A Scalable Video-On-Demand System Using Multi-Batch Buffering Techniques

Cyrus C. Y. Choi and Mounir Hamdi, *Member, IEEE*

***Abstract*—A Video-on-Demand (VOD) system delivers videos on demand over an installed network. Due to the large size of digitized videos, expensive video servers with high I/O capability are needed in order to provide VoD services in metropolitan areas. In addition, there is a great need for efficient networking distribution/interaction schemes so that the video servers can serve as many clients as possible. In particular, because of scalability problems, the classical unicast VoD system is not suitable for large-scale deployments. In this paper, a highly scalable VoD system with a low per-user cost is described and evaluated.**

We first analyze the performance degradation problems using recently proposed VoD systems, namely batched and centralized-buffer VoD systems that occur during the handling of interactions. Then a new system called the *Multi-Batch Buffer* (MBB) system, which attempts to solve these problems, is proposed. The proposed system handles a majority of interaction requests by scalable buffering techniques employed in the buffer of the local servers and the set-top boxes (STBs). We have performed extensive simulation for the analysis and performance evaluation of our proposed VoD system. The simulation results will demonstrate that our VoD system is very scalable and outperforms related state-of-the-art VoD systems.

***Index Terms*—Multicast, performance evaluation, slip-and-merge, Video-on-Demand.**

NOMENCLATURE

ATM:	Asynchronous Transfer Mode
IVS:	Interactive Video Stream
MBB:	Multi-Batch Buffer
PVS:	Potential Video Stream
RSV:	Real Video Stream
SAM:	Spilt-and-Merge
STB:	Set Top Box
SVS:	Source Video Stream
TVS:	Target Video Stream
VoD:	Video on Demand
VS:	Video Stream
VSR:	Video Start Requests
VIR:	Video Interaction Request
VVS:	Virtual Video Stream

I. INTRODUCTION

A VIDEO-ON-DEMAND (VoD) is a network-based system that delivers videos on demand. Other applications of VoD systems include home shopping and broadband Internet services [20]–[22]. Most of the commercial VoD systems, such as [1], use a unicast networking approach which makes them very costly. In addition, they are not scalable because in these systems a video stream (VS) can only serve one customer. As a result, the per-user cost of these unicast VoD systems can be higher than that of video rental shops [13], [17], [18]. The current research focus on VoD systems is on how to lower their cost and make them more scalable. In this paper, we propose a VoD system with these capabilities. Before explaining the proposed system, background information about VoD systems and some research work done by others are described.

Previous studies [2], [4], [5], [6], [7], [10], [11], [19] have shown that the batching technique, which serves the video start requests (VSRs) requesting the same video by a multicast VS, can reduce the VSR and video interaction requests (VIRs) blocking probabilities of VoD systems. One of the problems of the batching VoD systems is that customers have to wait for a while before they can watch the requested videos. Another problem is that the VSR and VIR blocking probabilities increase as more customers perform VCR-like interactions. This is because when a customer issues a VIR, he is spilt from the multicast group and has to be served by a new VS unless at that moment there is an existing VS delivering the same video and playing at the required play point, and this chance is quite low (experiments have shown that it is less than 0.03). In particular, as more customers issue VIRs, the system will degrade to a unicast system. Hence, special interactions handling mechanisms are required under batching VoD systems. Section I-B overviews some of the proposed mechanisms.

The system proposed by Ammar *et al.* [7] attempts to solve the system degradation problem by providing discontinuous rather than continuous interaction functions. Although this system uses the least additional resources compared to the systems discussed hereafter, the degradation of interactions because of this discontinuous property is not acceptable in high quality VoD systems. In the same paper, the authors have proposed another system that uses an STB buffer to store past and future video frames to provide limited continuous interactions. However, the system will degrade to a unicast system if the required frames are not stored in the buffer after performing the interactions.

The Spilt-and-Merge (SAM) system proposed by Li *et al.* [2], allows continuous interactions and reduces the number of VSs required for handling interactions. However, the problem is that the interaction handling mechanism of the SAM system demands a high IO power buffer in the access node, or referred to as *local server* in this paper. In the worst case, all of the customers in the SAM system perform VIRs and the total number of

Manuscript received May 16, 2002; revised January 31, 2003.
The authors are with the Department of Computer Science Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong (e-mail: cyrus.choi@alumni.ust.hk; hamdi@cs.ust.hk).
Digital Object Identifier 10.1109/TBC.2003.813435

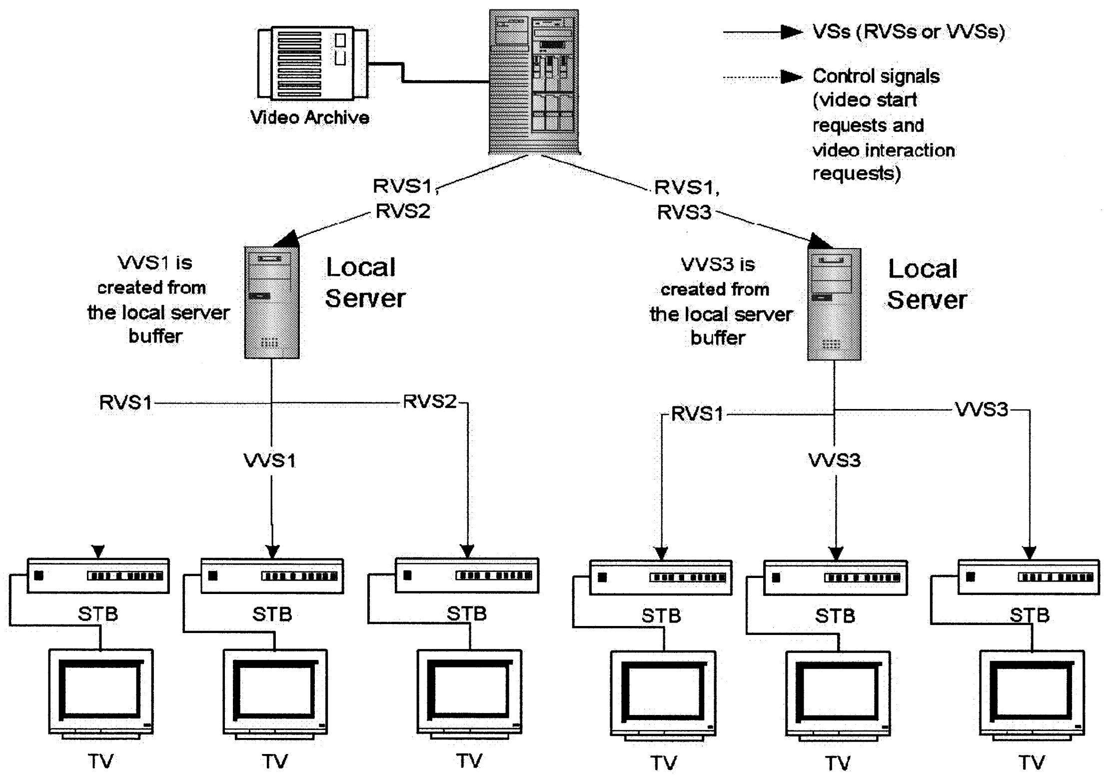


Fig. 1. Architecture of MBB system.

VSs required would be equal to the total number of customers, and this resource requirement becomes similar to that in unicast systems. In addition, the IO workload of the buffer of the local server, in order to produce a VS, is greater than that of a traditional video server. This is because the buffer of the local server, compared with the disk of video server, also has to receive the frames prefetched from the video server and writes them onto the buffer, besides reading the required frames from the buffer and transmitting them to the customers. Thus, given the same number of VSs available in a video server and a local server, the hardware requirement of the local server in terms of disk IO and network bandwidth is at least twice than that of the video server.

While our proposed system uses similar interaction handling mechanism to that in SAM, it is shown in Section III how our system reduces the workload of the local server by effectively using a slotted start time and the STB buffer.

Shin *et al.* [5] propose using the set top boxes' (STBs) buffer and the interactive video streams (IVSs) created from a video server to handle interactions. For convenience, it is named as STB *system* in this paper (they did not name it in their paper). The main difference, when compared with the SAM system, is that the STB system uses STBs' buffer instead of the local server buffer to handle interactions. The STB system, similar to the one proposed by Ammar *et al.* [7], uses the buffer of the STB to keep a window of past and future video frames to serve interactions. If the required frames are not in the buffer, then the STB system handles the interaction by a free IVS and the virtual video stream (VVS) created by prefetching another VS. The drawback of the STB system compared with the SAM system is that the additional cost of the STB buffer might be high in a large-scale deployment.

The focus of this paper is on how to handle interaction requests in batching VoD systems, so that the benefits of batching can still be maintained. The proposed system is named the *Multi-Batch Buffer* (MBB) system. As its name suggests, this system is able to serve VSRs as well as VIRs using the batching technique.

The rest of the paper is organized as follows. In Section II, we introduce the architecture of the MBB system. Section III is devoted to explaining the VSRs and VIRs handling mechanisms of this system. Detailed comparisons of the MBB system with the SAM and the STB systems are given in Section IV. Then, the simulation results of the three systems and the discussion of the results are given in Section V. Finally, this paper concludes at Section VI.

II. THE ARCHITECTURE OF THE MBB SYSTEM

The basic system architecture of the proposed MBB system is shown in Fig. 1.

Similar to the architecture of the unicast VoD system described in [3], the video servers of the MBB system store videos in secondary and/or tertiary storage devices [15]. The VSs created by fetching video frames stored in the disks of the video server are called real video streams (RVSs). Similar to other batching VoD systems described in [2], [4], [5], [6], [7], the MBB system is operated under a network environment that allows multicast data delivery, thus, the RVSs and the IVSs of the MBB system can deliver data in a multicast fashion. For example, according to [2], the network environment can be a Hybrid Fiber Coax network running in Asynchronous Transfer Mode (ATM).

An additional point that is worth mentioning is the multicasting capabilities of the Internet switches/routers in our VoD system. Our system requires that the Internet switches/routers have a multicasting capability. However, our system is independent from the way multicasting is being handled by these switches and routers. If the switches/routers have advanced and very fast multicasting capability (e.g., using hardware ASIC), then obviously this will help the scalability and performance of our VoD System. On the other hand, if the switches/routers rely on slow software schemes for perform their multicasting, then it can have negatively affect the performance of our system.

The local servers of the MBB system, located between the video servers and the STBs, are used to create virtual video streams (VVSs), which are constructed by prefetching RVSs or VVSs at the local servers' buffer. The main function of a VVS is to handle VCR-like interaction requests such as forward, rewind and pause. Similar to RVSs, VVSs can deliver data in a multicast fashion. While the local server of the MBB system has similar functions compared with the access node in the SAM system, the name "local server" is chosen in this paper to indicate that it has similar functions compared with a video server and it is closer to the customer side [14].

Set-top boxes (STBs) are located at the customers' side. Besides the basic hardware components in STBs described in [3], the STBs of the MBB system have an additional buffer space for prefetching RVSs and VVSs. This buffer can store several minutes of video frames and can be either a hard disk or RAM. The cost of this buffer is affordable compared to other system components. For example, a 5 minutes MPEG-1 compressed video of play rate 1.5 Mbps requires about 56 MB space of storage, which costs around US $100 in year 2000 if it is stored in RAM and much less than that if it is stored in a hard disk. The virtual video stream created from this buffer is called an STB-VVS. The STB-VVS, is not like the RVS and the VVS, which deliver video frames via the network, it just exists logically and represents the current playing location. In case a customer does not have an STB, then the system will transfer the STB functionalities to the closest storage of the customer side. This is similar to the way it is being handled by the SAM system. Of course, this will have a negative effect on the customer's interaction response.

As can be seen from this description, this architecture is based on a hierarchy of storage devices. Further performance improvement by increasing this hierarchy of storage devices, like caching techniques in computer systems, depends on many factors such the size of the whole VoD system, size and access speed to the storage devices, and speed of the transmission links.

For simplicity, only two local servers and one video server are shown in Fig. 1. However, this does not mean that the MBB system is limited to this configuration. The MBB system can has 4 different number of video servers, local servers, but the following points should be emphasized:

- All RVSs generated by the video server are broadcast to all local servers.
- All the local servers work independently. Thus, the VVSs created from a local server only serve customers under this local server and the VVSs are not broadcast to other local servers and their customers.

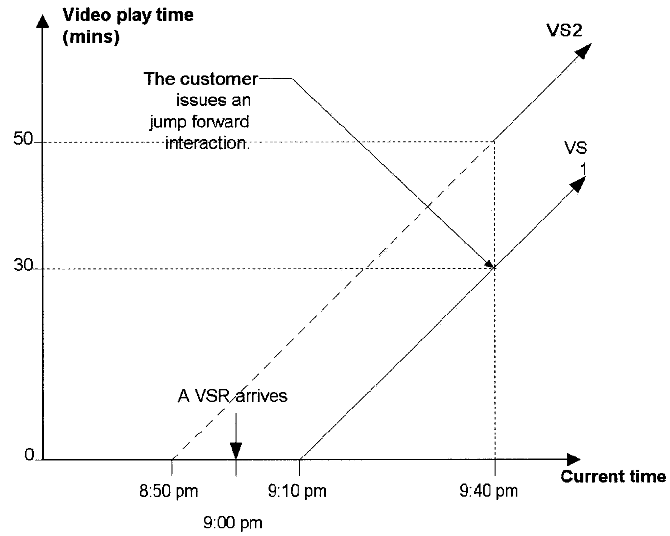


Fig. 2. A graph to visualize the state of the MBB system.

- For simplification, in the examples and discussions hereafter, there is only one local server and one video server in the MBB system.

III. The Working Mechanism of the MBB System

The working mechanism of the MBB system can be divided into two parts: The handling mechanism of the VSRs and that of VIRs, which are explained in Sections III-A and Section III-B respectively. Fig. 2 introduced in [7] is used in this paper to help readers understand how VSRs and VIRs are handled. The x-axis and the y-axis represent the current time and the current playing location or playtime of a VS, respectively. The figure shows that a customer sent a VSR at 9:00 pm, and was served by VS 1 at 9:10 pm. Later, an interaction request from the customer jump from playpoint 20 minutes to 50 minutes. To serve this request, a VS logically starts as 8:50 pm but physically starts at 9:40 pm is created, and they are referred to as logical and physical start times respectively. The start time of a VS refers to the logical start time of this VS unless specified.

A. The Handling Mechanisms of the Video Start Requests (VSRs)

In brief, the main difference in handling VSRs between the MBB system and other batching systems is on the start time calculation of a new RVS. Some terms are defined here for clarification. $\boldsymbol{T_b}$ denotes the batch time. A VS playing the video requested by a VSR or a VIR, or playing the same video of another VS is termed a *potential video stream* (PVS) of that request or of that VS. And according to whether the VS is waiting for the requests or it is serving the customers, the VS is in the reserve state or in the operation state. In the following examples we set $\boldsymbol{T_b}$ to be 10 minutes and the arrival time of the VSR that we are interested in is 9:00 pm.

Case I) A PVS of the New Arrived VSR is in the Reserve State: When a VSR arrives at the video server, if there is a PVS of the VSR in the reserve state, the customer will join this PVS and it will become the target video stream TVS of this request.

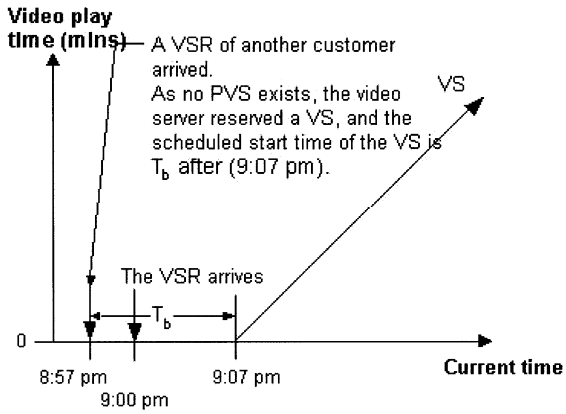


Fig. 3. Case i) A PVS exists on reserve state.

Note that the PVS begins not earlier than $\boldsymbol{T_b}$ from now, otherwise, the VS is already in the operation state. This VSR and the VSRs grouped by the batch group of the TVS will be served together by the TVS at the scheduled start time. The start time of a VS is determined when the VS changes from the free state to the reserve state, and is described in Case ii).

Case II) All the PVSs of the New Arrived VSR are in the Operation States: If there is no PVS of the newly arrived VSR in the reserve state but there is at least a PVS of the VSR in the operation state, then a free RVS is required to serve the VSR. If the free VS is available, it is put into the reserve state, and scheduled to start at time $\boldsymbol{i * T_b}$ after the PVS that has the largest start time, where $\boldsymbol{i}$ is the smallest integer such that **the start time of the last PVS $+\boldsymbol{i} * \boldsymbol{T_b} >=$ the current time**. For example, if $\boldsymbol{T_b}$ is 600 second or 10 minutes, the start time of the last PVS is 8:45 pm and the current time is 9:00 pm, then $\boldsymbol{i}$ equals to 2 and the RVS will be scheduled to start at 9:05 pm. This start time allocation scheme is referred to as the slotted start time allocation. The reason for using this time allocation is to ensure the start time separation between all the PVSs of all VSRs are integral multiples of $\boldsymbol{T_b}$. This enhances the MBB system performance during the handling of the VIRs, and detail explanations will be given in Section III-B.

Case III) No PVS of the New Arrived VSR is in the System: If none of the above cases hold, then there is no PVS of the new arrived VSR in the system, and a free RVS is required to serve the VSR. This RVS is put in the reserve state, and scheduled to start at time $\boldsymbol{T_b}$. If there is no free RVS, the VSR is blocked.

B. The Handling Mechanisms of the Video Interaction Requests (VIRs)

1) A Classification of Video Interactions: According to the classification in [2], [5], interactions can be classified into the following types: Play, resume, stop, pause, jump backward and forward, fast forward and rewind, and slow motion. All of the functions of the above interactions are self-explanatory except for jump forward and backward where the playpoint is changed from the original location to the new location instantly, while for fast forward and fast rewind the viewers can watch the intermediate frames which are shown at a faster speed. For simplicity, only jump forward interactions are illustrated in the examples hereafter, and only jump forward and jump backward interactions are experimented in the simulations in Section V. Brief descriptions about how to handle other types of the interactions are given in Section III-C.

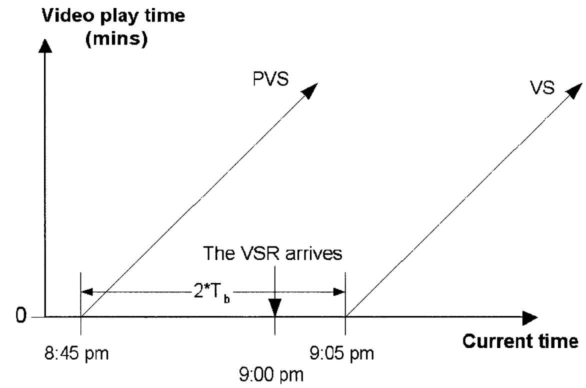


Fig. 4. Case ii) A PVS exists in operation state.

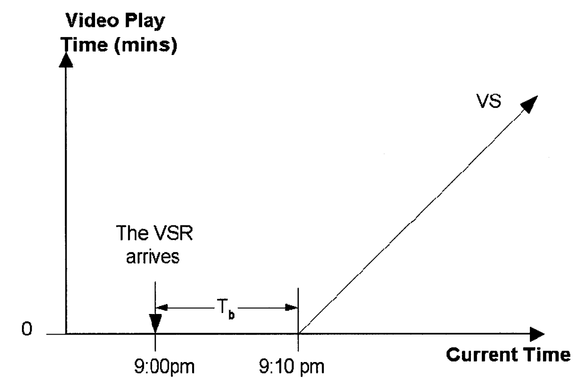


Fig. 5. Case iii) No PVS exists.

2) The Working Mechanisms of the Buffer of the Local Servers and the STBs: The buffer located at the local servers and the STBs are important components in the MBB system. In brief, the functions of the buffers are to fetch the required video frames from VSs called Source VSs or SVSs, store these frames in the buffer, and send these frames to the customers later. The SVS plays the same video of SVS but with greater logical start time, and the logical start time difference between them cannot be greater than the size of the buffer unit. The STB uses the whole buffer while the local server uses part of its buffer space to cache the VS. The VS created by the buffer is called virtual VS or VVS. The time period when the buffer only fetches the video frames is called the *prefetch stage*. Afterward, the buffer, not just fetches the frames from the SVS, but also delivers the stored frames to the customers, and the buffer is in the *operation stage*.

3) Detail Handling Mechanisms of the Video Interaction Requests (VIRs): The VIR handling of the MBB system can be classified into three types according to the position of the SVS of the VIR being found: I) $T_{diff} <= T_{stbbuffer}$, II) $T_{stbbuffer} < T_{diff} <= T_b$ and III) $T_{diff} > T_b$, where T_{diff} is the start time difference between the SVS and the TVS of the VIR, $T_{stbbuffer}$ is the length of the video frames the buffer of the STB can hold, and T_b is the batch time. $T_{stbbuffer}$ is a system parameter, which is set to $T_b/2$ in the examples below, and is set to T_b and $T_b/2$ in the simulation experiments in Section V.

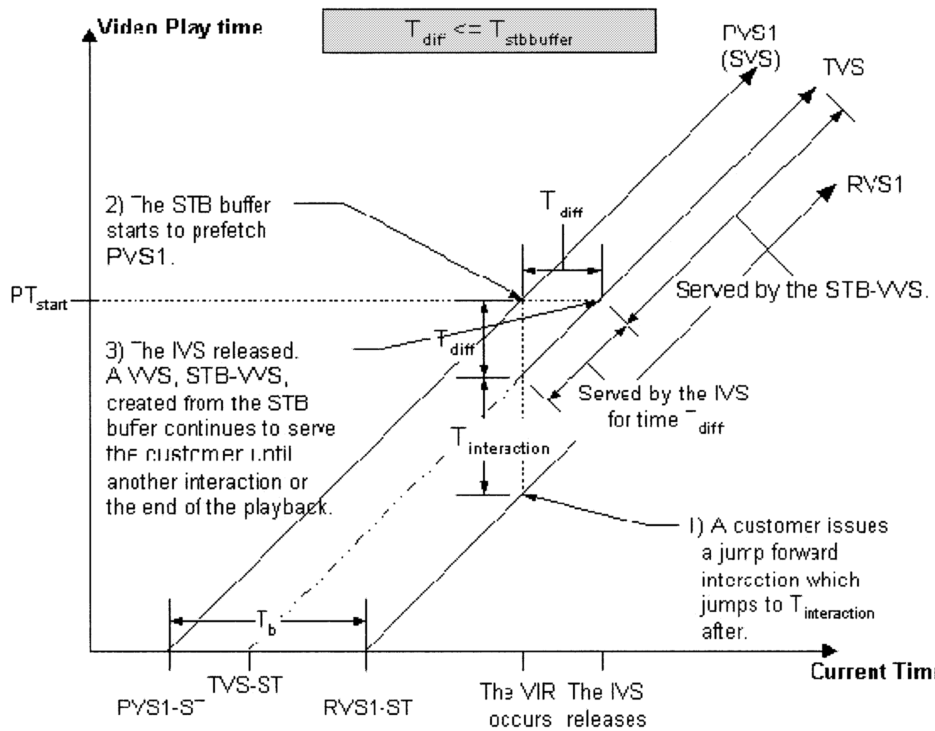


Fig. 6. Case i) A handling of a jump forward interaction.

Case I. $T_{diff} <= T_{stbbuffer}$ *and* $T_{stbbuffer} = Tb/2$: As shown in Fig. 6, the customer served by RVSI issues a jump forward VIR that can be handled by the TVS starting at TVS-ST. However, the probability of the existence of this VS is very low, thus the majority of the VIRs cannot be handled in this way.

If such VS does not exist, then the system tries to handle the VIR by a VVS, and as stated in the condition, the SVS was found and started not greater than $T_b/2$ from the start time of TVs, TVS-ST. As the time difference is not greater than the size of the buffer of the STB, a VVS called STB-VVS can be created from the STB according to above buffering mechanism.

Since during the prefetching stage, the STB-VVS cannot provide the required video frames to the customer, the system tries to allocate another VS from the video server called the Interaction VS or IVs to serve the customer during this period. After the prefetching period, the IVs will be released and the customer will be served by the STB-VVS until the customer issues another VIR or reaches the end of the playback. If no IVs is available in the system at that moment, the VIR is handled according to case iii.

Case II. $T_{stbbuffer} < T_{diff} <= T_b$ *and* $T_{stbbuffer} = T_b/2$: In case ii, where the start time of PVS (PVS1-ST) is started earlier than TVS (TVS-ST) is greater than the size of STB buffer. And using only STB buffer cannot serve the user. An extended method of case i is used in here.

When the user issued the VIR and requesting a VS with start time at TVS-ST to serve him, an IVS (IVS$_{stb}$) is allocated by the system to serve him for $T_{vvsdiff}$. At the same time, a VVS is created from prefetching the PVSI and an IVS (IVS$_{vvs}$) is used to support this VVS during the prefetching time. The STB buffer prefetches VS from this VVS.

After time $T_{vvsdiff}$, when the STB buffer has prefetched enough data and the IVS$_{stb}$ can be released. And after time $T_{diff} - T$, the IVS$_{vvs}$ can be released also. At this stage, the interaction handling is completed.

The main difference in the handling of the VIR compared with case i is that a VVS is created from prefetching a PVS of the VIR in the local server (SVS-L), and uses this VVS to act as the SVS of the STB-VVS created from the STB buffer. The handling mechanism of this case is similar with case i. And as the VVS is a multicast VS, it can be the SVS of more than one STB buffer or of the local server buffer. The difference between SVS and SVS-L is that the SVS-L is used as an SVS for the local server (rather than the STB).

Another difference compared with the previous case is that in case ii the system requires two IVSs: one serves the customer which holds for time $T_{vvsdiff}$, and another one serves the VVS which holds for time $T_b/2$. The reasons for allocating an extra IVS serving the VVS although no customer is served directly in VVS are: 1) the IVS serving the customer can be released earlier, and 2) the VVS can handle other VIRs during the prefetch stage of the local server buffer unit. The advantage compared with using only one IVS is that the IVS holding time can be shortened if another VIR requires the VVS during the prefetch stage of the buffer unit to be the SVS of another VVS or the STB buffer.

Slotted start time ensures that the time difference between the RVSs is multiples of T_b. In addition, the logical start time of VVS is $T_b/2$ after the logical start time of the SVS. The separation between VSs (RVSs and VVSs) is uniform and multiples of $T_b/2$. This increases the chance of finding a SVS for handling the VIRs. In particular, as shown in the simulation result (Section V-C-1), the MBB system has a higher availability of SVS compared to SAM system.

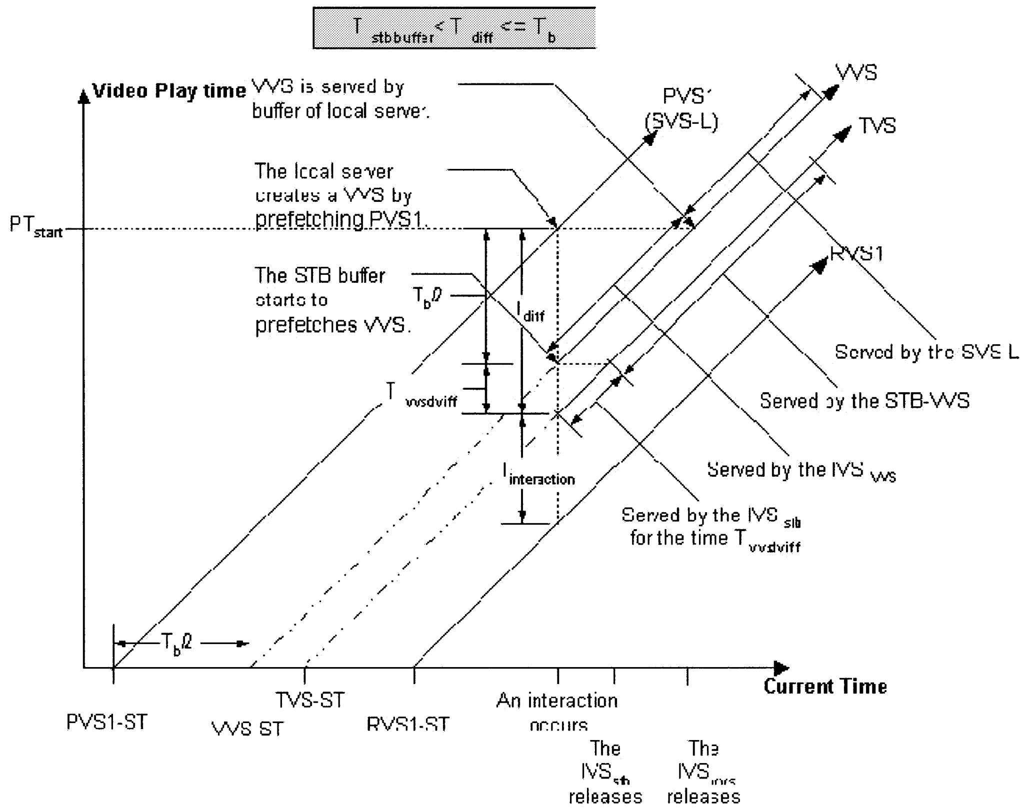


Fig. 7. Case ii) A handling of a jump forward interaction.

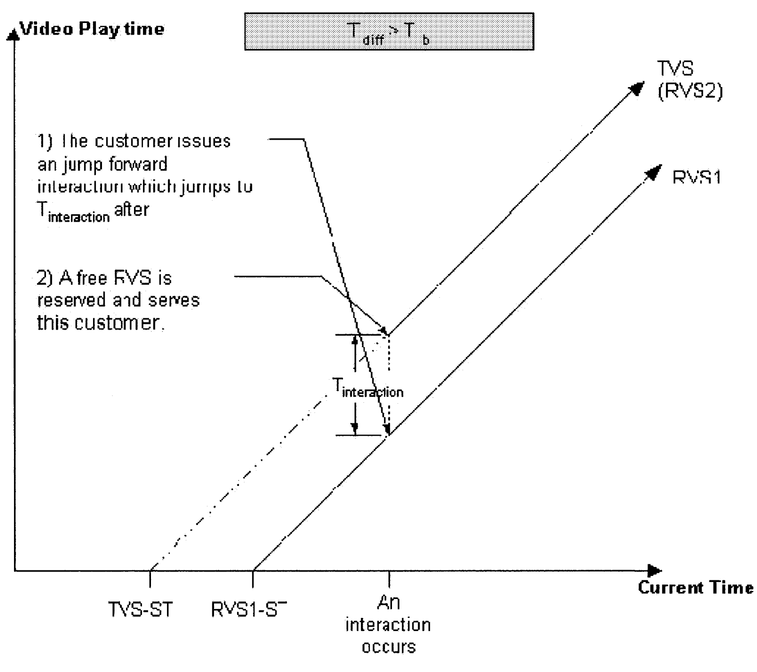


Fig. 8. Case iii) A handling of a jump forward interaction.

If the system fails to find two IVSs before the start of the prefetching in the buffers or there is no free VVS in the local server buffer, the VIR is handled according to case iii.

Case III. $T_{diff} > T_b$: If the above two cases do not hold or because lack of resources, the VIR is handled by case III. In this case, the VIR is handled by a free RVS, which is allocated in the video server and its start time equals to the TVS-ST. And only a RVS is required to handle the VIR. If no RVS is available, then the request is blocked.

C. The Handling Mechanism of Other Types of VIRs

For other types of interactions, the difference in how they are handled compared with the jump forward interaction is on the way the corresponding TVS-ST is computed. For example, for jump rewind interaction, the TVS-ST equals to the start time of the original VS plus the jump backward time. In addition, there is a minor difference on the calculation of TVS-ST for fast forward/rewind interaction because for these interactions intermediate frames are shown during the holding period of the buttons. Thus, the TVS-ST is known after the user releases the button, while the intermediate frames are supplied by an IVS. After the user releases the button, the calculation of the TVS-ST is the same as that of the jump forward/rewind interactions. After the system knows the TVS-ST, it handles all of the VIRs by the same method used in handling the fast forward interaction.

D. Resources Reclamation

The resources used during the handling of the VIRs are reclaimed after the customers release the VSs, which occurs when the customers issue another VIR or the video reaches the end of the playback. In brief, the MBB system checks whether the released VS can be free or not. If it can, the VS and its associated resources (for example, the buffer space used), if there is any, are also freed. Then, the system takes the SVS of the released VS as input and repeats the same procedure again, until the SVS of the released VS does not exist, which implies the VS is a RVS.

E. Variations of the MBB System

The mechanism described above is the general framework of the MBB system. The following points describe some further enhancements of this system.

The VVSs created from the local server, besides serving VIRs, can serve VSRs also. When a new VSR arrives, if there is a VS (RVS or VVS) that started no more than $\boldsymbol{T_b}$ earlier and delivering the same video requested, then the VS is used as SVS, and a VVS is created from the local server and begins to prefetch at $\boldsymbol{T_b}$ after the start time of the VS. At that moment an IVS is also created to serve the customers during the prefetch stage, which lasts for time $\boldsymbol{T_b}$. After time $\boldsymbol{T_b}$, the buffer has prefetched enough frames and the IVS is released, and the VVS serves the customers until the end of the playback. The procedure can be cascaded because the VVS, which uses a RVS as the SVS in handling VSR, can be used like SVS in handling another VSR that arrives at a later time but no more than T_b. The whole procedure is referred to as the improved VSR handling mechanism.

In Section III-B, the size of the buffer unit of the local server and the size of buffer of the STB is configured as $T_b/2$. The sizes of the buffers can affect the system performance. If the sizes of both buffers are enlarged to T_b, the number of the VVSs required in the VIRs handling are decreased, but the number of the IVSs used is increased. If the sizes of both buffers are reduced to, for example $T_b/3$, then the IVS holding time is reduced, at the expense of increasing the number of the VVSs required in the buffer of the local server.

For the less popular videos, only a smaller number of customers is watching them, and it might be better to serve these VSRs immediately by RVSs. In this way, the startup delay can be eliminated, and the same VS, instead of extra resources, can serve the VIRs of these customers. Overall, the system can save the resources used in the handling of the VIRs and reduce the VIR blocking probability due to the lack of resources.

In the simulation section, an improved version of the MBB system called the MI system is examined. The difference compared with the MBB system is that in the MI system 1) the improved VSR handling is used; and 2) The size of buffer unit of local server and the size of buffer of STB is changed to T_b. As shown in Section V, the MI system uses fewer resources than the MBB system.

IV. Comparisons of the MBB System With Other Systems

A. Comparisons Between the MBB System With the SAM System

The difference between the proposed MBB system and the SAM system in handling VSRs is that the MBB system ensures that the time separation between PVSs is an integral multiple of the batch time. As will be explained in Section V, this can increase the probability of finding an SVS when handling VIRs. However, the side effect is to shorten the batch time and this reduces the number of requests that can be served per batch. This is shown in detail in Section V.

The MBB system handles the VIRs by two levels of buffers which aim to solve the high workload problem of the local server buffer in the SAM system. In the MBB system the local server buffer will not degrade to the same scenario as in unicast systems because at most, ***video length of a video/$\boldsymbol{T_b}$***, number of VVSs are required to handle all the VIRs requesting the video.

Although the MBB system requires additional buffer space in the STBs, that buffer is required anyway to ensure the smoothness of play back and the workload of the STB in the MBB system is low because the buffer of the STB needs to serve only one customer. Thus, the additional cost is low.

B. Comparison Between the MBB System and the STB System

The STB system uses the same VSR handling mechanism of the SAM system. Thus, the comparison of MBB system with STB system on VSR handling is similar to that with SAM system and is not repeated in this section. The main difference between the MBB system and the STB system in handling VIR is that the MBB system uses an extra centralized buffer located at the local server to assist in handling the interaction requests. The advantages are:

1) Reduces the size of buffer of STB and the total buffer size required. In an STB system the size of the buffer of STB is T_b, while in MBB system the size is a fraction of T_b. In the above examples, the size of the buffer of STB in an MBB system is set to $T_b/2$. While an extra buffer in local server is required in MBB system, the required size of the extra buffer is not great and is shared by a large number of customers. According to the simulation results, the total size of buffer used in an MBB system is lower than in an STB system.
2) Decrease the holding time of IVS. The holding time of IVS is the same as the time difference between the SVS and the required VS. The VVSs in our system decreases the time separation between them. Hence, the holding time of IVS is reduced. As our simulation results indicate, the requirement of IVS is linearly proportional

to arrival rate. And the unavailability of IVSs means a RVS is needed to handle the request. Decreasing the holding time of IVS helps reduce the interaction blocking probabilities.

V. Simulation Results

A simulation program written in C++ and based on Sim++ [9] is used to model the customer behaviors and the working mechanisms of the VoD systems, and to observe the different system performance indicators to compare their capabilities under different areas. The video population distribution and the customer interaction model used in the simulation experiments are similar to the ones presented in [2]. The simulation experiments compare the MBB and the MI system (a variation of the MBB system described in Section III-E) with the SAM and the STB system as a function of the VSR and VIR blocking probabilities, the reasons of those blocking, resources usage like the number of RVSs, IVSs and VVSs used, and the space used in the buffer of the local server and the STB. Based on these results, the system that uses the minimum resources and achieves certain blocking probabilities requirements can be found.

A. Simulation Parameters

The parameters used in the experiments affect the performance of the systems. The main parameters are described below:

The Video Population Distribution: The video population distribution used in the simulation experiments follows the Zipf distribution, which is commonly used in related papers, and can model real scenarios accurately well [2], [5], [7]. The formula for generating the Zipf distribution in the simulator is $S * log(i)$, where i is a random variable and $0 < i < 1$, and S represents the skewing factor of the Zipf distribution. In a real environment, the value of S is unknown and varies from time to time. If 80/20 rule is used (this means 80% of customers select the top 20% of the videos), then the value of S is around 4 to 5. In the simulation experiments, the tested values of S are 1, 4, and 10, and this can represent the high (100/20), medium (80/20) and low (50/20) video probabilities distribution respectively.

The Video Interaction Modeling: The customers interactions behaviors are difficult to predict, therefore a complete model is difficult to construct. A number of papers such as [2] choose the following model: After a video started playing, the customer watches the video for an exponential amount of time with mean m, then the customer performs either one of the allowed interaction requests with probability p or continues to watch the video with probability $1 - p$. The length of interactions follows a certain distribution (e.g., uniform distribution). After the interaction is completed, the same procedure repeats until the end of video. In the simulation experiments of [2] and this paper, m is set to 30 minutes, p is set to 0.75, and the length of interactions (jump forward and jump backward) is 1 to 1000 seconds which follows a uniform distribution.

The Number of RVSs and IVSs: As indicated in Section V-B, the simulator tests the performance of the system with different combinations of RVSs and IVSs. The ranges of RVS and IVS allowed are 50 to 300 and 25 to 200 respectively.

The VSR Arrival Rate: The VSR arrives in a uniform distribution, and VSR arrival rates used in the experiments are 60, 80, 120, 180, 360 and 1200 req/hr. These arrival rates are chosen to represent the low, medium, or high load situations. Note that in practice the VSR arrival does not follows a uniform distribution, but the effects of different distributions on the testing systems that serve VSRs using batching approach should be more or less the same as the one of uniform distribution.

The Length of Videos and the Simulation Time: The length of all videos is two hours, which is a standard length of movies. The simulation experiments are run for four hours and this represents the peak hours from 19:00 to 23:00.

The Batch Time: In the experiments, the batch time is set to 10 minutes in all the systems.

The buffers of local servers and STBs, and the number of VVSs in local servers need also to be 15 taken into consideration.

In the simulations, for the SAM, MBB and MI system, the buffers of the local servers can cache 50 hours of video frames which corresponds to 33 GB if the videos are encoded with MPEG 1 (1.5 Mbps) quality, the maximum number of the video streams delivered by the local servers' buffer is 300, and the buffer unit can hold 5 and 10 minutes of video frames in the MBB system, and the SAM, STB and MI systems respectively. For the STB buffer, it can hold 5 minutes and 10 minutes in the MBB system, and the MI and STB systems, respectively.

B. Simulation Procedures

The focus of the simulation analysis in this paper is on finding out the minimum cost or the minimum required resources in each of the systems to serve a given number of customers and a certain video distribution under the same QoS requirements. To achieve this, the following procedure is used:

For each system, different configurations, which are combinations of the basic resources used in the systems such as RVS, IVS, VVS and STB-VVS, are examined. Then, the actual usage of these resources and the QoS measurements like the VSR blocking probabilities and VIR blocking probabilities are measured in each configuration under each of the tested arrival rates and video distributions. For each of the tested arrival rates and video distributions, the configuration reaching the QoS requirement and having the lowest resources requirement, or the lowest cost, in each of the systems is selected to represent the performance of this system in this case. The system performances of the tested arrival rates and video distributions are plotted in charts.

As different configurations have different amounts of resources used and the cost of each resource is different, computing the exact total resource used or the total cost of each configuration is difficult. Since finding this cost is difficult, an alterative approach is to find the cost relationship between them and to compare the total cost of each configuration by converting them to the same unit. To achieve this, the cost per

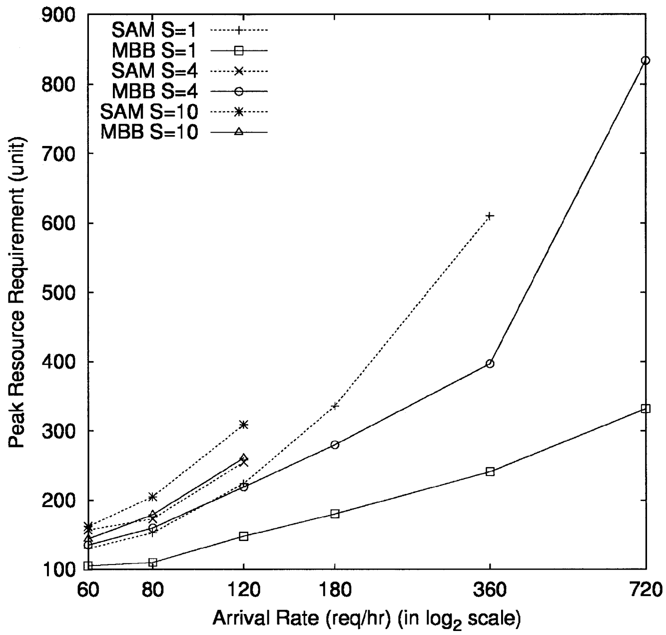


Fig. 9. SAM vs. MBB system, 0% VSRs and VIRs blocking probabilities.

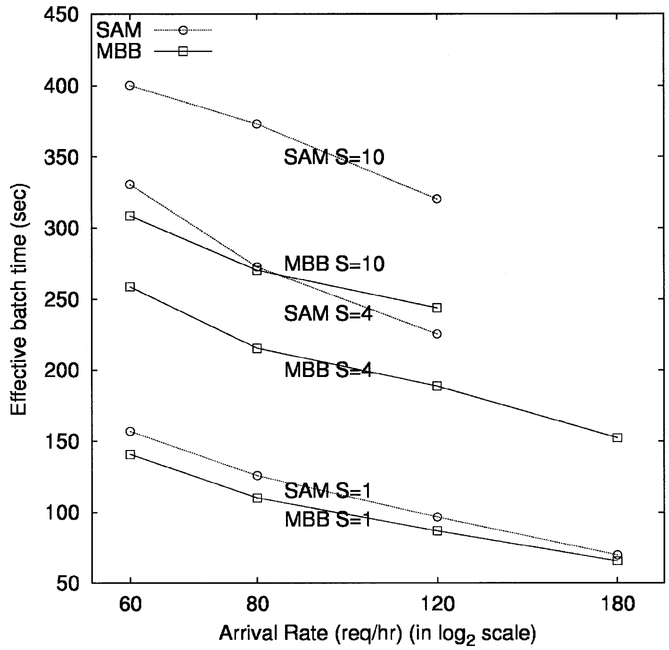


Fig. 10. SAM vs MBB system, Effective Batch time.

RVS is selected as the unit and the following approximations are used:

1) The cost per RVS is approximately the same as IVS.
2) The cost per RVS is approximately the same as VVS.
3) The cost per STB-VVS is negligible.
4) The cost per RVS, IVS, VVS, STB-VVS is constant.

In practice, only approximation 1 is more likely to hold. But the other approximations are good for a first degree approximation.

As limited by space, only the simulation results that can result with 0% blocking probabilities of VSRs and VIRs are listed for comparison. Simulations have been carried to allow 1% and 5% of VSRs and VIRs. The difference between the compared systems is similar to the case with the 0% blocking probabilities of VSRs and VIRs.

C. Simulation Analyses

1) The SAM System vs. the MBB System: Fig. 9 shows the minimum resource requirements of the SAM and the MBB systems with no blocking in handling VSRs and VIRs under different arrival rates and video distributions. It is clearly shown that the MBB system requires lower resources in all the cases. Because of limited space, only the total resource used is shown in the figures.

Fig. 10 shows the effective batch time of both systems. The effective batch time equals the average waiting time of the customers. From the system point of view, the longer the effective batch time is, the more customers can be grouped per RVS. Fig. 11 proves this is correct. Both figures imply that the MBB system requires more resources in handling VSRs than the SAM system. However, as shown in Fig. 12 the way in which the VIRs are handled affects the overall system performance.

According to Fig. 12, which lists out how VIRs are handled in both systems, the MBB system handles more VIRs by VVSs than the SAM system. As a single VVS can serve many customers, this implies fewer resources are used in the MBB system. Due to limited space, the figure lists the comparison under a single arrival rate only, but similar result holds in other arrival rates.

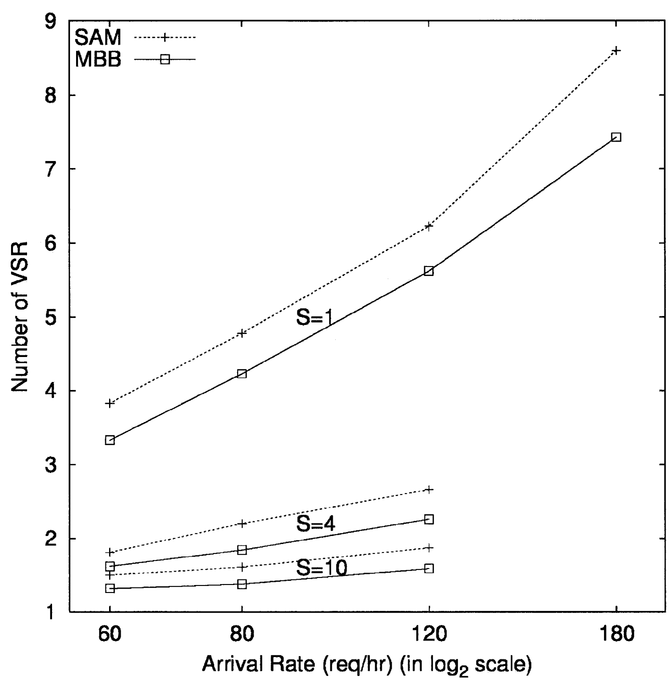


Fig. 11. SAM vs MBB system, Average number of VSRs handled by a RVS.

The reasons for why there are more VIRs being handled by RVSs in the SAM system are shown in Fig. 13. The figure shows that in the SAM system a number of VIRs not able to allocate VVSs, while given the same size in the local server buffer, for the MBB system all of the VIRs can get VVSs if required. In addition, for the SAM system there is a higher chance that the SVS is not available. Note that the higher availability of the SVS

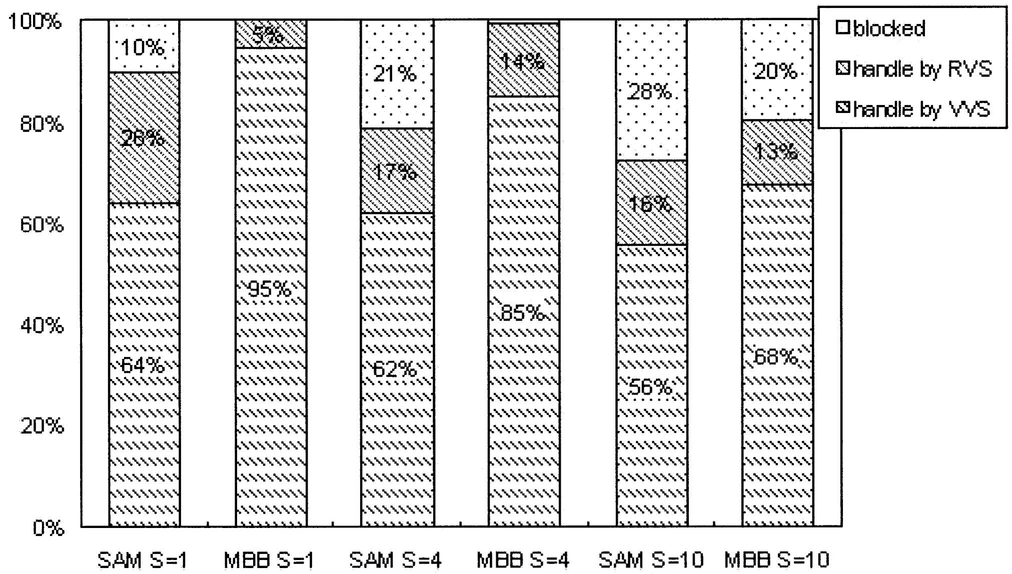


Fig. 12. Interaction handling of SAM and MBB system at arrival rate 360 request/hour.

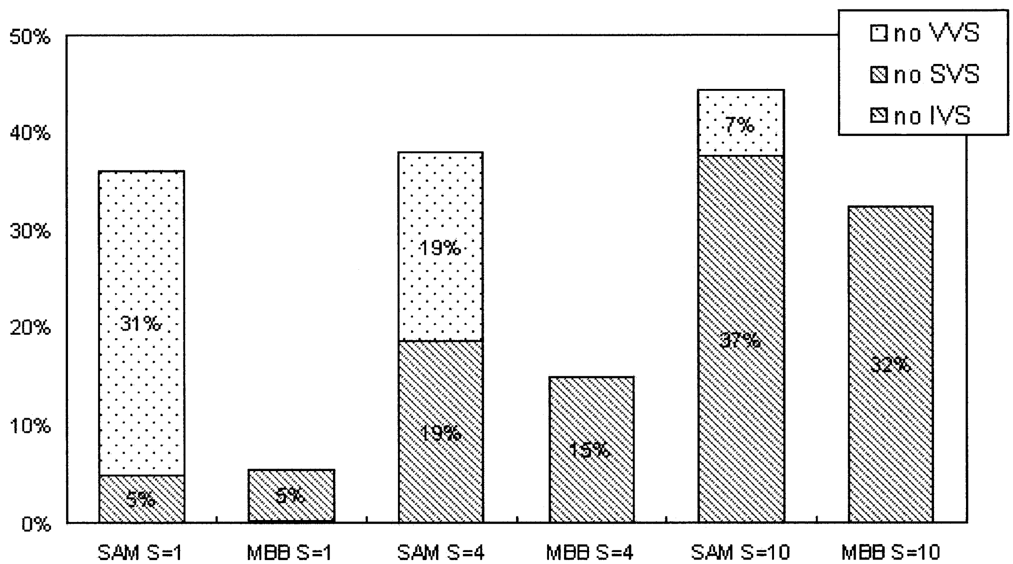


Fig. 13. The reasons why the VIRs are not handled by VVS in the SAM and MBB system at arrival rate 360 request/hr.

in the MBB system is not due to the higher number of VSs in the system. In fact, it can be shown that the number of RVSs and VVSs used in the MBB system is lower than in the SAM system. The higher availability is due to the slotted start time allocation scheme.

One of the differences between the architecture of the STB system and the MBB system is the size of the STB buffer. In the STB system the size is doubled compared with that of the MBB system. To have a more fair comparison and to further enhance the MBB system performance, the variation of the MBB system described in Section III-E called MBB-Improved or MI system is used for the comparison with the STB system.

Fig. 14 shows that the MI system has lower resource requirement compared to the MBB system. This is because some VVSs can serve VSRs and VIRs. Although the VVSs are used to serve VSRs as well as VIRs, the buffer of local server and the STBs are enlarged so that fewer numbers of VVSs are required. Fig. 15 shows that a larger number of VSRs can be served by VVSs instead of RVSs. If the cost per VVS is cheaper than that per RVS, then the required cost of the MI system is lower than that of the MBB system.

Fig. 16 shows the minimum resource requirement of the STB system and the MI system with no blocking.

At a first glance according to Fig. 16, the STB system requires fewer resources than the MI or the MBB system. However, the following points should be noticed:

The total resources used is an estimated value since the cost of the VVS is believed to be lower than that of the RVS. Thus, the approximations are not accurate enough.

The main difference between the MI and STB system is in the use of the VVSs in the handling of the VSRs and VIRs in the MI system. Since in the MI system the VVSs can be used to handle

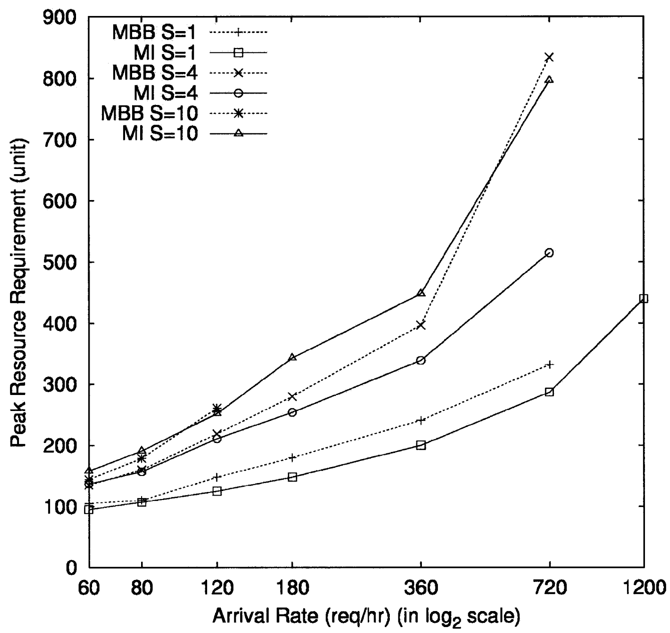


Fig. 14. MBB vs. MI system 0% VSR and VIR blocking probabilities.

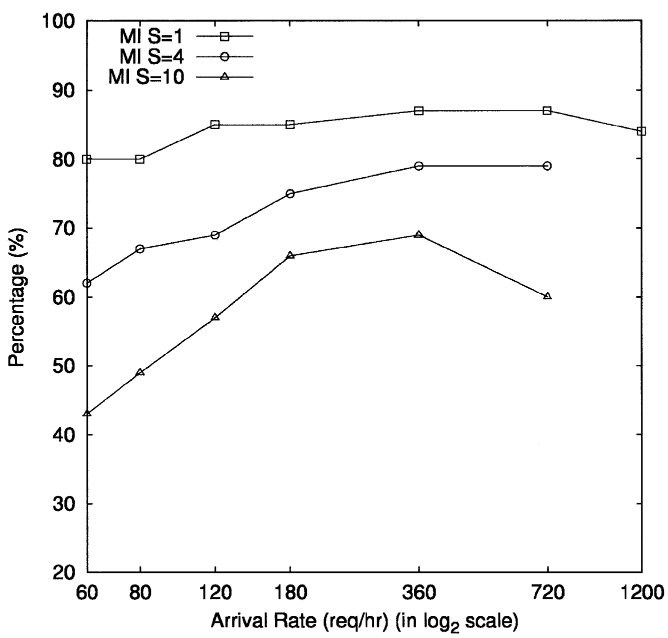


Fig. 15. Percentage of VSRs handled by VVSs in MI system with zero blocking probabilities.

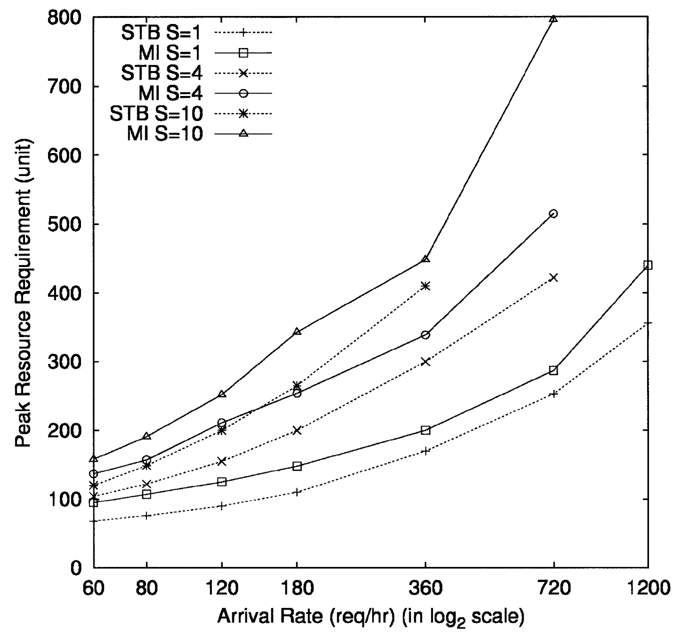


Fig. 16. STB vs MI system, 0% VSRs and VIRs blocking probabilities.

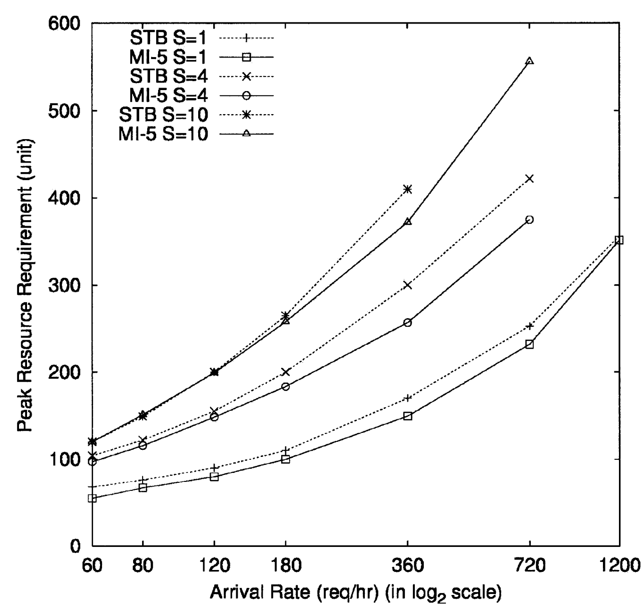


Fig. 17. STB vs MI-5 system, 0% VSRs and VIRs blocking probabilities.

the VSRs and the VIRs only if both the SVS and a free IVS exist, but in the STB system the RVSs can be used to handle the VSRs and VIRs under any condition and without the existence of IVS. If the assumption, in which a RVS and a VVS equals to one resource unit holds, then for lower resource requirements RVSs are more favored than VVSs. If the assumption does not hold, as shown in Fig. 17, the resource requirement of the MI system can be lower than that of the STB system.

Since the VIRs can be handled by VVSs only if SVS is available, and there is at least one free VVS and IVS. A key indicator of whether the VIRs can be handled by VVSs or not depends on the availability of SVSs because this is not controlled directly by adding more resources. Table I shows that for an arrival rate of 360 req/hr and video distribution of $S = 4$, the percentage of the VIRs that fail to find the SVS is lower in the MI system than the STB system.

In Table I, although having a higher VSR and VIR blocking probabilities, the MI system still handles the VIRs better than the STB system. For example, a larger percentage of the VIRs are handled by the VVSs and fewer VIRs are handled by RVSs in the MI system. Although the MI system has higher blocking probabilities, the result still implies that the MI system handles the VIRs in a more scalable manner than the STB system.

As the cost relations between RVSs, VVSs and IVSs are not understood clearly, whether the cost of a RVS is higher than that of a VVS is unknown. In the following part, a new assumption

TABLE I

	Estimated total resources	RVS	VVS	IVS	Blocking probabilities of		% of VIRs handled by		No IVSs	No VVSs	No SVSs
					VSRs	VIRs	VVS	RVS			
STB	300	227	0	73	0%	0%	80%	20%	0%	0%	20%
MI	292	150	50	92	7%	5%	84%	11%	0%	7%	10%

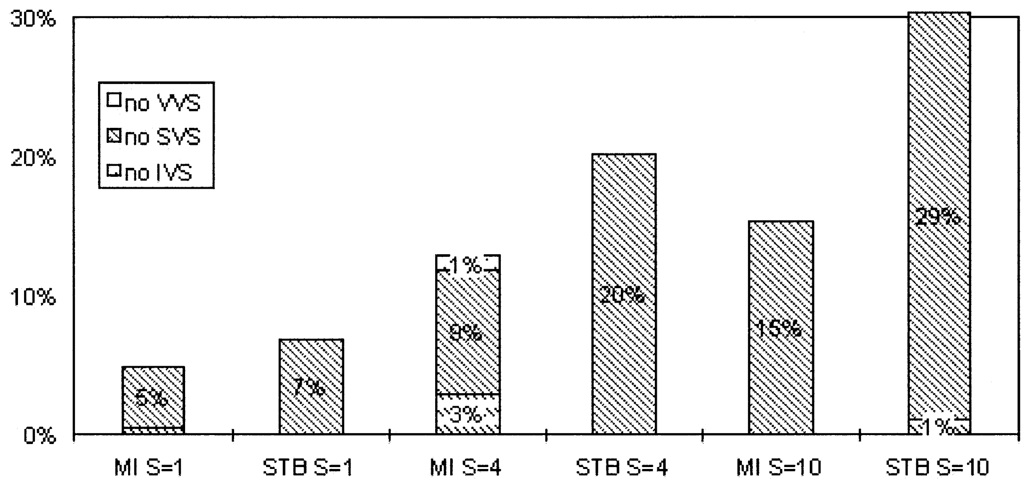


Fig. 18. The reasons why VIRs are handled by RVSs in the STB and MI-5 system at arrival rate of 360 req/hr.

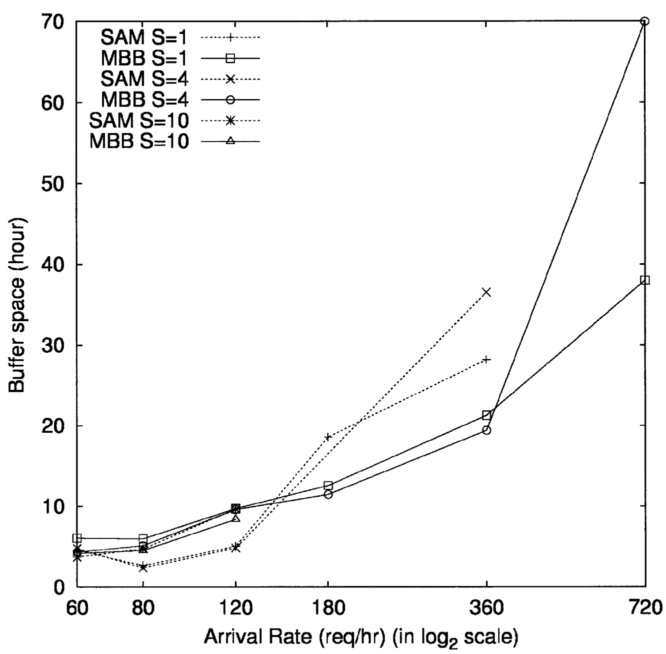


Fig. 19. SAM vs MBB system, Total buffer space used.

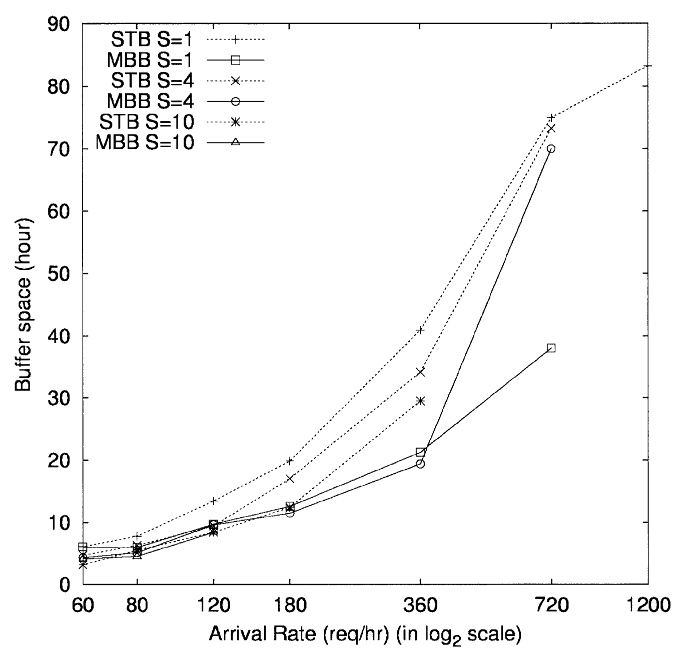


Fig. 20. STB vs MBB system, Total buffer space used.

is used—the cost of five VVSs equals to the cost of a RVS. This assumption is chosen because at this ratio the MI system has a lower resource requirement than the STB system in the tested arrival rates and video distributions. Notice that this assumption should be viewed like this: If the cost of five VVSs is lower than the cost of a RVS in a particular implementation of a VoD system, then the MI system should be more favorable than the STB system, and vice versa.

This new assumption affects only the calculation of the total resources in the SAM, MI and MBB systems but not the STB system, because the STB system uses no VVS. In the following, the MI-5 and SAM-5 system is used to evaluate the systems that use the new assumption in the resource calculation.

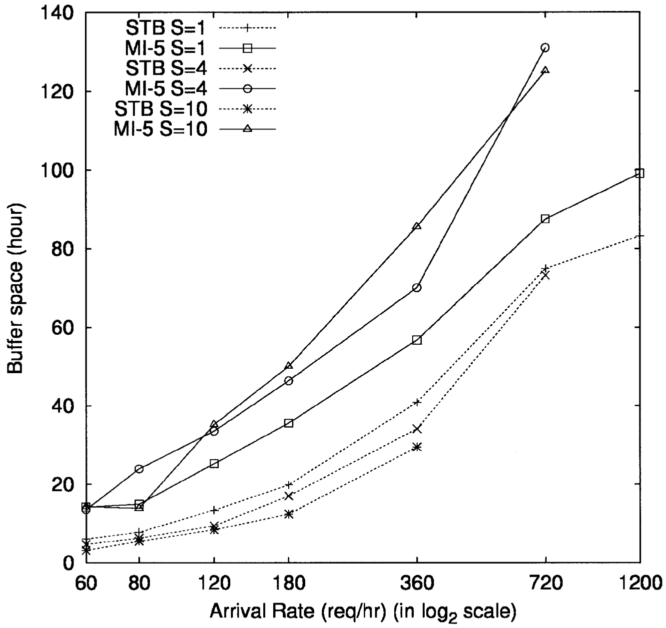


Fig. 21. STB vs MI-5 system, Total buffer space used.

Fig. 17 shows the lowest resource requirement of the STB system and the MI-5 system with zero VSR and VIR blocking probabilities. As is clearly shown the MI-5 system performs better in all the tested arrival rates and video distributions. Fig. 18 shows the reasons why VIRs are handled by RVSs in both systems.

D. Buffer Utilization

Fig. 19–Fig. 21 shows the total buffer space used in the local server and STB. For nearly all the cases, the MBB system requires lower buffer space than the SAM and STB system. Due to a larger number of VVSs used, the MI-5 system requires more buffer space than the STB system.

VI. Conclusion

In this paper, a scalable batching VoD system with a low per-user cost called the MBB system is proposed. The MBB architecture attempts to solve the system degradation problem that occurs during the handling of interactions in unicast systems and other proposed batching VoD systems. In the simulation experiments, the effects of arrival rates, video population distributions and different combinations of system resources on the performance of the SAM, STB and MBB systems are investigated. In particular, their effects on the VSR and VIR blocking probabilities, start-up delays of customers, number of VSRs grouped per RVS, required size of the buffer of the local servers and of the STBs, and the minimum resource usages to achieve zero blocking probabilities, of the tested systems are analyzed. The simulation experiments demonstrate that the performance of the MBB system is better than that of the SAM system, and this enhancement is mainly due to the help of the STBs buffer in the MBB system. Comparing the performance of the MBB system and MI system with that of the STB system, we find that the higher probabilities in finding the SVSs and the handling of the VSRs by the VVSs instead of the RVSs in the MBB and MI system can give tremendous benefits to the system.

References

[1] *Interactive TV (ITV)*. Hong Kong: CWHKT.

[2] L. Wanjiun and V. O. K. Li, “The split and merge protocol for interactive video-on-demand,” *IEEE Multimedia*, vol. 4, no. 4, pp. 51–62, Oct.–Dec. 1997.

[3] S. A. Barnett and G. J. Anido, “A cost comparison of distributed and centralized approaches to video-on-demand,” *IEEE J. Select. Areas Commun.*, vol. 14, no. 6, pp. 1173–83, Aug. 1996.

[4] F. Zongming, I. Kamel, S. Mukherjee, and M. H. Ammar, “Providing interactive functions for staggered multicast near video-on-demand systems,” in *Proc. IEEE Int. Conf. Multimedia Computing and Systems*, 1999, pp. 949–53.

[5] E. L. Abram-Profeta and K. G. Shin, “Providing unrestricted VCR functions in multicast video-on-demand servers,” in *Proc. IEEE Int. Conf. Multimedia Computing and Systems*, 1998, pp. 66–75.

[6] S.-H. G. Chan, F. Tobagi, and K. Tsz-Mei, “Providing on-demand video services using request batching,” in *Proc. IEEE Int. Conf. Communications*, 1998, pp. 1716–22.

[7] K. C. Almeroth and M. H. Ammar, “The use of multicast delivery to provide a scalable and interactive video-on-demand service,” *IEEE J. Select. Areas Commun.*, vol. 14, no. 6, pp. 1110–22, August 1996.

[8] S.-H. Chan and F. Tobagi, “Providing distributed on-demand video services using multicasting and local caching,” in *Proc. IEEE Multimedia Applications, Services, and Technologies*, June 7, 1999.

[9] Simpack. [Online]. Available: http://www.cise.ufl.edu/~fishwick/simpack/simpack.html.

[10] F. Zongming, A. Ammar, I. Kamel, and S. Mukherjee, “Providing interactive functions through active client buffer management in partitioned video broadcast,” in *Proc. 1999 Workshop on Networked Group Communication*, ser. Lecture Notes in Computer Science: Springer Verlag, Nov. 1999, vol. 1736, pp. 152–169.

[11] L.-S. Juhn and L.-M. Tseng, “Adaptive fast data broadcasting scheme for video-on-demand service,” *IEEE Trans. Broadcast.*, vol. 44, no. 2, pp. 182–185, June 1998.

[12] M. Y. Y. Leung, J. C. S. Lui, and L. Golubchik, “Buffer and I/O resource pre-allocation for implementing batching and buffering techniques for Video-on-Demand systems,” in *Proc.13th Int. Conf. Data Engineering*, 1997, pp. 344–53.

[13] G. Kerr, “BT’s video on demand trials,” in *Proc. 2nd Int. Workshop on Community Networking Integrated Multimedia Services in the Home*, 1995, pp. 199–204.

[14] S.-H. G. Chan and F. A. Tobagi, “Caching schemes for distributed video services,” in *1999 IEEE Int. Conf. Communications*, 1999, pp. 994–998.

[15] A. Dan, D. M. Dias, R. Mukherjee, D. Sitaram, and R. Tewari, “Buffering and caching in large-scale video servers,” in *Digest of Papers of COMPCON ’95*, 1995, pp. 217–24.

[16] K. C. Almeroth, A. Dan, D. Sitaram, and W. H. Tetzlaff, “Long term resource allocation in video delivery systems,” in *Proc. IEEE INFOCOM ’97*, 1997, pp. 1333–40.

[17] S. Viswanathan and T. Imielinski, “Metropolitan area video-on-demand service using pyramid broadcasting,” in *Multimedia Syst.*, Germany: Springer-Verlag, Aug. 1996, vol. 4, no. 4, pp. 197–208.

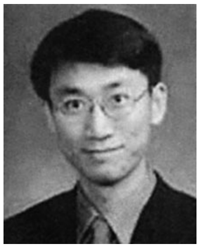
[18] W. J. Bolosky, J. S. Barrera, R. P. Draves, R. P. Fitzgerald, G. A. Gibson, M. B. Jones, S. P. Levi, N. P. Myhrvold, and R. F. Rashid, “The tiger video fileserver,” in *Proc. Int. Workshop on Network and Operation System Support*, April 1996.

[19] A. Dan, D. Sitaram, and P. Shahabuddin, “Dynamic batching policies for an on-demand video server,” *Multimedia Syst.*, vol. 4, no. 3, pp. 112–121, June 1996.

[20] T. Little and D. Venkatesh, “Prospects for interactive video-on-demand,” *IEEE Multimedia Mag.*, pp. 14–24, Fall 1994.

[21] F. A. Tobagi, “Distance learning with digital video,” *IEEE Multimedia Mag.*, pp. 90–94, Spring 1995.

[22] J. Sutherland and L. Litteral, “Residential video services,” *IEEE Commun. Mag.*, pp. 36–41, 1992.



Cyrus C. Y. Choi is a Senior System Application Analyst of Interactive Communication Online Networks Ltd (ICON). His research interests include E-Learning Repository system, XML applications, and multimedia and network application developments. He received the BEng and MPhil from Hong Kong University of Science and Technology in 1998 and 2001, respectively.

Mounir Hamdi received the B.S. degree in Computer Engineering (with distinction) from the University of Louisiana in 1985, and the MS and PhD degrees in Electrical Engineering from the University of Pittsburgh in 1987 and 1991, respectively. He has been a faculty member in the Department of Computer Science at the Hong Kong University of Science and Technology since 1991 where he is now Associate Professor of Computer Science and the Director of the Computer Engineering Programme. In 1999 and 2000 he held visiting professor positions at Stanford University and the Swiss Federal Institute of Technology.

His general areas of research are in high-speed packet switches/routers and all-optical networks, in which he has published more than 150 research publications, and for which he has served as consultant to various international companies. Currently, he is working on high-speed networks including the design, analysis, scheduling, and management of high-speed switches/routers, wavelength division multiplexing (WDM) networks/switches, and wireless networks. He is leading a team that designed one the highest capacity chip sets for Terabit switches.

Dr. Hamdi is/was on the Editorial Board of IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE Communication Magazine, Computer Networks, Wireless Communications and Mobile Computing, and Parallel Computing, and has been on the program committees of more than 50 international conferences and workshops. He was a guest editor of IEEE Communications Magazine, IEEE Journal on Selected Areas of Communications, and Optical Networks Magazine, and has chaired more than 5 international conferences and workshops. He received the best paper award at the International Conference on Information and Networking in 1998 out of 152 papers. He received the best 10 lecturers award and the distinguished teaching award from the Hong Kong University of Science and Technology. He is a member of IEEE and ACM.