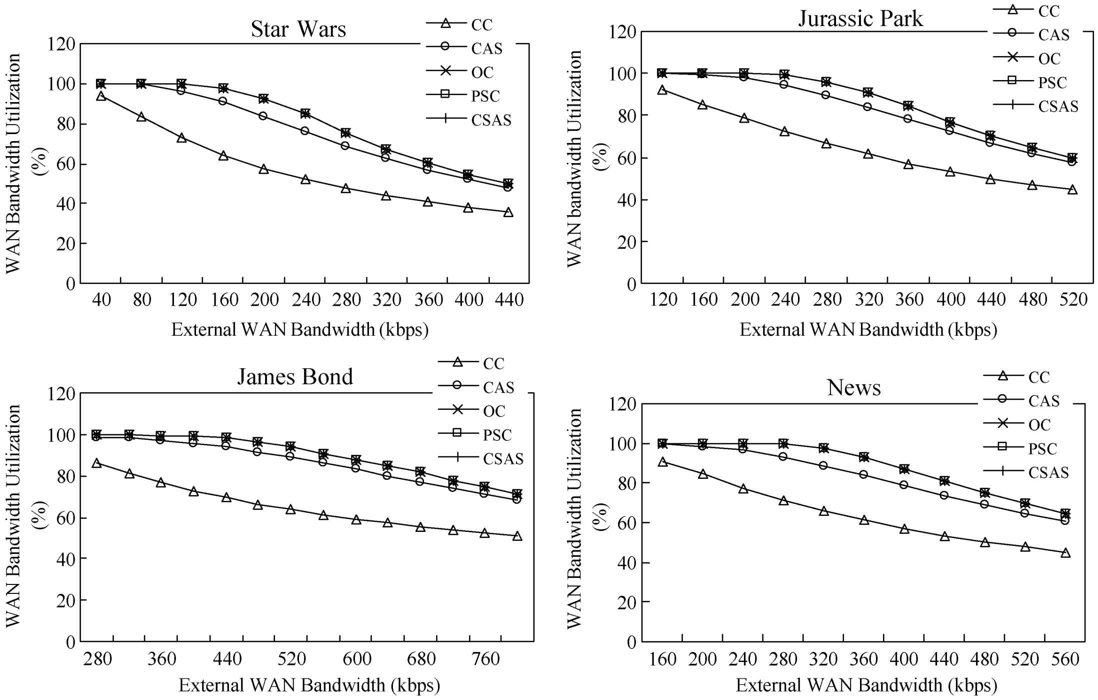

Fig. 15. Utilization of the external WAN bandwidth: (a) Star Wars; (b) Jurassic Park; (c) James Bond; (d) News.

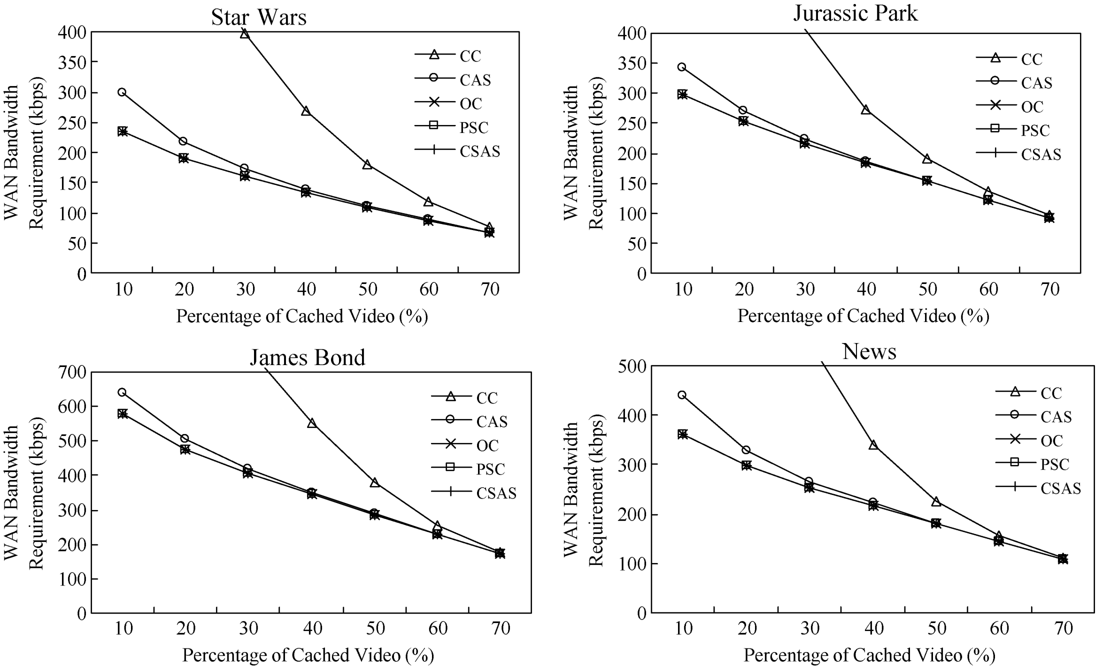

Fig. 16. The external WAN Bandwidth requirements: (a) Star Wars; (b) Jurassic Park; (c) James Bond; (d) News.

D. *Percentage of Cached I-Frame Video*

On the basis of each frame with different importance in a compressed video, we give high priority to the I-frame video data for been cached in the relay video proxy. In this section, we present the relationship between the percentage of cached I-frame video data and the variation of external WAN bandwidth. In Fig. 17, we find that the higher the external WAN bandwidth is allocated, the more I-frame video data will be cached by these two algorithms. We want to present which algorithm caches the most high priority video data into the relay video proxy.

First of all, we want to explain why we only compare the experimental results of OC and PSC algorithm in this experiment. From the experiment results in Section IV-A, we find that the CC and CAS caches too much video data in the relay video

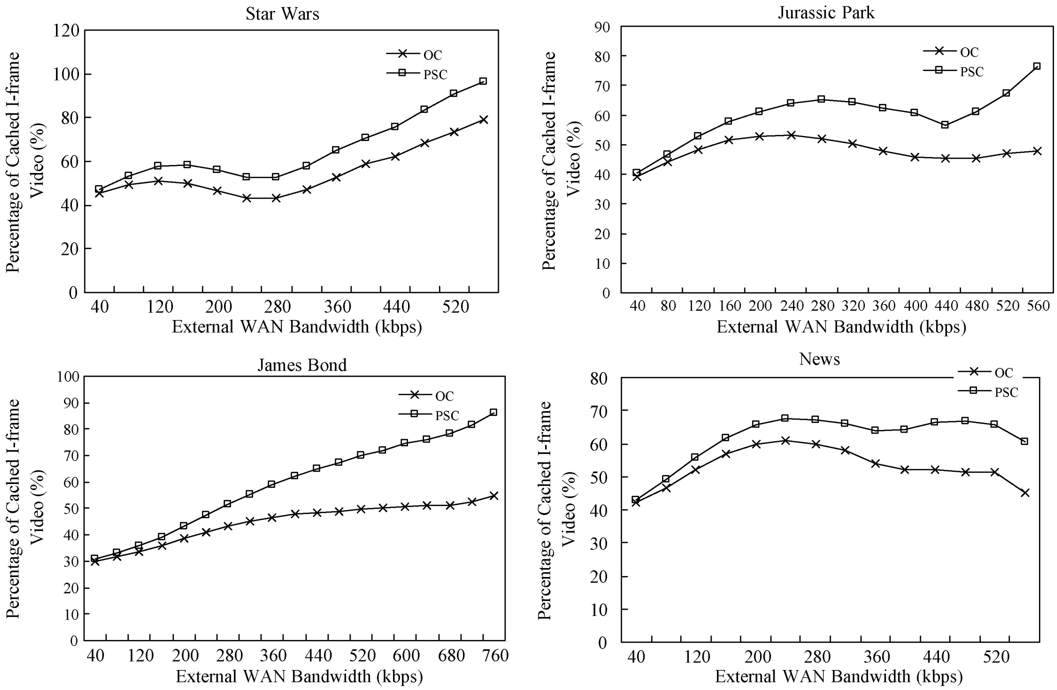

Fig. 17. Percentage of cached I-frame video: (a) Star Wars; (b) Jurassic Park; (c) James Bond; (d) News.

proxy, so we only test the cached I-frame video data computed by the PSC and OC algorithms to present the effectiveness of the PSC algorithm based on the equal amount of total cached data. In Fig. 17, take video "Star War" for example, we show that 39% of the cached video computed by the OC algorithm and 55% of cached video computed by the PSC algorithm are high priority video data (I-frame) in the relay video proxy, when we stream this video with using its average bit rate 218 kbps.

Experiments on the benchmark videos show that, on average, the PSC algorithm improves the ratio of I-frame video cached in a relay video proxy by more than 15% compared to the conventional OC algorithm. The PSC algorithm effectively selects the maximum amount of high priority frame data and caches it in the relay video proxy so as to minimize packet loss and improve error recovery. Additionally, the PSC algorithm gathers the cached data together belonging to a smallest set of I-frame video data. If the frame is a unit of cache video data, the PSC algorithm will cache the smallest number of I-frames into the relay video proxy subject to QoS-guaranteed video playback. This will causes that the PSC algorithm is easy to implement in video streaming applications.

V. Conclusions

In recent years, due to advances in broadband technology, video streaming services across the Internet have gained in popularity. Most current commercial streaming products only provide poor quality video streaming (with low bit rate). Because of insufficient bandwidth, it is difficult for service providers to stream high quality videos across the Internet. However, by caching portions of a video in a relay video proxy closed to clients, the video content can be streamed with CBR (constant-bit-rate) services. The conventional CC algorithm is a good design for selecting the cached data of each video into the relay video proxy. However, the CC algorithm usually caches too much video data in the relay video proxy. It causes that the relay video proxy requires a large amount of storage space for video caching. On the other hand, the OC algorithm is optimal to cache the minimum amount of video data in the relay video proxy for QoS-guaranteed video playback. In a lossless network environment, the OC algorithm is the most cost-effective approach to minimizing cache storage in the relay video proxy and the external WAN bandwidth requirement. However, data packets often lost while delivering video data across the Internet and the video playback quality is seriously affected. Additionally, because of high bit rate requirement, high quality video is generally in a compressed format. While streaming this high quality video, once the loss of packets belonging to a high priority frame occurs, it will be difficult to decode all subsequent low priority frames. The quality of playback will downgrade very fast. In this paper, we therefore propose the PSC algorithm, a novel approach that selects the maximum amount of video data from high priority frames and caches this selected video data in the relay video proxy. Experiment results on several benchmark videos show that the PSC algorithm uses the minimum amount of storage space in a relay video proxy and reduces the maximum bandwidth (as does the OC algorithm). Additionally, the PSC algorithm also improves the ratio of I-frame data cached in a relay video proxy by more than 15% compared to the conventional OC algorithm. With using the PSC algorithm, decoding errors caused by packet loss will be reduced and video playback quality will be guaranteed. Furthermore, if the frame is a unit of cache video data, the PSC algorithm will gathers the cached video data together belonging to a smallest set of I-frames. This property of the PSC algorithm causes that video caching process is easy to implement for video streaming applications.

REFERENCES

[1] C. C. Han and K. G. Shin, “Scheduling MPEG-compressed video with firm deadline constraints,” in *Proceedings of ACM Multimedia*, 1995.

[2] D. L. Gall, “MPEG: A video compression standard for multimedia applications,” *ACM Communications*, 1991.

[3] E. W. Knightly, D. E. Wrege, J. Liebeherr, and H. Zhang, “Fundamental limits and tradeoffs of providing deterministic guarantees to VBR video traffic,” in *Proceedings of ACM SIGMETRICS*, 1995.

[4] G. D. Stamoulis, M. E. Anagnoustou, and A. D. Georgantas, “Traffic source models for ATM networks,” *Computer Communications*, 1994.

[5] H. Zhang and E. W. Knightly, “A new method to support delay-sensitive VBR video in packet-switched networks,” in *Proceedings of International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, 1995.

[6] ——, “RED-VBR: a renegotiation-based approach to support delay sensitive VBR video,” *ACM Multimedia Systems Journal*, 1997.

[7] I. Dalgic and F. A. Tobagi, “Performance evaluation of ATM networks carrying constant and variable bit-rate video traffic,” *IEEE Journal of Selected Areas in Communications (JSAC)*, August 1997.

[8] I. Kim, H. Y. Yeom, and J. Lee, “Analysis of buffer replacement policies for WWW proxy,” in *Proceedings of the 20th International Conference on Information Networking*, 1998.

[9] J. M. McManus and K. W. Ross, “Video on demand over ATM: Constant-rate transmission and transport,” in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 1996.

[10] J. Salehi, Z. L. Zhang, J. Kurose, and D. Towsley, “Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing,” in *Proceedings of the ACM SIGMETRICS*, 1996.

[11] J. Rexford, S. Sen, and D. Towsley, “Online smoothing for live, variable-bit-rate video,” in *Proceedings of the International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, 1997.

[12] K. Zhu, Y. Zhuang, and Y. Viniotis, “Achieving end-to-end delay bounds by EDF scheduling without traffic shaping,” in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2001.

[13] M. Garrett and W. Willinger, “Analysis, modeling and generation of self-similar VBR video traffic,” in *Proceedings of ACM SIGCOMM*, 1994.

[14] M. Grossglauser, S. Keshav, and D. Towsley, “PCBR: a simple and efficient service for multiple time scale traffic,” in *Proceedings of ACM SIGCOMM*, 1995.

[15] M. Grossglauser and S. Keshav, “On CBR service,” in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 1996.

[16] M. Krunz and S. K. Tripathi, “On the characteristics of VBR MPEG streams,” in *Proceedings of ACM SIGMETRICS*, June 1997.

[17] P. Pancha and M. E. Zarki, “MPEG coding for variable bit rate video transmission,” *IEEE Communications Magazine*, 1994.

[18] P. Cao and S. Irani, “Cost-aware WWW proxy caching algorithms,” in *Proceedings of USENIX Symposium on Internet Technologies and Systems*, 1997.

[19] R. I Chang, M. Chen, M. T. Ko, and J. M. Ho, “Designing the on–off CBR transmission schedule for jitter-free VBR media playback in real-time networks,” in *Proceedings of the IEEE International Conference on Real-Time and Embedded Computing Systems and Applications (RTCSA)*, 1997.

[20] R. Tewari, H. M. Vin, A. Dan, and D. Sitaram, “Resource-based caching for web servers,” in *Proceedings of SPIE/ACM Conference on Multimedia Computing and Networking*, January 1998.

[21] R. I Chang, M. C. Chen, J. M. Ho, and M. T. Ko, “Characterizing the minimal required resources for admission control of pre-recorded VBR video transmission by an $O(n \log n)$ algorithm,” in *Proceedings of International Conference on Computer Communications and Networks (ICCCN)*, 1998.

[22] ——, “An effective and efficient traffic-smoothing scheme for delivery of online VBR media streams,” in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 1999.

[23] S. S. Lam, S. Chow, and D. K. Y. Yau, “An algorithm for lossless smoothing of MPEG video,” in *Proceedings of ACM SIGCOMM*, 1994.

[24] S. Williams, M. Abrams, C. R. Standbridge, G. Abdulla, and E. A. Fox, “Removal policies in network caches for world wide web documents,” in *Proceedings of ACM SIGCOMM*, 1996.

[25] S. Sen, J. Rexford, and D. Towsley, “Proxy prefix caching for multimedia streams,” in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 1999.

[26] S. H. Chang, R. I Chang, J. M. Ho, and Y. J. Oyang, “OC: an optimal cache algorithm for video staging,” in *Proceedings of the IEEE International Conference on Networking (ICN)*, 2002.

[27] ——, “An effective approach to video staging in streaming applications,” in *Proceedings of IEEE Globe Communication Conference (GLOBECOM 2002)*.

[28] T. Ott, T. V. Lakshman, and A. Tabatabai, “A scheme for smoothing delay-sensitive traffic offered to ATM networks,” in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 1992.

[29] T. V. Lakshman, A. Ortega, and A. R. Reibman, “Variable bit-rate (VBR) video: Tradeoffs and potentials,” *Proceedings of the IEEE Multimedia*, May 1998.

[30] W. Feng and S. Sechrest, “Smoothing and buffering for delivery of prerecorded compressed video,” *Computer Communications*, October 1995.

[31] W. C. Feng and J. Rexford, “A comparison of bandwidth smoothing techniques for the transmission of prerecorded compressed video,” in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 1997.

[32] W. Feng, F. Jahanian, and S. Sechrest, “An optimal bandwidth allocation strategy for the delivery of compressed prerecorded video,” *Springer-Verlag Multimedia Systems Journal*, September 1997.

[33] W. H. Ma and D. H. C. Du, “Reducing bandwidth requirement for delivering video over wide area network with proxy server,” in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2000.

[34] Z. Miao and A. Ortega, “Proxy caching for efficient services over the internet,” in *Proceedings of the 9th International Packet Video Workshop*, 1999.

[35] Z. L. Zhang, Y. Wang, D. H. C. Du, and D. Su, “Video staging: a proxy server based approach to end-to-end video delivery over wide-area-networks,” *IEEE/ACM Transaction on Networking*, 2000.

[36] [Online]. Available: http://www-info3.informatik.uni-wuerzburg.de/MPEG/traces/

Shin-Hung Chang received the B.S. degree in Computer Science and Information Engineering Department from Fu Jen Catholic University, Taiwan, in 1996. Then, he joined Computer Systems and Communications Laboratory (CSCL) in Institute of Information Science, Academia Sinica, to develop multimedia systems. He received M.S. degree in Computer Science and Information Engineering Department from National Taiwan University, Taiwan, in 1998 and Ph.D. degree from Computer Science and Information Engineering Department, National Taiwan University, Taipei Taiwan in 2005. His research interests include audio/video codec, media streaming, media relay proxy, multimedia networking, and peer-to-peer (P2P) applications.

Ray-I Chang received his Ph.D. degree in Electrical Engineering and Computer Science from National Chiao Tung University in 1996, where he was a member of Operating Systems Laboratory. Then, he joined Computer Systems and Communications Laboratory (CSCL) in Institute of Information Science, Academia Sinica, to develop video-on-demand servers and digital library systems. In 2003, he joined Department of Engineering Science, National Taiwan University. His current research interests include multimedia networking and data mining. Dr. Chang is a member of IEEE.

Jan-Ming Ho received the B.S. degree in Electrical Engineering from National Cheng Kung University, Taiwan, in 1978 and the M.S. degree from Institute of Electronics at National Chiao Tung University, Taiwan, in 1980. He received the Ph.D. degree in electrical engineering and computer science from Northwestern University, USA, in 1989. He joined the Institute of Information Science, Academia Sinica, Taipei Taiwan, as a associate research fellow in 1989 and was promoted to research fellow in 1994. He visited IBM T.J. Watson Research Center in the summers of 1987 and 1988, Leonardo Fibonacci Institute for the Foundations of Computer Science, Italy, in the summer of 1992, and Dagstuhl-Seminar on "Combinatorial Methods for Integrated Circuit Design," IBFI-Geschaftsstelle, Schlo£ Dagstuhl, Fachbereich Informatik, Bau 36, Universitat des Saarlandes, Germany, in October 1993. He is a member of the IEEE and ACM. His research interests target at the integration of theoretical and application-oriented research, including mobile computing, environment for management and presentation of digital archive, management, retrieval, and classification of Web documents, continuous video streaming and distribution, video conferencing, real-time operating systems with applications to continuous media systems, computational geometry, combinatorial optimization, VLSI design algorithms, and implementation and testing of VLSI algorithms on real designs. He is associate editor of IEEE TRANSACTIONS ON MULTIMEDIA. He was program chair of the Symposium on Real-Time Media Systems, Taipei, 1994–1998, general co-chair of the International Symposium on Multi-Technology Information Processing, 1997, and general co-chair of IEEE RTAS 2001. He was also a steering committee member of the VLSI Design/CAD Symposium, and program committee member of several previous conferences including ICDCS 1999, and IEEE Workshop on Dependable and Real-Time E-Commerce Systems (DARE'98), etc. In domestic activities, he is program chair of the Digital Archive Task Force Conference, the First Workshop on Digital Archive Technology, a steering committee member of the 14th VLSI Design/CAD Symposium and the International Conference on Open Source 2002, and is also a program committee member of the 13th Workshop on Object-Oriented Technology and Applications, the Eighth Workshop on Mobile Computing, the 2001 Summer Institute on Bio-Informatics, Workshop on Information Society and Digital Divide, the 2002 International Conference on Digital Archive Technologies (ICDAT2002), the APEC Workshop on e-Learning and Digital Archives (APEC2002), and the 2003 Workshop on e-Commerce, e-Business, and e-Service (EEE'03).

Yen-Jen Oyang received the B.S. degree in Computer Science and Information Engineering from National Taiwan University, Taipei Taiwan, in 1982, the M.S. degree in Computer Science from the California Institute of Technology, USA, in 1984, and the Ph.D. degree in Electrical Engineering from Stanford University, USA, in 1988. He is currently a Professor in the Department of Computer Science and Information Engineering, National Taiwan University, Taipei Taiwan. His research interests include data mining/machine learning, video on demand, and disk storage scheduling.