Available online at www.sciencedirect.com
SCIENCE DIRECT®
J. Parallel Distrib. Comput. 66 (2006) 334–344
Journal of Parallel and Distributed Computing
www.elsevier.com/locate/jpdc

An integrated admission control scheme for the delivery of streaming media

Zhonghang Xia[a,*], I-Ling Yen[b], Donglei Du[c], Peng Li[b]

[a]Department of Computer Science, Western Kentucky University, Bowling Green, KY 42101, USA
[b]Department of Computer Science, University of Texas at Dallas, Richardson, TX, 75083, USA
[c]School of Administration, University of New Brunswick, Fredericton, NB, Canada

Received 27 October 2003; received in revised form 6 June 2005; accepted 6 December 2005

Abstract

Quality of service (QoS) assurance is a major concern in media-on-demand (MoD) systems. Admission control is one of the most important issues that need to be addressed for QoS assurance. Also, smoothing is a basic technique for the media server to improve its bandwidth and buffer utilization. However, existing approaches cannot achieve the best resource utilization because (1) they cannot fully utilize the time-varying buffer spaces available at both server and client sides due to the separation of admission control and smoothing processes, and (2) the computing time is unacceptable for media of long duration. In this paper, we formulate the admission control as an integer programming problem and propose several heuristic methods for solving the problem. Specially, we introduce an efficient scheme, called batched admission (BA) scheme, which integrates admission control, transmission rate smoothing, and batching, to achieve best resource utilization with guaranteed QoS. Experimental studies show that the BA scheme outperforms existing approaches.

Keywords: Streaming media; Admission control; Transmission schedules; Smoothing; Multi-commodity network flow

1. Introduction

Rapid advances in networking have made many media-on-demand (MoD) applications, such as distance learning, remote operation and training, teleconferencing, video-on-demand, etc., feasible. In a typical MoD system, the multimedia server needs to support media streaming service for a large number of clients concurrently. Various techniques have been developed to improve the efficiency of media delivery.

Admission control is a key process for guaranteeing continuous real-time playback. It restricts the number of clients accessing critical resources, such as disk bandwidth, server and client buffer space, etc., to ensure QoS for existing client requests. Many admission control algorithms based on disk bandwidth limitations have been proposed in the literature [4,21,22]. In general, admission control algorithms can be classified into deterministic, statistical, and observation-based approaches. Statistical and observation-based algorithms use aggressive admission control criteria and can yield high utilization of server resources; however, they may not guarantee QoS due to the potential of over-commitment. On the other hand, deterministic algorithms can provide strict performance guarantees and are more preferable in MoD systems.

To support guaranteed QoS services, deterministic admission control algorithms reserve system resources at peak retrieval rate for the new requests. Hence, this method, when applied to variable bit-rate (VBR) media, under-utilizes the critical resources due to the significant rate variability in VBR streams. Reducing the rate variability of VBR media is a basic technique to improve the utilization of system resources. Two basic approaches along this direction are temporal smoothing and aggregation. Temporal smoothing is a work-ahead approach on a per-stream basis [8,9,17,18,23]. In this approach, a buffer is introduced somewhere on the path so that media frames can be sent ahead of their playback time. The aggregation approach reduces the rate variability by applying spatial averaging schemes [12,11]. Multiple streams are transmitted over the same virtual channel to average out the traffic bursts. Statistical multiplexing (SM) is a typical aggregation mechanism which allows the sum of the peak rates of multiple streams to exceed the allocated

* Corresponding author. Fax: +1 270 745 6449.
E-mail addresses: zhonghang.xia@wku.edu (Z. Xia), ilyen@utdallas.edu (I.-L. Yen), ddu@unb.ca (D. Du), pengli@utdallas.edu (P. Li).

0743-7315/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jpdc.2005.12.002

bandwidth. Generally, SM algorithms are used in the stochastic traffic models. Thus, it only provides statistical guarantees and can result in buffer overflow at the server and/or the client sides.

Batching is another technique to improve resource utilization. In batching, requests are delayed for a short time to allow multiple requests that access the same media being served in one multicast channel. The performance of several video scheduling policies using batching scheme has been studied in [2,6]. The experimental results show that the First-Come-First-Served (FCFS) policy has a better performance than the maximum queue length (MQL) policy (which chooses the media with the maximum number of outstanding requests to be delivered first) [6]. In [2], a better MQL scheduling policy is proposed. It schedules the media with the maximum factored queue length first. The policy results in reduced latency and increased fairness compared with the other policies. In general, it has been shown that batching is a bandwidth efficient technique for large scale MoD systems [19].

In this paper, we develop heuristic methods to improve the utilization of critical system resources by combining admission control, smoothing, and batching techniques. Existing research works consider these techniques separately and, hence, do not gain the full advantages for media delivery. In our approach, all requests arriving within a time interval are batched and processed at the beginning of the next interval. Due to batching, the server can make admission decisions for multiple requests simultaneously. This facilitates the derivation of a transmission schedule that aggregates multiple streams and smoothes across these streams. Compared to existing techniques that consider smoothing individual streams, this approach further reduces peak bit rates. To further enhance the utilization of system resources, we consider processing short media and long media in different manners. For a short media, the server can make an admission decision and compute a transmission schedule immediately. A new request for a short media can only be admitted when the server can determine a feasible transmission schedule for the client. For a long media, however, the server first makes an admission decision and then computes its transmission schedule over one period at a time if the request is admitted. Our goal is to maximize the admission rate. We formulate this admission control problem as a mixed integer programming problem and introduce an efficient integrated scheme by transforming the original problem into a multi-commodities network flow problem.

The rest of the paper is organized as follows. Related work is reviewed in Section 2. Section 3 presents the problem setting, system model, and the integer programming problem formulation. Section 4 studies two heuristic methods. The integrated scheme is introduced in Section 5. Numerical results are presented in Section 6. Section 7 states the conclusion of the paper.

2. Related work

A VBR video can be stored on the disk in three models: CTL, CDL, and hybrid. CTL model is characterized as having variable data length with constant playback duration. CDL model is characterized as having constant amounts of data for each time slot. In the hybrid model, media data are stored in fix-sized blocks as in CDL, but multiple blocks can form a logical CTL block. Chang and Zakhor [5] presented several deterministic admission control algorithms for these data placement models using a priori knowledge of the bit rate traces of the requested video. Also, the cost analysis is conducted for the three techniques and the results show that the hybrid scheme has the advantages of high efficiency and low fragmentation. Biersack et al. [4] estimate an upper bound on the data amount of each stream over a time interval. They assume that the server retrieves data for multiple streams in a round-robin schedule. The number of streams admitted is limited by the total transmission time, disk rotation latency and seek time which should not exceed the service round time. In [13], authors present exact schedulability conditions for admission control functions in packet switching networks. Based on the conditions, the maximum number of admissible streams for a given transmission schedule and delay bound can be determined.

Several deterministic admission control algorithms for VBR media streams were evaluated in [15]. The disk retrieval capacity is measured by the read bandwidth which is defined as the maximum number of blocks the media server can guarantee to retrieve from the disk during each service round, namely, *MinRead*. The performance tests show that some algorithms are too conservative while others are too aggressive. For example, the simple maximum algorithm has a fast execution time, but accepts too few streams. The instantaneous maximum (IM) algorithm sums all the currently admitted block schedules and keeps the sum of the required media blocks during each time slot in a table. To make an admission decision for a new request, the server adds the block schedule for the requested stream to the table. If the number of required media blocks during any of the time slot is greater than *MinRead*, the new request is rejected, otherwise it is accepted. The average algorithm acts more aggressively by reading disk blocks ahead of the schedule. But it does not provide guaranteed QoS since media blocks may be lost if there is insufficient buffer space. The VBR Simulation (vbrSim) algorithm takes the buffer size into consideration and performs better than the other deterministic algorithms. We will compare the BA scheme with a modified vbrSim scheme in Section 6.

Along another direction, various smoothing techniques have been developed to reduce burstiness of VBR video. Several papers have studied off-line work-ahead smoothing of stored video. In [7], authors present a bandwidth allocation algorithm to minimize the number of bandwidth increases, the smallest peak bandwidth requirements, and the largest minimum bandwidth requirements. Although the algorithm minimizes peak transmission rate, it does not minimize rate variability. In [16], McManus and Ross establish a constant-bit-rate (CBR) channel between the video server and the receiver. A large amount of video data is sent to the receiver's buffer ahead of their playback time. Since the transmission rate is maintained to be constant, the buffer size requirement at the receiver is a big concern. Salehi et al. proposed a smoothing schedule for transmitting a single stream. Instead of using constant transmitting rate throughout, their transmission schedule is piecewise-constant

and they prove that this schedule is an optimal solution under the majorization smoothing definition. Wang et al. [23] reduce the burstiness of a VBR video stream by introducing buffers at proxy servers. A video stream is divided into two parts by a predefined cut-off rate. The lower part consists of a sequence of partial frames less than the cut-off rate and the upper part consists of a sequence of partial frames greater than the cut-off rate. The upper part is duplicated and always stored at a proxy server. The smaller the cut-off rate is, the more stream data can be stored at the proxy server.

Krunz and Tripathi [11] propose a SM approach that can provide a deterministic guarantee on traffic rate. They use the time-varying traffic envelope to estimate a bound on the bit rate. By exploiting the pre-defined compression video frame pattern, the starting times of I, P, and B frames are strategically arranged so that the peak bit rates of various video streams are evened. Liew and Chan [14] proposed a lossless aggregation method to transmit multiple video streams over a common CBR channel. Within the fixed total transmitting rate of multiple streams, each individual stream rate can be adjusted dynamically depending on its relative traffic characteristic. This approach provides a deterministic guarantee on transmission bandwidth, but the buffer requirement might be large due to the CBR channel. Recently, a mechanism combining temporal averaging and SM was presented by Kweon and Shin [12] for aggregate VoD streams transmission. Buffers are introduced at the server and the client sites to smooth the transmission rate.

More recently, an optimal multiplex method [24] has been proposed to achieve high bandwidth utilization while maintaining deterministic QoS guarantees. Given a group of streams, a family of optimal multiplexing schedules can be computed such that multiple streams can be delivered under the optimal multiplexing bandwidth B. The multiplexing schedules are then mapped into individual per-stream schedules. The running time of computing the optimal bound B is $O(T^2)$, where T denotes the maximal length of multiple streams. Hence, this algorithm for long media is time-consuming.

Batching is another important approach to enhance the performance of MoD systems. The idea of batching was first proposed in [3] to support end-to-end performance guarantee for continuous media. In general, there is a trade-off between increasing the batching requests and increasing the average client waiting time. Dan et al. [6] explore various scheduling policies for selecting the movie to be multicast. In [20], proxy server partitions time into equal length intervals and the time slots are labeled by $0, 1, \ldots, t, \ldots$ Request arriving at time $t' \in [t-1, t)$ are batched and processed by the proxy server at the beginning of the time slot t.

There have been a lot of works in the areas of admission control, batching, and transmission rate smoothing. However, current works separate these issues. Existing approaches (addressing the issues independently) cannot best utilize system resources. We combine admission control, batching, and smoothing techniques and propose an integrated approach to significantly improve system resource utilization while providing deterministic QoS assurance.

3. Problem description

3.1. Problem setting

Consider an MoD system which consists of a media server with a storage disk, a high-speed network, and a group of client receivers (Fig. 1). We assume a set of VBR media are stored on the disk with fix-sized blocks and each block is of size b bytes (e.g. 64 kbyte), but different media may require different numbers of blocks to be retrieved each round based on their playback rates. However, the total guaranteed number of data blocks that can be read from the disk during a service round is limited, denoted by C, which can be measured by the method discussed in [15].

We also assume that the server and clients are equipped with fixed size buffers and the sequence of frame sizes for each video stream is known a priori to the server. Upon receiving requests from clients, the server reads data blocks from the disks into a buffer in rounds for each client according to the transmission schedules. Data blocks are stored in the server buffer in one service round and emptied into the network in the following round. Given current disk read capacity and buffer space, the server is expected to provide guaranteed service for as many clients as possible. Without loss of generality, the network is assumed to be lossless and zero delay for off-line analysis. For non-zero delay networks, results can be adjusted with an upper bound of jitter.

Here we give the list of notations used in the paper. Some of these will be explained in more detail when they are used:

- C disk read capacity (measured with the number of locks)
- B server buffer size
- $\overline{C}_j$ disk read capacity available at jth service round
- R_{it} transmission bit rate for ith stream at time slot t
- T time period over which transmission schedules are determined
- b bit size of each block
- $\overline{B}^s_j$ server buffer size available at the jth service round
- d_{it} the amount of data consumed by the ith client at time t (the tth frame size)
- $\overline{B}_i$ buffer size at the ith client

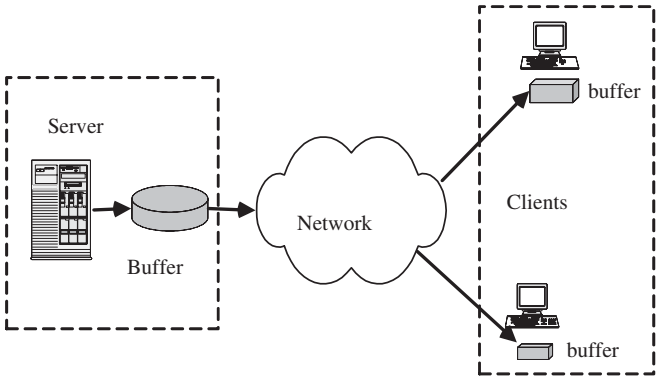


Fig. 1. Media-on-demand architecture.

B_{it} buffer fullness level at the ith client during time t
τ_i the deadline time of sending the first frame for the ith client
L_i the length of the ith video

3.2. Feasible transmission schedules

Major system resources, such as disk bandwidth and buffer space, need to be pre-scheduled to assure continuous playback. The system model is developed based on the discrete time unit, which is the duration to display an uncoded frame. The time is labeled by $t \in \{1, 2, \ldots\}$. A service round is chosen as a multiple of the frame display durations and denoted by r (i.e., r is the number of frame durations in one service round). Also, we assume that the size of a batching interval is r. The server processes the requests at the beginning of time slot tr, where $t = 1, 2, \ldots$.

Assume that there are N client requests from clients $A_1, A_2, \ldots, A_N$, requesting different media $F_1, F_2, \ldots, F_N$, respectively. The length of video stream F_i is L_i, which is measured in discrete time units. The server decides which request can be admitted and computes a transmission schedule for the request if admitted. Let R_{it}, $i = 1, \ldots, N$, be the transmission bit rate for the ith media stream at time unit t. We define a transmission schedule for stream F_i over a given time period T to be a set of bit rates $\{R_{i1}, R_{i2}, \ldots, R_{iT}\}$. To support continuous playback, a transmission schedule should avoid buffer starvation and overflow. Buffer starvation occurs when the buffer is completely empty. In this case, jitter will occur. On the other hand, buffer overflow occurs when data are sent to a full buffer. In this case, some information will be lost. Generally, we consider that the server is serving some requests when N new requests arrive. The available disk read capacity, $\overline{C}_j$, and the available server buffer space, $\overline{B}^s_j$, vary from round to round. Therefore, data blocks retrieved from the disk during the jth round should not exceed $\overline{C}_j$ and $\overline{B}^s_j$. In other words, if all N requests are admitted, then the total number of blocks retrieved during the jth service round should be less than the disk read bandwidth capacity, that is, $\frac{1}{b}\sum_{i=1}^{N}\sum_{t=(j-1)r+1}^{jr} R_{it} \leqslant \overline{C}_j$ Also, they should satisfy the server buffer constraint, that is, $\sum_{i=1}^{N}\sum_{t=(j-1)r+1}^{jr} R_{it} \leqslant \overline{B}^s_j$. Let $C^{\min}_j = \min\{b\overline{C}_j, \overline{B}^s_j\}$, $j = 1, \ldots, m$. We have,

$$\sum_{i=1}^{N}\sum_{t=(j-1)r+1}^{jr} R_{it} \leqslant C^{\min}_j, \quad j = 1, \ldots, m. \tag{1}$$

When the stream frames are transmitted to the clients, they are saved in the client buffers first and, then, decoded and displayed. Thus, the data transmitted to the ith client should satisfy its client buffer constraint $\overline{B}_i$. Let d_{it} be the amount of data consumed by the ith client at time t and B_{it} the difference between cumulative amounts of data sent and consumed by the ith client until time t. That is,

$$B_{it} = \sum_{k=1}^{t} R_{ik} - \sum_{k=1}^{t} d_{ik}. \tag{2}$$

We call B_{it} the ith client buffer fullness level at the end of time t. To prevent buffer underflow or overflow, the stream data cannot be transmitted ahead arbitrarily. The buffer level in every period should satisfy

$$0 \leqslant B_{it} \leqslant \overline{B}_i, \quad t = 1, \ldots, T. \tag{3}$$

Substituting B_{it}, we obtain the condition for a feasible transmission schedule:

$$\sum_{k=1}^{t} d_{ik} \leqslant \sum_{k=1}^{t} R_{ik} \leqslant \sum_{k=1}^{t} d_{ik} + \overline{B}_i. \tag{4}$$

Combining (2) and (4), we have

$$B_{i,t-1} + R_{it} - d_{it} = B_{it}. \tag{5}$$

We consider a transmission schedule $\{R_{i1}, R_{i2}, \ldots, R_{iT}\}$ over period T to be feasible if and only if it satisfies (1), (3), and (5).

3.3. Mathematical programming formulation

To address the admission control problem formally, we formulate it as an optimization problem of maximizing the number of feasible transmission schedules over a given time period for N requests. If there is no feasible transmission schedule for a request, then the request is rejected.

Let τ_i denote stream F_i's deadline when the first frame must be transmitted. Then we have $d_{it} = 0$, for all $\tau < \tau_i$, but $d_{i,\tau_i} > 0$. In order to achieve continuous playback for F_i, the client must receive all frames until $\tau_i + L_i$, and thus $d_{i\tau} = 0$, for all $\tau > \tau_i + L_i$. Without loss of generality, we consider transmission schedules over a period of time $T_{\max} = \max_i\{\tau_i + L_i\}$. Let $\delta_i = 1$ if the ith request is admitted, and $\delta_i = 0$ otherwise. That is, $\delta_i = \min\{\sum_{j=1}^{m}\sum_{t=(j-1)r+1}^{jr} R_{it}, 1\}$, $i = 1, \ldots, N$. Since short media consume less bandwidth in the long run and our goal is to maximize the number of admitted requests, we give the admission priority to short media. By assigning a value of gain to a successful admission, the optimization problem can be formulated as follows:

$$\max \quad \sum_{i=1}^{N} \frac{\delta_i}{L_i} \tag{6}$$

$$\text{s.t.} \quad B_{i-1,t} + R_{it} - d_{it} = B_{it}, \quad i = 1, \ldots, N, \quad t = 1, \ldots, T_{\max}, \tag{7}$$

$$0 \leqslant B_{it} \leqslant \overline{B}_i, \quad i=1, \ldots, N, \quad t=1, \ldots, T_{\max}, \tag{8}$$

$$B_{i,0} = 0, \quad i = 1, \ldots, N, \tag{9}$$

$$B_{i,t}=0, \quad i=1, \ldots, N, \quad \tau_i+L_i \leqslant t \leqslant T_{\max}, \tag{10}$$

$$\sum_{i=1}^{N} \delta_i \sum_{t=(j-1)r+1}^{jr} R_{it} \leqslant C^{\min}_j, \quad j=1, \ldots, m, \tag{11}$$

$$\delta_i = \min\left\{\sum_{j=1}^{m}\sum_{t=(j-1)r+1}^{jr} R_{it}, 1\right\}, \ i=1, \ldots, N, \tag{12}$$

$$R_{it} \text{ integers.} \tag{13}$$

According to the solution to this problem, we know that the ith request is admitted if $\delta_i = 1$ and its transmission schedule is R_{it}. Note that the 0–1 knapsack problem, which is NP-hard, is a special case of problem (6)–(13). Thus, the above problem is also NP-hard. In the following, we propose heuristic methods to solve it.

4. Two heuristic methods

4.1. A per-stream based method

The simplest heuristic to solve this problem can be executed in two steps: smoothing and, then, making an admission decision. The smoothing step is per-stream based (PSB). Starting from the shortest stream, we smooth the media stream by using the work-ahead scheme to minimize its peak bit rate. If the transmission schedule is acceptable according to current resources, then the request is admitted. Otherwise, we select the next shortest stream, and repeat the process. A work-ahead example is shown in Fig. 2. Fig. 2(a) illustrates a transmission schedule given by $\{5, 7, 4, 10, 6, 7\,\text{kB}\}$. The maximum bandwidth required for schedule A given in (a) is 10 kB. If the current disk read capacity and buffer space available are 8 kB, then the server will reject the request with transmission schedule A. However, we can pre-send 3 kB of frame 4 in time slot 3 (as shown in Fig. 2(b)). Then, the maximum bandwidth required is 7kB and the server can admit the request with schedule B.

A feasible transmission schedule for the ith stream can be obtained by solving the following optimization problem:

$$\min \max_{1 \leqslant t \leqslant T} \{R_{it} | R_{it} \text{ satisfies constraints (1), (3), (5)}\}. \quad (14)$$

In problem (14), for simplicity, we relax the integral constraints on R_{it}. Since problem (14) can be transformed into a standard linear programming problem by introducing an additional variable, we can use the CPLEX solver to solve this optimization problem and the numerical experiment is given in Section 6. Since the length of some videos may be very large, e.g., more than 2 h, computing a complete transmission schedule over such a period using the simple approach can be time-consuming.

4.2. Aggregation of multiple streams

When multiple streams are transmitted simultaneously, the total rate variability may further be reduced by aggregating multiple streams. An aggregation example is shown in Fig. 3,

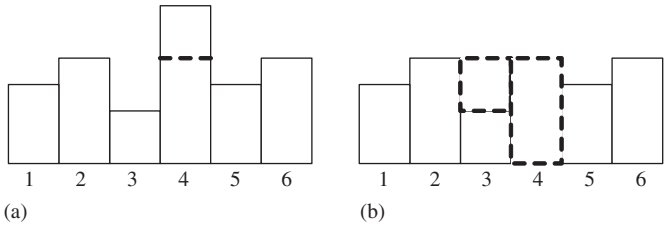


Fig. 2. (a) Schedule A, without work-ahead. (b) Schedule B, with work-ahead.

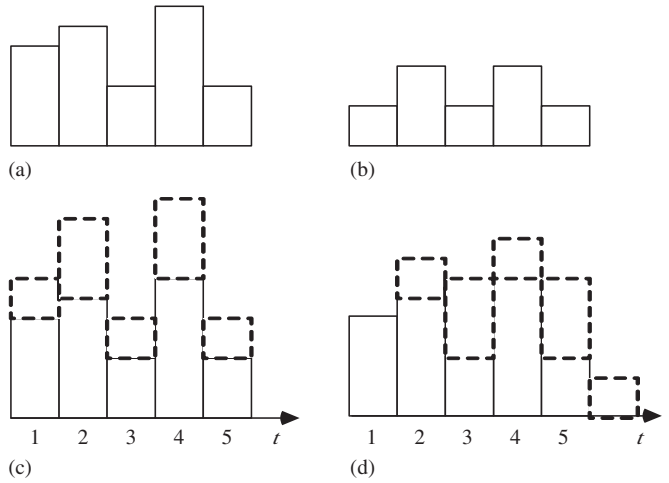


Fig. 3. (a) Frame sequence of media a. (b) Frame sequence of media b. (c) Schedule A, sending two media frames at the same time. (d) Schedule B, sending the second stream with delay.

where (a) and (b) show two media frame sequences with 5 frames. The frame sizes of media a are 5, 6, 3, 7 and 4 kB. The frame sizes of media b are 2, 4, 2, 4 and 2. Consider schedule A shown in Fig. 3(c). It sends the first frames of the two media simultaneously in time slot 1, second frames in time slot 2 and so on. Then, the maximum bandwidth required in this schedule is 11 kB. However, if media b is sent one time slot later than media a, then, the maximum bandwidth required in this schedule is only 9 kB. Hence, schedule B shown in Fig. 3(d) is more bandwidth-saving than schedule A.

The problem of smoothing the aggregate rate for multiple streams can be written as an optimization problem as follows:

$$\begin{aligned} \min \max_t \quad & \sum_{i=1}^{N} R_{it} \\ \text{s.t.} \quad & \sum_{i=1}^{N} \sum_{t=(j-1)r+1}^{jr} R_{it} \leqslant C_j^{\min}, \quad j = 1, \ldots, m, \\ & \text{constraints (7)–(10).} \end{aligned} \quad (15)$$

Actually the aggregate bit rate of transmission schedules obtained from (15) is less variable than the summation rates of these individually smoothed transmission schedules obtained by sequentially solving problem (14). Let R_{it}^* denote the optimal solution obtained by solving problem (15) and R'_{it} the optimal solution obtained by solving problem (14). Then we have the following proposition.

Proposition 1. $\max_{1 \leqslant t \leqslant T} \sum_{i=1}^{N} R_{it}^* \leqslant \max_{1 \leqslant t \leqslant T} \sum_{i=1}^{N} R'_{it}$.

Proof. Let S be the feasible set of problem (15) and S_i be the feasible set of problem (14). Since $S_i \subset S$, we have $\forall R'_{it} \in$

S. Furthermore, since R^*_{it}, $i = 1, \ldots, N$, is a set of optimal solution of problem (15), all feasible solutions including $R'_{it} \in S$ satisfy $\max_{1 \leqslant t \leqslant T} \sum_{i=1}^{N} R^*_{it} \leqslant \max_{1 \leqslant t \leqslant T} \sum_{i=1}^{N} R'_{it}$. □

The admission control problem (6)–(13) can be solved by a series of solutions of problem (15). To be specific, if problem (15) has a feasible solution, then all N requests can be admitted. Otherwise, we drop a request accessing the longest media and solve problem (15) with $N-1$ requests. The process continues until a feasible solution is found. Obviously, this process is also time-consuming.

5. Batched admission scheme

We now propose the BA scheme to further improve server resource utilization by integrating admission control, smoothing and batching techniques. At the beginning of each batching interval, the server executes the BA scheme. Upon receiving a request, the server keeps the requests in a queue until next batching interval and, then, makes a decision based on the admission and smoothing algorithm (as shown in Fig. 5). To reduce the computation time of long media, the server computes their transmission schedules period by period. Over a predefined period of time, called decision period T, only a part of a long transmission schedule is calculated. We define a media as a short media if its duration is less than T and a long media otherwise. Let Y_1 denote the set of requests accessing long media which have been admitted previously and need transmission schedules for next T frames, Y_2 denote the set of requests which access short media and are waiting for admissions, and Y_3 denote the set of requests which access long media and are just admitted in the current batching interval but still need transmission schedules for the next T frames.

Our heuristic method includes three steps. In the first step, the server uses the admission and smoothing algorithm to find feasible transmission schedules for all requests in Y_1. Since all requests in Y_1 have been admitted previously, their feasible transmission schedules should always exist in later decision periods. A new variable will be introduced to guarantee the feasibility in step 3. In the second step, the server uses the admission and smoothing algorithm again to exam all requests in Y_2. A request is admitted if there exists a feasible transmission schedule and rejected otherwise. In the third step, the server exams new long media requests which just arrived during the previous interval. We first use a simple admission method and, then, the smoothing algorithm to compute feasible transmission schedules over T for those admitted requests. As aforementioned, virtual capacity at round j, denoted by C^v_j, is introduced for those admitted requests accessing long media to guarantee that their transmission schedules always exist in later decision periods. Different from actual capacity $C^{\min}_j$, virtual capacity estimates a lower bound of the available capacity at round j by virtually allocating resources to requests according to media frame sizes. Currently, $C^v_j = \min\{B, C\} - \sum_{i \in Y_1 \cup Y_2} \sum_{t=(j-1)r+1}^{jr} d_{it}, t \leqslant \tau_i + L_i$. Obviously, $C^{\min}_j \geqslant C^v_j$. A new request accessing a long media F_i can be admitted and put into Y_3 if $\sum_{t=(j-1)r+1}^{jr} d_{it} \leqslant C^v_j, t \leqslant \tau_i + L_i \ldots$ and rejected otherwise. If admitted, we update virtual capacity $C^v_j = C^v_j - \sum_{t=(j-1)r+1}^{jr} d_{it}$. Since this admission decision is made according to the lower bound of available capacity, the transmission schedule always exists. For those admitted requests, the server computes actual transmission schedules over the current decision period by the smoothing algorithm. Note that $C^{\min}_j$ is actually used in the smoothing process and, hence, the utilization of system resources can be improved. Finally, we set $L_i = \max\{L_i - T, 0\}$ and $\tau_i = \tau_i + T$ if $L_i > 0$, and let $Y_1 = Y_1 \cup Y_3$ and $Y_3 = \emptyset$. This procedure is repeated at the beginning of the each batching interval.

In the remaining of this section, we address the details of the admission and smoothing algorithm. Note that constraints (7) can be written as $B_{i-1,t} + R_{it} = B_{it} + d_{it}$, $i = 1, \ldots, N$, $t = 1, \ldots, T$ which is the network flow format (see [1]). By introducing an auxiliary directed graph $G(V, E)$ depicted in Fig. 4, we develop a heuristic method based on multi-commodity network flow theory. The graph $G(V, E)$ consists of a finite set of nodes V and a set of arcs E, in which there are N source nodes, denoted by S_i, and $N*T$ sink nodes, denoted by V_{it}, $i = 1, \ldots, N$, $t = 1, \ldots, T$. The capacity is $C^{\min}_j$ on arc (S, W_i), $\overline{B}_i$ on arc (V_t, V_{t+1}), and infinite on other arcs. Assume that the source node S_i generates net outflow $\sum_{t=1}^{T} d_{it}$ and the sink node V_{it} requires d_{it} units of net inflow. Let $f^i(u, v)$ be the ith commodity flow on the arc $(u, v) \in E$. Then a set of multi-commodity network flow $f^i(u, v)$ satisfying constraints for the graph in Fig. 4 can be written as follows:

$$\sum_v f^i(u, v) - \sum_v f^i(v, u) = \begin{cases} \sum_{t=1}^{T} d_{it} & u = S_i \quad \text{for } i = 1, \ldots, N, \\ 0 & u \neq S_i, V_{it} \quad \text{for } i = 1, \ldots, N, \ t = 1, \ldots, T, \\ d_{it} & u = V_{it} \quad \text{for } i = 1, \ldots, N, \ t = 1, \ldots, T, \end{cases}$$

$$0 \leqslant f^i(u, v) \leqslant C^{\min}_j, \quad u = S, v = W_j, \quad j = 1, \ldots, m,$$

$$0 \leqslant f^i(u, v) \leqslant \overline{B}_i, \quad u = V_{i,t-1}, v = V_{it}, \ i = 1, \ldots, N, \quad t = 1, \ldots, T. \tag{16}$$

Proposition 2. *$\{(\delta_i, R_{it}, B_{it}) | i = 1, \ldots, N, \ t = 1, \ldots, T, \ j = 1, \ldots, m\}$ is a feasible solution of problem* (7)–(13) *if and only if there exists a feasible flow $\{f^i(u, v) : (u, v) \in E\}$ satisfying* (16).

Proof. First, if $\{(\delta_i, R_{it} B_{it}) | i = 1, \ldots, N, \ t = 1, \ldots, T, \ J = 1, \ldots, m\}$ is a feasible solution of problem (7)–(13), then we can construct a flow satisfying constraints (16) as follows. Let K be the number of non-zero δ_i's. Without loss of generality, we assume $\delta_i = 1$, $i = 1, \ldots, K$. Let $f^i(S_i, S) = \sum_{t=1}^{T} R_{it}$, $f^i(S, W_j) = \sum_{i=1}^{K} \sum_{t=(j-1)r+1}^{jr} R_{it}$,

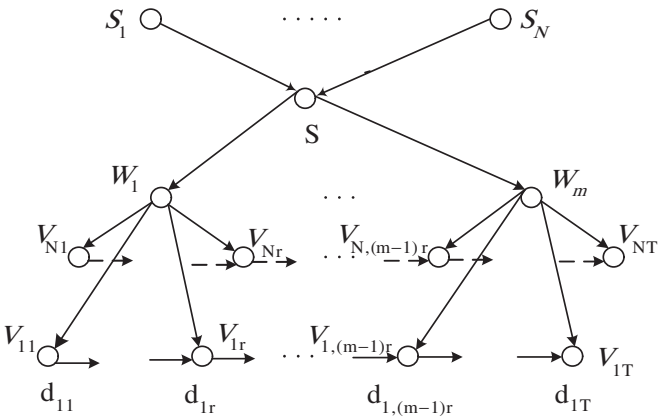


Fig. 4. Multi-commodity network flow.

$f^i(W_j, V_{it}) = R_{it}$ and $f^i(V_{it-1}, V_{it}) = B_{it}$. According to constraint (7), we have $f^i(W_j, V_{it}) + f^i(V_{it-1}, V_{it}) - f^i(V_{it}, V_{it+1}) = d_{it}$ for all nodes V_{it}. Summing equalities (7) for all $t = 1, \ldots, T$, we have $B_{i0} + \sum_{t=1}^{T} f^k(S_i, S) - \sum_{t=1}^{T} d_{it} = B_{iT}$. Since $B_{i0} = B_{iT} = 0$ from constraints (9) and (10), we know $f^i(S_i, S) = \sum_{t=1}^{T} d_{it}$, for all $i = 1, \ldots, N$. Hence, the network flow constraints hold for all S_i, V_{it}. Furthermore, the inflow of node S is $\sum_{i=1}^{K} f^i(S_i, S) = \sum_{i=1}^{K} \sum_{t=1}^{T} d_{it}$ and the outflow of node S is $\sum_{j=1}^{K} f^i(S, W_j) = \sum_{i=1}^{K} \sum_{t=(j-1)r+1}^{jr} R_{it} = \sum_{i=1}^{K} \sum_{t=1}^{T} d_{it}$, which implies that the flow conservation constraints hold. Since the inflow of node W_j, $j = 1, \ldots, m$, is $f^i(S, W_j)$ and the outflow is $\sum_{i=1}^{K} f^i(W_j, V_{it})$, the equality also implies the flow constraints for all W_j. Last, the capacity constraints in (7)–(13) imply that the capacity constraints in (16) hold. Similarly, we can prove the other side. □

From Proposition 2, we know that flow $f^i(W_j, V_{it})$ corresponds to a feasible transmission schedule R_{it}, and flow $f^i(V_{it-1}, V_{it})$ corresponds to buffer level B_{it}. The ith request can be admitted if we can find a set of feasible flow $f^i(W_j, V_{it})$ for all $t = 1, \ldots, T$. Also, the total number of admitted requests is the number of arcs (S_i, S) on which the flow is non-zero.

In the following, we omit the superscript of $f^i(u, v)$ to indicate that we are dealing with an arbitrary commodity. In order to explain the process of finding a feasible flow from source node S_i to V_{it}, we introduce the concepts of residual network and augmenting path. Given a network $G(V, E)$ and flow f, the residual network of G introduced by f is denoted by $G_f(V, E_f)$, where each arc $(u, v) \in E$ is replaced by two arcs, namely, the forward arc (u, v) with capacity $c_f(u, v) = c(u, v) - f(u, v)$, and the backward arc (v, u) with capacity $c_f(v, u) = f(u, v)$. Also E_f contains all the arcs with positive capacities. An augmenting path p_i is a directed path from a source node S_i to a sink node V_{it}.

Our heuristic iterates from time 1 to T, and first examines the commodities in Y_1, then Y_2, and Y_3. Repeatedly, the network flow between each pair of source and sink nodes is increased by finding augmenting paths between the pair of nodes until the demand of the sink node is reached. The detailed procedure is as follows. In the beginning, the flows on all arcs are zero. Then, for each commodity i, we increase the flow value by finding augmenting paths from the source nodes S_i to every sink nodes V_{it}, $t = 1, \ldots, T$. To obtain a smoothing transmission schedule, we exploit the buffer space at both the server and the clients. Hence, as many flows as possible are sent on arc (V_{it-1}, V_{it}). The process is repeated for each commodity until the flow satisfies $f(V_{it-1}, V_{it}) + f(W_i, V_{it}) \geqslant d_{it}$. For commodities in Y_1, flows can always be found to satisfy this condition for all t. We start by finding the augmenting path for $i \in Y_2$ from the shortest media. However, no augmenting path may be found for some t and the process finally stops at $f(V_{it-1}, V_{it}) + f(W_i, V_{it}) < d_{it}$. If the process stops, then the request corresponding to this commodity will be rejected. In addition, we drop the ith commodity by deleting nodes S_i, V_{it} and all adjacent arcs in network $G(V, E)$ and residual network $G_f(V, E_f)$. At the same time, since commodity i is dropped, we need to recover the capacities on all the arcs which were used by commodity i to augment the flow. In the next step, we check the requests accessing the long media with the simple admission control rule. If $Y_3 \neq \emptyset$, then we repeat the process in Y_1; otherwise, the computation of transmission schedules over T stops. Finally, the number of admitted requests is the number of arcs (S_i, S), $i = 1, \ldots, N$, with positive flows and the transmission schedule R_{it} is the flow value $f(W_i, V_{it})$ on arc (W_i, V_{it}), $t = 1, \ldots, T$. This process is summarized in Fig. 5.

Note that we only need to find augmenting paths between each pair of source and sink nodes such that the flow satisfies the sink node's demand. Let $D = \max\{d_{it}, i = 1, \ldots, N, t = 1, \ldots, T\}$ and let $|E^i|(|V^i|)$ denote the number of arcs (nodes) of the ith commodity (as shown in Fig. 4). We have $|E^i| = 2T + T/r + 1 = O(T)$ and $|V^i| = 2(T - 1) + T/r + 2 = O(T)$. Then, it requires at most D iterations to find augmenting paths between each pair of source and sink nodes. Also, in each iteration, at most $O(T)$ time is required to find an augmenting path. Hence, the execution time for each commodity is at most $O(DT^2)$. Therefore, the total running time for N commodities is $O(NDT^2)$. Since the computation depends on the decision period T, there is a tradeoff between the computation time and bandwidth utilization. With a smaller T, the computation time is smaller but the bandwidth utilization may be low. The reason is that the work-ahead scheme can only be applied on a shorter period and thus fewer frames can be transported ahead resulting in low bandwidth utilization.

6. Numerical results

In this section, we examine the impact of smoothing algorithms on buffer and bandwidth usages and evaluate the effectiveness of the BA scheme by comparing it with other methods. Each algorithm runs on a PC with Pentium III 800 Hz CPU and 256 MB of memory. We generate 10 video segments which are randomly chosen from MPEG coded *Star Wars* from Bellcore [10]. The lengths of these segments vary from 10 s to 5 min. Since the movie was compressed at a rate of 24 frames/s, the length of the video clip is measured by frame duration (e.g.

Admission and Smoothing Scheme

Initialize $G(V,E)$, $G_f(V,E_f)$
for $i \in Y_1$
  Find Augmenting Path for ith stream
for $i \in Y_2$
  if $f(W_i,V_{it})+f(V_{i,t-1},V_{it}) < d_{it}$
    then update $G(V,E)$, $G_f(V,E_f)$ by dropping edges related to nodes S_i,V_{it}
      update flow in $G_f(V,E_f)$ by reducing the flow related to i
      update Y_2
for $i \in Y_3$
  Find Augmenting Path for ith stream

Find Augmenting Path for ith stream

for $t \leftarrow 1$ to T
  while $f(V_{i,t-1},V_{it}) < d_{it}$ and there exists augmenting path $p_i$including edge $(V_{i,t-1},V_{it})$
    do update the flow on each $(u,v) \in p_i$
  while $f(W_i,V_{it})+f(V_{i,t-1},V_{it}) < d_{it}$ and there exists augmenting path p_i
    do update flow on each $(u,v) \in p_i$

Fig. 5. Modified multi-commodity network algorithm for admission control and transmission schedule.

5 min = 7200 frames). Each request has a start-up delay randomly chosen from interval [1,10] (in frames). In order to study the sensitivity of various algorithms to transmission and admission rates, the client buffer sizes vary from 0 to 1 Mb (measured in bits) and the server capacities from 1.5 to 5 Mb.

First, we study the sensitivity of the client buffer size to the maximum smoothed transmission rate. Fig. 6 shows maximum transmission rates and client buffer usages when client buffer sizes are 32, 64, 512 kB and 1 Mb, respectively. The smoothed transmission schedules are obtained by solving problem (14) with CPLEX solver. To examine the impact of the client buffer size on transmission rates, we relax capacity constraints on the server site. The example video length is randomly selected and has 6120 frames. It is well known that transmission rate variability can be reduced by using the client buffer. As shown in Fig. 6, maximum transmission rate decreases when client buffer size increases from 32 to 512 kB. However, the transmission rate does not reduce further when the buffer size increases from 512 kB to 1 Mb. The reason is that initial frame sizes of the example video are much larger than those of other time intervals. These initial frames cannot be sent ahead of schedule and stored in client buffer temporally even when client buffer space is available. Hence, transmission rates depend not only on buffer sizes but also frame size distributions.

We have demonstrated that the transmission rate variability could be reduced by PSB smoothing methods. Actually, it can also be reduced by aggregating multiple streams. Authors in [18] have shown that the aggregate bit rate of individually smoothed streams is less bursty than the bit rate of these streams sent in an unsmoothed manner. By solving problem (15), we can show that the peak rate of aggregate streams is less than the summation of individually smoothed streams. Consider 10 incoming requests for different streams. The stream lengths, start-up delays, and client buffer sizes are randomly generated from the predefined intervals. We sum the transmission rates of 10 individually smoothed streams over the entire stream period and present the results in Fig. 7. The peak transmission rate is 316.43. Fig. 8 shows the transmission rates of multiple streams obtained by the aggregation method and the peak rate is 215.75. The peak transmission rate by aggregation method is less bursty than the summation of individually smoothed streams and the peak rate is reduced by approximately 30%.

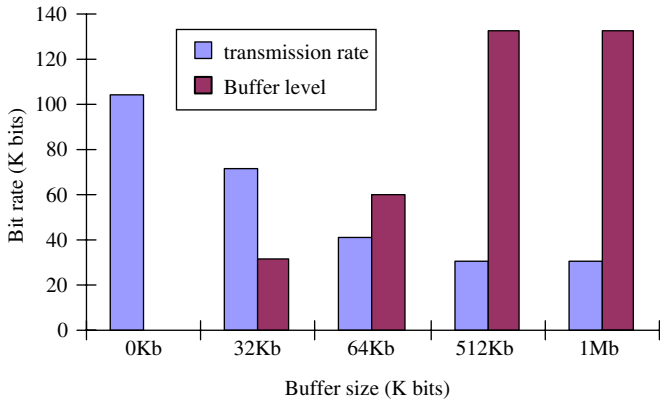


Fig. 6. Maximum transmission rate and client buffer level with different buffer spaces.

An advantage of the BA scheme is its execution time. Note that the decision period T is a dominant factor in computation complexity of the admission and smoothing algorithm. The execution time is mainly determined by T. We study the execution time with different lengths of T and present the results in Fig. 9. As can be seen, a longer decision period yields a longer execution time. The BA scheme takes roughly 30 ms when the decision period T is chosen to be 5 min.

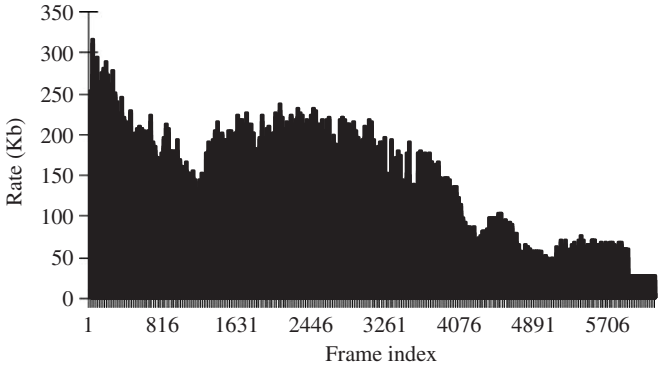


Fig. 7. Aggregation of individually smoothed streams.

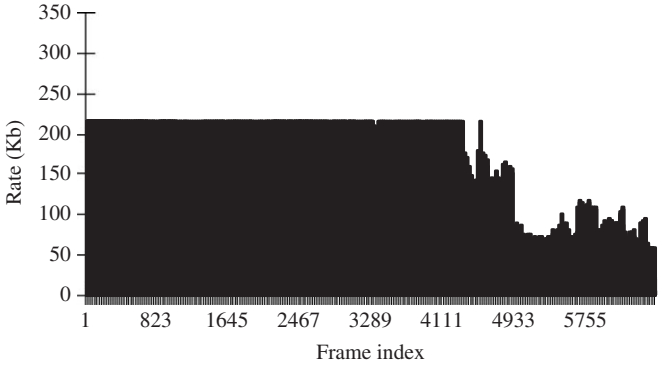


Fig. 8. Aggregation rate of multiple streams by solving problem (15).

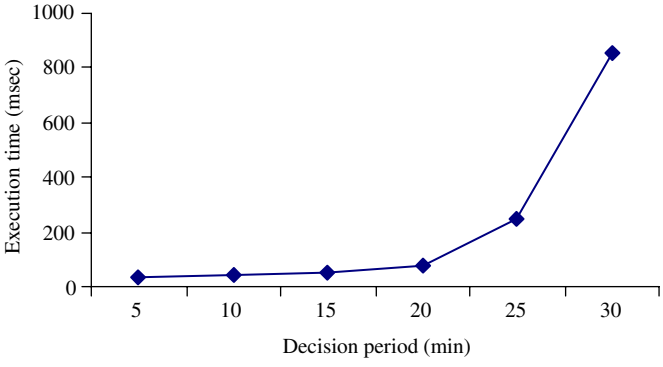


Fig. 9. Execution times with different decision periods.

To test the effectiveness of the BA scheme, we compare it with the PSB scheme proposed in Section 4.1 and the batched vbrSim (BvbrSim) scheme. The BvbrSim scheme is actually a modified vbrSim algorithm. Since the vbrSim algorithm [15] does not consider the batching technique and the constraint of client buffer space, we modify it for comparison. In the BvbrSim scheme, we delay requests for batching and make admission decisions at the beginning of the next batching interval. In PSB, we check individual requests one by one and reserve enough disk bandwidth and buffer space for the peak rate of a smoothed stream schedule. The criterion of the admission control algorithm is to admit as many requests as possible without

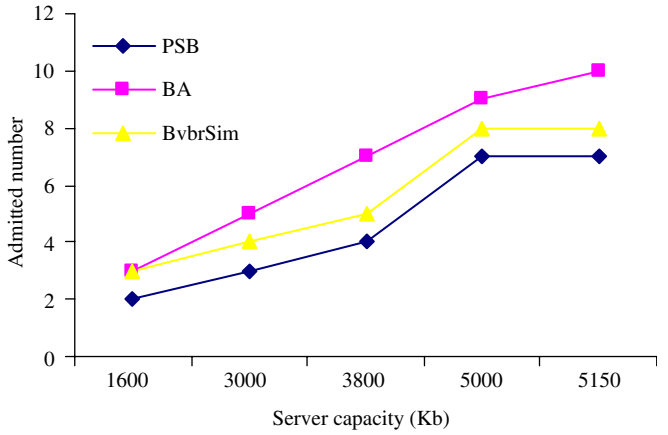


Fig. 10. Comparison of three admission control algorithms when client buffer sizes are 512 kB.

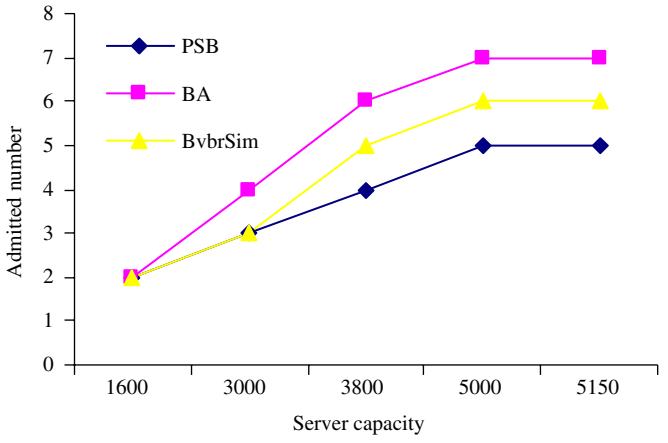


Fig. 11. Comparison of three admission control algorithms when client buffer sizes are 32 kB.

violating their QoS requirements. We compare the numbers of requests admitted by the three approaches. Note that the maximum number of admissible streams depends on smoothness of transmission schedules which are constrained by server capacity and client buffer space. To analyze the sensitivity of server buffer size to admission rates, we test the three algorithms with server buffer capacity varying from 1.5 to 5 Mb and fixed client buffer capacity. Different client buffer sizes are used for different runs, which are 32 and 512 kB. Note that for each run, the client buffer size is the same for all 10 clients. To simplify the computation, the disk read capacity is represented by bits instead of by number of blocks in this numerical example. Fig. 10 shows the numbers of admissible streams with different admission algorithms when all client buffers are of size 512 kB. Since all three schemes use read-ahead techniques and client buffer sizes are large enough, rate variability of transmission schedules after smoothing is very small. In this case, two algorithms with batching techniques can accept more requests than the PSB scheme. Furthermore, by integrating aggregation, smoothing and batching, the BA algorithm outperforms the other two algorithms. Fig. 11 shows the comparison

of the three algorithms when client buffers are of size 32 kB. Due to smaller client buffer space, rate variability of transmission schedules becomes larger. As we can see, the numbers of admissible streams of all three algorithms decrease, compared to the case when client buffer sizes are 512 kB. But the relative performance among the three algorithms is still the same, i.e., BA scheme outperforms all other schemes.

7. Conclusion

We have studied the problem of transmitting stored video from a server to multiple clients for the given level of available system resources, such as disk bandwidth and buffers at both client and server sites. Due to the stringent QoS requirement for streaming media, an admission control mechanism must be used to provide guaranteed service for clients. We characterize the admission control problem by a mixed integer programming problem. Different from other approaches for improving the utilization of system resources, the BA scheme can utilize system resources more efficiently by integrating admission control, transmission rate smoothing, and batching. In addition, we compute the transmission schedules for long videos period by period and, thus, their execution times are short. Based on network flow algorithms, the running time of the BA algorithm is $O(NDT^2)$. The computation time depends on the pre-defined period T. If we need a fast execution time, then we can use a smaller T, though it may yield lower bandwidth utilization. Numerical studies show the BA scheme outperforms the other methods. As part of ongoing work, we are taking into account some other techniques, such as patching, to further improve the utilization of critical resources.

References

[1] A. Aggarwal, J.K. Park, Improved algorithms for economic lot size problems, Oper. Res. 41 (3) (1993) 549–571.

[2] C. Aggarwal, J. Wolf, P. Yu, On optimal batching policies for video-on-demand storage servers, in: Proceedings of MULTIMEDIA'96.

[3] D. Anderson, Meta-scheduling for continuous media, ACM Trans. Comput. Systems 11 (3) (1993).

[4] E. Biersack, F. Theiesse, C. Bernhardt, Constant data length retrieval for video servers with variable bit rate streams, International Conference on Multimedia Computing and Systems (ICMCS) 1996, pp. 151–155.

[5] E. Chang, A. Zakhor, Cost analyses for vbr video servers, IEEE Multimedia 3 (4) (1996).

[6] A. Dan, D. Sitaram, P. Shahabuddin, Scheduling policies for an on-demand video server with batching, in: Proceedings of the 2nd ACM Multimedia Conference, 1994.

[7] W. Feng, F. Jahanian, S. Sechrest, An optimal bandwidth allocation strategy for delivery of compressed prerecorded video, ACM/Springer-Verlag Multimedia System J. 5 (5) (1997) 297–309.

[8] W. Feng, J. Rexford, A comparison of bandwidth smoothing techniques for the transmission of prerecorded compressed video, in: Proceedings of the IEEE INFOCOM, Kobe, Japan, April, 1997, pp. 58–66.

[9] W. Feng, S. Sechrest, Critical bandwidth allocation for delivery of compressed video, Comput. Commun. 18 (10) (1995) 709–717.

[10] M.W. Garrett, W. Willinger, Analysis, modeling, and generation of self-similar vbr video traffic, in: Proceedings of the ACM SIGCOMM, September 1994, pp. 269–280.

[11] M. Krunz, S.K. Tripathi, Impact of video scheduling on bandwidth allocation for multiplexed MPEG streams, Multimedia Systems 5 (1997) 347–357.

[12] S.K. Kweon, K.G. Shin, Transmission of aggregate VoD streams using playback rate, REAS'01, Taipei, Taiwan, 2001, pp. 205–215.

[13] J. Liebeherr, D.E. Wrege, D. Ferrari, Exact admission control in networks with bounded delay services, IEEE/ACM Trans. Networking 4 (1996) 885–901.

[14] S.C. Liew, H.H. Chan, Lossless aggregation: a scheme for transmitting multiple stored vbr video streams over a shared communications channel without loss of image quality, IEEE J. Selected Areas Commun. 15 (6) (1997) 1181–1189.

[15] D. Makaroff, G. Neufeld, N. Hutchinson, An evaluation of VBR disk admission algorithms for continuous media file server, ACM Multimedia, Seattle, WA, 1997.

[16] J.M. McManus, K.W. Ross, Video-on-demand over ATM: constant-rate transmission and Transport, IEEE J. Selected Areas Commun. 14 (6) (1996) 1087–1098.

[17] J. Rexford, D. Towsley, Smoothing variable-bit-rate video in an internetwork, in: Proceedings of the SPIE Conference on Performance and Control of Network Systems, Dallas, TX, November 1997, pp. 345–357.

[18] J.D. Salehi, Z.L. Zhang, J. Kurose, D. Towsley, Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing, IEEE/ACM Trans. Networking 6 (4) (1998) 397–410.

[19] S. Sheu, A. Hua, T.H. Hu, Virtual batching: a new scheduling technique for video-on-demand servers, in: Proceedings of the 5th DASFAA Conference, Melbourne, Australia, 1997.

[20] O. Verscheure, C. Venkatramani, P. Frossard, L. Amini, Joint sever scheduling and proxy caching for video delivery, Comput. Commun., 25, March 2002 (Special issue).

[21] H.M. Vin, P. Goyal, A. Goyal, A. Goyal, A statistical admission control algorithm for multimedia servers, in: Proceedings of the 2nd ACM International Conference on Multimedia, San Francisco, CA, October, 1994.

[22] H.M. Vin, P. Goyal, A. Goyal, A. Goyal, An observation-based approach for designing multimedia servers, in: Proceedings of the IEEE International Conference on Multimedia Computing and Systems, Boston, MA, May, 1994, pp. 234–243.

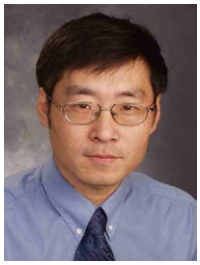
[23] Y. Wang, Z. L. Zhang, D. Du, D. Su, A network conscious approach to end-to-end video delivery over wide area networks using proxy servers, IEEE INFOCOM'98.

[24] W. Zhao, S. Tripathi, Bandwidth-efficient continuous media streaming through optimal multiplexing, in: Proceedings of the 1999 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems.

Zhonghang Xia received the B.S. degree in applied mathematics from Dalian University of Technology in 1990, the M.S. degree in Operations Research from Qufu Normal University in 1993, and the Ph.D. degree in Computer Science from the University of Texas at Dallas in 2004. He is now an assistant professor in the Department of Computer Science, Western Kentucky University, Bowling Green, KY. His research interests are in the area of multimedia computing and networking, distributed systems, and combinatorial optimization.

Dr. I-Ling Yen received her B.S. degree from TsingHua University, Taiwan, and her M.S. and Ph.D. degrees in Computer Science from the University of Houston. She is currently an Associate Professor of Computer Science at University of Texas at Dallas. Dr. Yen's research interests include fault-tolerant computing, security systems and algorithms, distributed systems, Internet technologies, E-commerce, and self-stabilizing systems. She had published over 100 technical papers in these research areas and received many research awards from NSF,

DOD, NASA, and several industry companies. She has served as Program Committee member for many conferences and Program Chair/Co-Chair for the IEEE HASE, COMPSAC, ISADS, etc.



Donglei Du received the B.S. degree in Mathematics from Fudan University, Shanghai, China, in 1990, M.S. degree in Operations Research at Shandong University, Jinan, China, in 1993, and two Ph.D. degrees: one in Operations Research at Institute of Applied Mathematics, Chinese Academy of Sciences, Beijing, China, in 1996, and another in Computer Sciences at the University of Texas, Dallas, USA, in 2003. He is now an Associate Professor of Management Sciences at the Faculty of Business Administration, University of New Brunswick, Fredericton, Canada. His research interests include combinatorial optimization, online and approximation algorithms, scheduling theory, graph theory and network flows, supply chain management, computational game theory, and financial mathematics.

Dr. Peng Li received his M.S. and B.S. in computer science from the Renmin University of China, Beijing, China in 1996 and 1999, respectively. He received his Ph.D. in computer science from the University of Texas at Dallas in 2005. Currently, he is a visiting assistant professor of Computer Science at Western Kentucky University. His research interests include database and web applications, transaction processing, data replication, distributed and internet computing, database security and e-commerce.