AN EFFICIENT AND SCALABLE SMOOTHING ALGORITHM OF VBR STREAMS

Kaihui Li[1,2], Changqiao Xu[1,2], Jin Xu[1,2], Yuanhai Zhang[1,2]
[1] Institute of Software, The Chinese Academy of Sciences, Beijing, P.R. CHINA 100080
[2] Graduate University of the Chinese Academy of Sciences, Beijing, P.R. CHINA 100039
kaihui01@ios.cn, changqiao@ios.cn, xujin03@ios.cn, yuanhai02@ios.cn

Abstract—In order to obtain better video quality, media files are required to use variable-bit-rate (VBR) encoding. However, it produces traffic burst and unbalanced resource utilizations to translate VBR-encoded video. In this paper, we propose a novel bandwidth smoothing algorithm, Buffer Sharing and Bandwidth Smoothing of VBR Streams (BSBS-VBR), which combines with prefetching and interval caching, allows video server to transmit a VBR-encoded stream at a fixed rate and makes users to share a disk stream. BSBS-VBR can also allocate and adjust buffer size dynamically according to the current request distribution and available resources. It can reduce the peak requirements of disk bandwidth and network bandwidth, improve utility of the resources, and serve more users by using this algorithm. These conclusions are proved by comparing with several existing methods experimentally.

1. INTRODUCTION

Since video services have been an important part of our life, and some unique features of streaming media, such as huge size, long-lived session and demand of timely delivery, need to consume a tremendous amount of network bandwidth and server resources. For providing the service of video stream, media files must be compressed by using compression technique. The encoding technique for streaming can be either constant-bit-rate (CBR) or variable-bit-rate (VBR). CBR encoding enforces a constant bit rate and the video quality is variable. VBR encoding enforces a constant quality of video and the video bit rate is variable. Compared with the CBR counterpart serving videos of the same average bit rate, it can achieve better quality of video with VBR encoding. That is, VBR-encoded video can achieve visual quality similar to that of CBR-encoded video at lower bit rate [1]. So, high quality video content is usually stored and streamed in a compressed format with a VBR property.

When users are served with VBR-encoded streams, they can get constant quality compressed video. However, due to frequent traffic burst of bit rate, VBR streams will have high variability in their resource requirements which can lead to low utilization of disk and network bandwidth in the common case [2]. For improving the utility of the resources and system throughput, many researchers have proposed a lot of smoothing measures which can serve more requests by removing the peak resources requirements. These measures can be generalized three kinds: (1) Video Segmenting Measure, which divides a VBR encoding stream into *n* segments whose rates are constant or whose playback times are the same, then makes a programming to these segments [3]. For example, the segments are stored in the disk in term of different storage techniques so that they can be retrieved respectively in a fixed period [4]; (2) to utilize the remaining disk bandwidth and buffer space evenly and adequately, the period lengths of retrieving data blocks are changed dynamically according to the state of the remaining resources, so as to maximize the throughput [5]; (3) stream data is prefetched into the server, proxy or client buffer so that the leaving data can be obtained and transmitted with constant bit rate [6]. For example, it can smooth out disk bandwidth peaks by prefetching stream data into the server buffer. Since the segments divided by Video Segmenting Measure are different each other, it complicates the disk retrieval. The measure, which adjusts the period length dynamically according to available disk bandwidth and buffer space, can consume a great deal of resource during system run-time. The prefetching [7] needs enough available buffer space at the server, proxy or client. This method is effective and practical for smoothing VBR stream in the applications that provide continuous and real-time services with stored media files.

We propose a novel smoothing algorithm for Video-On-Demand (VOD) application that can support VBR-encoded stream. It combines with prefetching and interval caching [8], allocates and adjusts buffer size dynamically. It can reduce the peak requirements of disk and network bandwidth, improve utility of the resources.

2. SYSTEM ARCHITECTURE AND BSBS-VBR ALGORITHM

2.1. System Architecture

Fig. 1 shows a typical architecture of VOD. The server includes a number of disks and a memory. A simple treating of a request is following: when a request arrives, the server allocates some special resources to it. Then, the accepted client requests data periodically, the server reads the requested data from the disk to the memory and sends them to the client through the network, until either the end of the stream is reached, or the client requests halt of the playback.

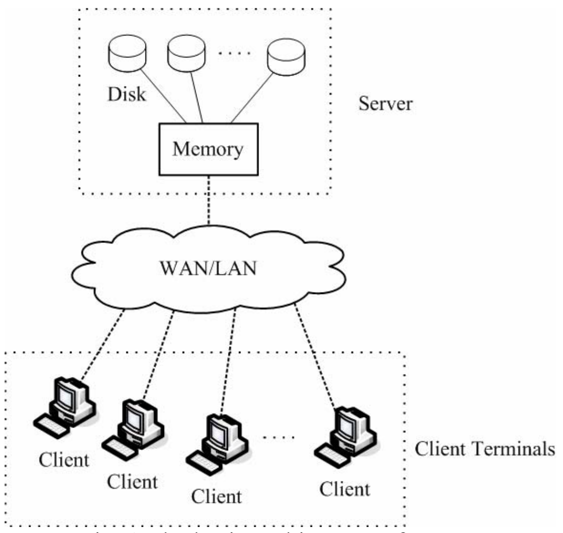

Fig. 1 The basic architecture of system

2.2. BSBS-VBR Algorithm

Buffer Sharing and Bandwidth Smoothing of VBR Streams (BSBS-VBR) algorithm transmits the suffix of the stream at a fixed rate by quick prefetching prefix. It can also make many users to share data in the buffer by interval caching. By these, this algorithm can reduce the requirement of disk bandwidth and allocate these resources fairly according to available disk bandwidth and cache space.

For the sake of obtaining the mathematical representation of the problem, we assume that the stream served by using disk stream will not consume the buffer resource. The stream served by using buffer will only demand relevant buffer space. The streams will be only served by using disk or buffer.

Table 1 gives the variables used in BSBS-VBR.

P_i and P_i' are the parameters to smooth VBR stream. In order to allow a server to transmit a VBR stream at a fixed rate, close to its mean encoding bit rate, we need to allocate the size P_i' of the buffer at the client and fetch the prefix P_i of the stream into P_i' before the user starts playback. The values of P_i and P_i' are calculated before system running by the following methods.

Assuming that the mean bit rate of a requested stream is b bits per second, there are total n frames in the requested video and the size of j_{th} frame is f_j bits. The mean frame size f_{mean} can be calculated by (1). A $Min(P_i)$ [9] or $Min(P_i')$ can be found by (2) or (3).

$$f_{mean} = \frac{1}{n}\left(\sum_{j=1}^{n} f_j\right) = \frac{b}{fram\_rate} \tag{1}$$

$$\forall j\left[\left(j \in (1,\dots,n)\right) \wedge \left(\left(\left(\sum_{k=1}^{j} f_k\right) - j \times f_{mean}\right) \le P_\mathrm{i}\right)\right] \tag{2}$$

Table 1 The variables used in BSBS-VBR

Variable	Description
M	Size of the whole memory
B	Size of the whole disk bandwidth
M'	Size of the currently available memory
B'	Size of the currently available disk bandwidth
M_i	Size of the cache used by the i_{th} request
B_i	Size of the disk bandwidth used by the i_{th} request
B_i'	Average bit rate of the requested stream
B_{Mi}'	Size of the disk bandwidth allocated to transmit P_i, $B_{Mi}'=P_i/t_i$
t_i	Start-up delay limit at the client
P_i	Size of the data prefetched by i_{th} request
P_i'	Size of the buffer allocated for the i_{th} request at the client, $P_i' \ge P_i$
N	Number of user requests in service at some time
d_m	Limit of interval cache size between two adjacent displays referencing the same video
T	Size of whole interval cache of a certain video

$$\forall j\Big[\left(j \in (1,\dots,n)\right) \wedge \left(\left(P_\mathrm{i} + j \times f_{mean}\right) \le n \times f_{mean}\right) \wedge \left(\left(P_\mathrm{i} + j \times f_{mean} - \left(\sum_{k=1}^{j} f_k\right)\right) \le P_\mathrm{i}'\right)\Big] \tag{3}$$

If we prefetch P_i bits video data into the client buffer, server's mean-bit-rate suffix transmission can guarantee that the video data will arrive before it is required to playback. And the user will not experience any unexpected pause in playback.

This algorithm needs to allocate P_i' bits buffer at the client for the requested stream so that it is sufficient to guarantee that overflow will not occur.

After media server accepts a request, each display served by disk stream or buffer needs to consume some resources. According to table 1, the resources used by N users can not exceed the total resources of the server [10]. These are shown in (4) and (5).

$$\sum_{i=1}^{N} M_i \le M \tag{4}$$

$$\sum_{i=1}^{N} B_i \le B \tag{5}$$

Here, d_m is a distance threshold [11]. It limits the size of the needed sharing buffer between two adjacent displays referencing the same stream. If the needed buffer size between two adjacent displays exceeds d_m, they are allowed to neither cache their intermediate data nor share one disk stream by using memory. The requirements of the cache and disk bandwidth are affected by the value of d_m. In the practical system, choosing optimal d_m can make the best of the existing resources. The value of T denotes the size of the interval cache finally formed by these requests sharing one disk stream. The value of d_m is calculated by (6) in this algorithm:

$$d_m = \begin{cases} \left\lceil \frac{B_i^{'}}{B^{'}} M^{'} \right\rceil (B^{'} \neq 0 and B_i^{'} \leq B^{'}) \\ Min(M^{'}, \left\lceil 2\frac{M - M^{'}}{n} \right\rceil)(B^{'} = 0 or B_i^{'} > B^{'}) \end{cases} \quad (1 \leq i \leq N) \tag{6}$$

The computation of the buffer size to be allocated considers both available resources and disk bandwidth required by the requested stream in (6), so that it can keep balance of the cache and disk bandwidth consumptions.

Request management process of BSBS-VBR is described in the following.

1) If the video A is requested by the i_{th} request and the requested data is in the buffer of the server, this request will be served directly from the buffer. The server sends the prefix P_i bits at a $Max(B_{Mi}', B_i')$ rate and transmits the suffix at a B_i' rate after finishing P_i transmission. The client can start playback after accumulating P_i bits of video data.
2) If the video A is requested by the i_{th} request and the requested data is not in the buffer of the server, the bandwidth $Max(B_{Mi}', B_i')$ and the buffer d_m calculated by (6) ($T=d_m$) are allocated for this request. After finishing P_i transmission, if $B_{Mi}' > B_i'$ and the suffix can be found in the buffer of the server, this request will be served from the buffer; the disk bandwidth $Max(B_{Mi}', B_i')$ and the buffer d_m, which is not used by new request, are reclaimed. Otherwise, reclaim the disk bandwidth $(B_{Mi}' - B_i')$ and do not change the buffer size.
3) If new request is served from the buffer T at the start or after finishing P_i transmission, the size d_m' of a new buffer is calculated by (6) and d_m' is compared with $(d_m - t*B_i')$, which t is the interval time between the new request and the former request. $Max(d_m', (d_m - t*B_i'))$ is the size of the spare buffer that will be used to form interval cache, namely, $T=T+Max(d_m', (d_m - t*B_i'))$. Above procedure repeats until no new request arrives before the spare buffer is full.
4) If no new request for the video A arrives in the period of the buffer remained by T, superfluous buffer space is reclaimed at the end of T and only part of the buffer is left to provide service for these requests arriving in this period.
5) When new request arrives, if there are not enough available resources and enough resources can not be replaced (i.e., (4) or (5) is not satisfied), this request is rejected.
6) If the service terminates, the resources of buffer and disk bandwidth are reclaimed.

2.3. Resources Allocation and Reclamation Algorithms of BSBS-VBR

Buffer allocation: when a new request arrives, d_m is calculated by (6). The buffer d_m is allocated to the request served by disk stream. If the request can be served from the buffer, we compare the value of d_m with the size of the buffer, which is remaining after we have allocated and some of them have formed interval, then set the bigger as the size of new available buffer. If available cache is not enough, the buffer reclamation algorithm is called.

Buffer reclamation:

1) when the end of a stream is reached, the idle buffer is reclaimed.
2) after P_i is transmitted by the disk stream and the suffix will be obtained from the buffer of the server, the allocated buffer, which is not used by new request, is reclaimed.
3) if no new request for the video A arrives in the period of the cache remained by T, superfluous buffer space is reclaimed at the end of T and only part of the buffer is left to provide service for these requests arriving in this period.
4) if available cache is not enough when new request arrives, buffer reclamation operations as follows: (a) if there is the buffer that has been allocated and interval cache is not formed in it yet, we directly reclaim it. (b) otherwise, if interval cache has been formed in M_i, there is no adjacent request after M_i and the rate of buffer M_i to bandwidth B_i' is the largest, then the buffer M_i is reclaimed, and these requests served by M_i will be served by disk bandwidth.

Disk bandwidth allocation: when available disk bandwidth is enough, the required bandwidth will be allocated to the request. If available disk bandwidth is not enough, the disk bandwidth reclamation algorithm is called.

Disk bandwidth reclamation:

1) when the end of a stream is reached, the corresponding disk bandwidth is reclaimed.
2) after P_i is transmitted by the disk stream, the spare bandwidth is reclaimed.
3) when available disk resource is not enough, the disk bandwidth, which is serving the request j which has the least interval time from the former request and is served by disk stream, will be reclaimed and request j will be served from the buffer.

3. EVALUATION OF BSBS-VBR

These experiments are based on the network architecture in Fig. 1. For simplicity, we assume that all the videos are of equal length, have equal mean bit rate, there is enough network bandwidth and network delay between server and client can be ignored. The server contains 100 VBR-encoded videos. We use the values of the video trace that is a high quality MPEG-4 encoded movie Robin Hood from [12]. The length of this video is 60 minutes. The frame rate is 25 frames per second. The mean bit rate of the video is 0.87 Mbps, namely B_i'=0.87Mbps($1 \leq i \leq N$). The peak bit rate is 3.2 Mbps. From expression (2), the size P_i($1 \leq i \leq N$) of the prefetched data is 3.9MB and from (3),

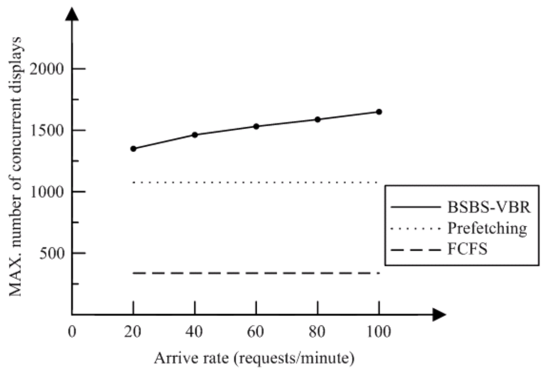

Fig. 2 Comparison of simultaneous clients for different *r*

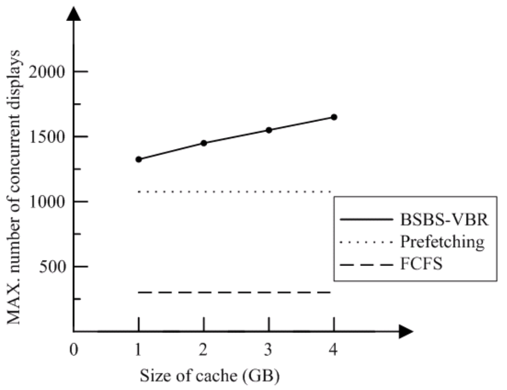

Fig. 3 Comparison of simultaneous clients for different *M*

the size P_i' ($1 \le i \le N$) of the allocated buffer at the client is 15.7MB. Assume that the limit t_i ($1 \le i \le N$) of start-up delay is 3 seconds at the client. The request arrival follows an Average process with rate *r* requests per minute. The frequency of access to each object is based on a Zipf distribution with parameter 0.271, and the videos in the disk are organized according to the access frequency of each video. We configure experimental server with *B*=1Gbps and *M*=1/2/3/4GB. We compare the performance of BSBS-VBR, Prefetching (P_i is prefetched into the client buffer at an average rate B_i') and FCFS by experimental results.

First, *M* is fixed 2GB. Experiments are executed by changing the value (20, 40, 60, 80, 100) of arrival rates *r*. Fig. 2 shows simultaneous clients sustained by each scheme. The result shows that the BSBS-VBR algorithm can serve the most displays and achieves better performance.

Secondly, *r* is fixed 50 requests per minute. Experiments are executed by changing the value (1, 2, 3, 4) of *M*. Fig. 3 shows simultaneous clients sustained by each scheme for different *M*. The result shows that the BSBS-VBR can serve more displays and the number of displays is increasing with the memory increase. So, adaptive ability of the BSBS-VBR is relatively better for different rate of the memory and disk bandwidth resources.

4. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

In this paper, we present an effective smoothing algorithm, which guarantees that neither starvation nor overflow occurs to transmit a VBR-encoded video at an average rate by prefetching some data and allocating the buffer for these data at the client. This algorithm can reduce the burst requirement of VBR-stream, simplify resources allocation methods of server and network and satisfy the start-up delay limit at the client. This algorithm also allocates resources for each request according to the available cache and disk bandwidth, and adjusts buffer size at run-time according to the current request distribution and available resources dynamically. It can reduce more consumption of disk bandwidth and improve the efficiency of the resources by using interval caching.

In our next research, we will add support of VCR functions to BSBS-VBR algorithm, and extend BSBS-VBR to FGS (Fine granularity scalability) media.

5. REFERENCES

[1] Chun Wai Kong, Jack Y. B. Lee. "Slice-and-Patch -- An Algorithm to Support VBR Video Streaming in a Multicast-based Video-on-Demand System", J. Inf. Sci. Eng. 19(3): 517-530(2003).

[2] Anastasiadis S V, Sevcik K C, Stumm M. "Server-based smoothing of variable bit-rate streams", In: Proc. ACM Multimedia, Oct.2001.

[3] Shin-Hung Chang, Ray-I Chang, Jan-Ming Ho, and Yen-Jen Oyang. "An Optimal Cache Algorithm for Streaming VBR Video over a Heterogeneous Network", Computer Communications 28(16): 1852-1861 (2005).

[4] D. Makaroff, J. Coutu, and F. Liu. "Disk Performance and VBR Admission Control for Media Servers", DMS '2005, Banff, Canada, pp. 224-227, Sep. 2005.

[5] Lee K, Yeom H Y. "An effective admission control mechanism for variable-bit-rate video streams", Multimedia Systems, 1999, 7(4): 305～311.

[6] S.V. Anastasiadis, K.C. Sevcik, and M. Stumm. "Shared-Buffer Smoothing of Variable Bit-Rate Streams", Performance Evaluation, vol. 59,no. 1, pp. 47-72, Jan. 2005.

[7] B. Sonah, Mabo Robert. "Ito: Considering video characteristics for improved cache performance in VOD systems", SAC 2001: 429-433.

[8] Nabil J. Sarhan and Chita R. Das. "Analysis of Caching Performance in Multimedia Servers", In the Proceedings of the 8th International Conference on Internet and Multimedia Systems and Applications, pp. 288-293, Aug. 2004.

[9] M. H. Kabir, Eric G. Manning, and Gholamali C. Shoja. "Scalable Multimedia Streaming Model and Transmission Scheme for VBR-Encoded Videos", in Proc. of 16th IASTED Int. Conf. on Parallel and Distributed Computing and Systems (PDCS), Cambridge, USA, pp.867-872, Nov. 2004.

[10] J. Fernández, J. Carretero, F. Garcia, J.. Pérez, A. Calderón. "Enhancing Multimedia Caching Algorithm Performance Through New Interval Definition Strategies", 36th Annual Simulation Symposium 2003.

[11] W. Shi, S. Ghandeharizadeh. "Controlled Buffer Sharing in Continuous Media Servers", Multimedia Tools Appl. 23(2): 131-159 (2004).

[12] Video trace of MPEG-4 encoded Robin Hood, http://www-tkn.ee.tu-berlin.de/research/trace/ltvt.html.