SOFTWARE—PRACTICE AND EXPERIENCE
Softw. Pract. Exper. 2004; **34**:1187–1210 (DOI: 10.1002/spe.609)

Performance evaluation for VBR Continuous Media File Server admission control

D. J. Makaroff[1,*,†], G. W. Neufeld[2] and N. C. Hutchinson[2]

[1]*Department of Computer Science, University of Saskatchewan, Saskatoon, SK, Canada*
[2]*Department of Computer Science, University of British Columbia, Vancouver, BC, Canada*

SUMMARY

A file server for continuous media must provide resource guarantees and only admit requests that do not violate the resource availability. This paper addresses the admission performance of a server that explicitly considers the variable bit rate nature of the continuous media streams. A prototype version of the server has been implemented and evaluated in several heterogeneous environments. The two system resources for which admission control is evaluated are the disk bandwidth and the network bandwidth. Performance results from both measurement and simulation are shown with respect to different admission methods and varying scenarios of stream delivery patterns. We show that the *vbrSim* algorithm developed specifically for the server outperforms the other options for disk admission especially with request patterns that have staggered arrivals, while the network admission control algorithm is able to utilize a large percentage of the network bandwidth available. We also show the interactions between the limits of these two resources and how a system can be configured without wasted capacity on either one of the resources. Copyright © 2004 John Wiley & Sons, Ltd.

KEY WORDS: multimedia; file servers; variable bit rate; admission control; performance evaluation

INTRODUCTION

Continuous media file servers require that several system resources be reserved in order to guarantee timely delivery of the data to end-user clients. These resources include disk, network, and processor bandwidth. In a heterogeneous system accommodating variable bit-rate data streams, the amount of each resource differs for each stream and varies over time. A key component of determining the amount

*Correspondence to: Dwight J. Makaroff, Department of Computer Science, University of Saskatchewan, Saskatoon, SK, Canada S7N 5AN.
†E-mail: makaroff@cs.usask.ca

Contract/grant sponsor: Canadian Institute for Telecommunications Research (CITR)

Received 18 July 2001
Revised 27 January 2004
Accepted 27 January 2004

Published online 12 July 2004
Copyright © 2004 John Wiley & Sons, Ltd.

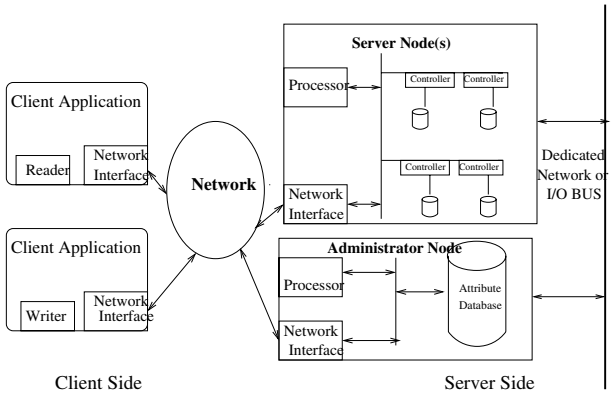

Figure 1. Organization of system.

of a resource to reserve is characterizing each stream's bandwidth requirements. Admission control is necessary to ensure adequate resources exist to sustain delivery for the duration of the playback. The two resources that we consider managing are the disk bandwidth and the server network interface bandwidth.

This paper presents the results of extensive performance evaluation of both the disk and the network admission control algorithms of the UBC Continuous Media File Server (CMFS) on a representative set of variable bit rate (VBR) video objects. We were able to verify that the algorithms used in the CMFS make use of a large proportion of system resources and still provide a deterministic guarantee that all data will be transmitted from the server as promised.

SYSTEM MODEL AND TEST ENVIRONMENT

System architecture

The design of the file server is based on an *administrator node* and a set of *server nodes*, each with a processor and disk storage on multiple local I/O buses. Each node is connected to a high-performance network for delivering continuous media data to the client systems (see Figure 1). This architecture can also consider each node as an 'edge server', or local cache server [1]. Federations of servers can serve a large geographic area and/or user population.

A server can be made up of different types of server nodes, ranging from powerful computers with a RAID configuration to smaller computers with fewer disks or even simple processor cards interconnected via an I/O bus such as PCI. A similar architecture is used in other scalable, high-performance video servers [2,3].

Most continuous media servers divide time into intervals called *slots* or *rounds*, during which sufficient blocks of data are read off the disk and/or transmitted across the network for each active

Copyright © 2004 John Wiley & Sons, Ltd.

stream to allow continuous playback by the client application. A reasonable length for such a slot is 500 ms. In the CMFS, the only significant parameter in the disk subsystem is the number of I/O operations guaranteed per slot (hereafter referred to as *minRead*). This 'number of reads per slot' value is constant for each disk configuration, and is determined by running a calibration program to determine the largest number of blocks that can be guaranteed to be read off the disk, assuming the reads are from the worst possible disk block locations [4]. One approximation to the worst case involves reading a single block at the outside edge of the disk and the remainder of the blocks on inside tracks, with each read of one block requiring a seek. This pathological case accesses all but one track of the data on the innermost region of the disk, so we assumed a more likely worst case in our experiments, which was a maximal number of equidistant seeks covering the entire disk surface.

With respect to the network interface, a fixed maximum bandwidth exists. The number of blocks that can be transmitted from the server during a slot is defined to be *maxXmit*, and can be calculated in the same manner as the value of *minRead*. This value depends on the bandwidth of the interface card as well as the packet size chosen. We do not consider end-to-end network throughput, since the server cannot enforce delivery.

Stream descriptions

The audio and video streams stored on the server vary in a number of dimensions, including average bit rates, stream playback durations, and peak rate of data transmission. Typical environments for such video servers can range from, Movie-on-demand, where the average length of a stream is 100 min, to News-on-demand, where many streams may be quite short in duration (i.e. less than 1 min) and nearly all will be shorter than 10 or 15 min.

The streams chosen to test the admission algorithms of the CMFS are full-motion, medium-to-high quality VBR video streams with playback length between 1 and 10 min[‡]. A large number of reasonably short video streams were digitized and compressed using a Parallax MJPEG encoder/decoder card attached to a SUN Sparc 10. This card captured VHS video at 30 frames per second (fps) at 640×480 pixels. The bandwidth requirements of a representative selection of the streams are summarized in Table I. In the version of the server used for the performance experiments of this paper, blocks are 64 KB and a slot is 500 ms.

Approaches to systems performance issues

In previous work, many different approaches have been introduced to reduce the amount and the variability of resources required to support the retrieval and delivery of continuous media. Buffering at the server is used in combination with buffering at the client and playback startup latency for the convenience of the smoother network transmission [5]. These techniques result in more efficient use of resources as well as better service to the client. In this work, we focus on two main techniques: pre-fetching/pre-sending at the server with precise knowledge of the data blocks required and prediction of the disk and network bandwidth.

[‡]Client hardware limited the available formats.

Copyright © 2004 John Wiley & Sons, Ltd.

Table I. Stream characteristics.

				Bandwidth required (blocks/slot)				
Stream	Frames	FPS	Time (s)	Min	Max	Ave	Stdev	Cov
Annie Hall	4503	30	150.1	3	6	3.71	0.84	0.23
Aretha Franklin	12 535	30	417.8	2	7	3.71	0.68	0.19
Baseball	2570	30	85.7	1	7	3.65	1.10	0.30
Basketball	4072	30	135.7	3	13	6.69	2.23	0.35
Cartoon Trailers	1791	20	89.6	2	9	5.26	1.59	0.30
Evacuation	13 888	30	462.9	1	7	3.33	0.77	0.23
George of the Jungle	1192	20	59.6	2	9	5.95	1.32	0.22
Island of Whales	2798	20	139.9	1	5	2.86	0.78	0.27
John Elway	3117	30	103.9	3	10	5.93	1.77	0.30
Christian Laettner	9973	30	332.4	2	11	5.94	1.86	0.32
Maproom	10 843	30	361.4	1	9	4.24	1.46	0.34
Minnesota Twins	5476	30	182.5	3	14	6.03	2.11	0.35
Moody Blues	6565	30	218.8	2	5	2.96	0.59	0.19
Mr White	2086	20	104.3	1	3	2.16	0.51	0.24
NFL Football	13 332	30	444.4	2	11	5.6	1.64	0.29
Plan-9	3186	20	159.3	2	4	2.29	0.46	0.20
Ray Charles	8491	30	283.0	5	9	7.28	0.86	0.12

In order to efficiently provision the disk bandwidth, some measure of the current available bandwidth and a reasonable method of predicting the future requirements is necessary. This can be done either via precise schedules, or probabilistic modeling. In particular, we focus on the explicit relationship between the future disk requirements and the future disk performance. Both of these are highly variable. A VBR stream's bit-rate requirements can vary by an order of magnitude over short periods of time, while disk bandwidth depends on the number of seeks required as well as location of the blocks on the disk surface. If either of these measures is inaccurate, the system will perform poorly. The performance experiments will show that our mechanisms enable a high resource utilization without introducing viewing latency.

Test environment

The CMFS has been implemented on several hardware and software platforms. Most of these are UNIX-based workstation environments. In particular, versions of the server exist for IBM RS/6000 (AIX 3.2.5), SUN SPARC (Solaris 2.5), and Pentium-based PCs (Linux, FreeBSD, Solaris, and Windows NT). The measurement experiments were carried out on AIX and Intel-Solaris, with both raw disk interface and Asynchronous I/O for high disk bandwidth. The AIX environment consisted of a Model 250 (66 MHz CPU), and a 100 Mbps ATM card with a Taxi interface on a Newbridge Mainstreet 31650 ATM switch. A Pentium III 200 MHz CPU ran Intel-Solaris with 100 Mbps Ethernet adaptors. The disks used in the experiments were 2 GB Seagate Barracuda disks (model ST32550W) attached via SCSI 2 Fast-Wide adaptors.

Copyright © 2004 John Wiley & Sons, Ltd.

Table II. Admission algorithms comparison.

Algorithm	Block schedule	Computational complexity	Buffer requirement for smoothing	Guarantee
Simple maximum	no	O(1)	none	deterministic
Instantaneous maximum	yes	O(slots of stream + sequences)	none	deterministic
Average	no	O(1)	undetermined	statistical
vbrSim	yes	O(slots of longest stream + sequences)	yes	deterministic

DISK ADMISSION CONTROL ALGORITHMS

The algorithms for disk admission control ensure the server has enough bandwidth to read the data by the deadline for transmission. They can provide either deterministic guarantees or statistical guarantees. Providing a deterministic guarantee may be too conservative and admit too few streams. Statistical-guarantee admission policies can typically admit more streams, which may result in over-utilization, manifesting itself as delay or loss of data at the client.

We considered four distinct approaches to VBR disk admission. Each approach represents a class of admission approaches which provide generally similar results. A summary of their characteristics is given in Table II. The run-time behavior of the disk scheduling and delivery is the same in all of the algorithms. Disk blocks are read earliest deadline first, with multiple reads performed asynchronously and in parallel where possible. In modern I/O subsystems, the device controller abstracts aspects of the physical layout making low-level scheduling impractical. We assume a SCAN disk scheduling algorithm at the lowest layer and explicitly provide a large number of simultaneous requests for the device controller.

To understand the relevant differences between the disk admissions algorithms, three resources are measured: the CPU cycles used, the disk read bandwidth, and the number of buffers available. An accurate algorithm that cannot make a decision in a timely manner is not useful. The bandwidth measure has three components itself: the bandwidth guarantee, the bandwidth requested, and the bandwidth achieved for a particular experiment. An algorithm is considered to perform well if it can accept workloads with average requirements that exceed *minRead* and approach the achievable bandwidth. An algorithm which makes use of significant buffer space is more costly than one which does not and increases rejections of valid requests.

Whenever a client requests a portion of a media stream, a bandwidth characterization for the stream is made. It may be a single value or a number of parameters (mean bandwidth, standard deviation of bandwidth) or a detailed time-varying schedule. The most detailed characterization is the *stream block schedule*. This schedule must be created at delivery time, since different presentation units may be required, dependent on the prepare parameters [4].

A set of stream requests submitted to a CMFS as a unit is defined as a *scenario*. Scenarios may consist of simultaneous request arrivals or staggered arrivals, modeling a more common workload for a single disk in a CMFS. A uniform stagger is used for simplicity of the experimental design.

Copyright © 2004 John Wiley & Sons, Ltd.

Although a more precise characterization of arrival times would provide more precise results in the nature of the achievable readahead on admission decisions, we were interested in the range of performance effect. A realistic workload would be likely to have performance somewhere in between these two extremes. The individual block schedules in a scenario are combined into a *server block schedule*. All server block schedules that do not exceed the disk resource capacity are said to be *valid* schedules, corresponding to *valid* scenarios.

The simplest bandwidth characterization of a stream is a single number. In the *Simple Maximum* algorithm, we choose the maximum number of reads required in any slot. If the sum of this maximum value for the new stream plus the current sum of the values for the accepted set of streams is greater than *minRead*, the new stream is rejected. This significantly under-utilizes resources with VBR streams.

The next admission control algorithm considers the stream block schedule in making admission decisions. It uses the current server block schedule. If the value in *any slot* exceeds *minRead*, the new stream is rejected. We call this the *Instantaneous Maximum* algorithm.

We can take into account the amount of read-ahead possible during slots that have lower requirements than *minRead*. One option is to consider the *average blocks per slot* for each stream as the bandwidth characterization and simply sum the averages. This algorithm is called *Average* and will admit more VBR streams than the previous algorithms, but does not provide deterministic guarantees, and may also reject valid scenarios. Variants of the *Average* algorithm take into account the shape and relative occurrence of bandwidth peaks.

Biersack and Thiesse [6], Vin *et al*. [7], and Chang and Zakhor [8] all base their admission control decision on calculating an overflow probability and accepting all requests that provide an acceptable level of failure probability. They are all more sophisticated than the simplistic version of *Average* considered in this paper, because these are quantitative ways to estimate when and by how much the system will fail to meet its commitments. Since these algorithms consider more than just the simple average, they will be more conservative in admission decisions, given the same measure of system capacity. We acknowledge that our version of average is the most simplistic and weakest of all possible statistical algorithms and that more careful characterization as in Vin *et al*. or measurement on which to base system capacity will lead to a more realistic comparison. This comparison has been left for future work.

The final algorithm uses both read-ahead provided by buffer space at the server and the detailed disk block schedule for each stream. The full algorithm is called *vbrSim* and is given in our previous work [9]. The admission process emulates the disk block reading process for the duration of the server block schedule, assuming no knowledge of data layout. The disk is assumed to read *minRead* blocks per slot. If the requirements in a given slot are less than *minRead*, the remaining bandwidth is used to read blocks for future slots, reducing future bandwidth requirements and smoothing out peaks in the server block schedule. The *vbrSim* algorithm takes advantage of read-ahead that has already been accomplished in the past and assumes the server will read the *minRead* blocks in each future slot, if there is buffer space available.

Since *minRead* is a worst-case value, the disk almost always reads faster. Experimental measurements indicated that the bandwidth varies by as much as a factor of 3 between sequential reading of blocks on outer disk tracks and reading one block per track near the inside of the disk. If the server reads ahead as far as possible, it may use up all of the buffers. If a new request arrives during steady state with all buffers occupied, the *vbrSim* algorithm simulates the stealing of buffers for data with the latest deadline and reading instead the data for the new stream. If there are enough buffers

Copyright © 2004 John Wiley & Sons, Ltd.

available for stealing, the request is accepted. This procedure is shown to preserve admission decision correctness [9].

In order to calibrate the algorithms, they are compared to an optimal algorithm with complete knowledge of the future. It uses the bandwidth achieved for every slot time in the future and thus can be thought of as performing the same process as the *vbrSim* algorithm, but with the number of blocks actually read in each slot as the value for *minRead*.

NETWORK ADMISSION CONTROL/BANDWIDTH SMOOTHING

Network interface admission control

Much previous work has been done on the statistical multiplexing of variable bit-rate data streams. Constant bit-rate connections have also been used to transport variable bit-rate data [5]. Statistical approaches cannot provide absolute guarantees, since the entire concept of multiplexing is based on providing low, but non-zero probabilities of transient network switch overload. Knightly *et al.* [10] prove that their deterministic-guarantee admission control method on VBR channels in ATM networks significantly improves the network utilization over peak rate allocations, but still provides a low network utilization. This led to an investigation of statistical admission guarantees, constant bit rate channels or smoothing with startup latency. In Pseudo Constant Rate Transmission and Transport (PCRTT), the server negotiates a constant bit rate with the network for a specific stream for a portion of the stream's delivery time, and renegotiates that bandwidth periodically.

A network that provided absolute guarantees of quality of service would result in the most favorable environment. While the current Internet and IP-based network infrastructure does not provide absolute guarantees, the server can at least police its outgoing interface.

A detailed schedule of the bandwidth needed can be constructed in terms of network slot values. The system can transmit data at a constant rate during a network slot in accordance with the bandwidth values in the schedule. The approach taken by the network admission control in the CMFS is to provide a deterministic guarantee at the server using constant-bit-rate network channels [5].

Network slots should be significantly larger than a disk slot for two main reasons. The first is the overhead of renegotiation. A renegotiation takes a non-trivial amount of time and, therefore, should be effective for a substantial amount of time. The second is the ability to smooth out the data delivery by sending data at an earlier time in the network slot than is absolutely required, making use of client buffer space. Other research has experimented with network slots ranging from 10 s to 1 min in length [11,12]. Zhang and Knightly [12] suggest that renegotiations at 20 s intervals provide good performance.

An intriguing possibility is to use *vbrSim* for the network. The smoothing enabled by sending data early eliminates transient network bandwidth peaks. One benefit of *vbrSim* for the disk system is the use of server buffer space to store read-ahead data. The server buffer space is shared between a small number of streams and is typically large enough to hold the data for dozens of slots per stream. If the same relative amount of buffer space was available at each client, then *vbrSim*'s send-ahead method for the network system could be effective. The server model only requires enough client buffer space to handle double buffering for a slot's worth of data. With only the required client buffer, very little send-ahead is possible. According to the MPEG-2 specifications [13], memory for only three or four frames is required. Many megabytes of client buffer would be needed for this amount of video data. In intervening years, memory has decreased in price, permitting large client buffers.

Copyright © 2004 John Wiley & Sons, Ltd.

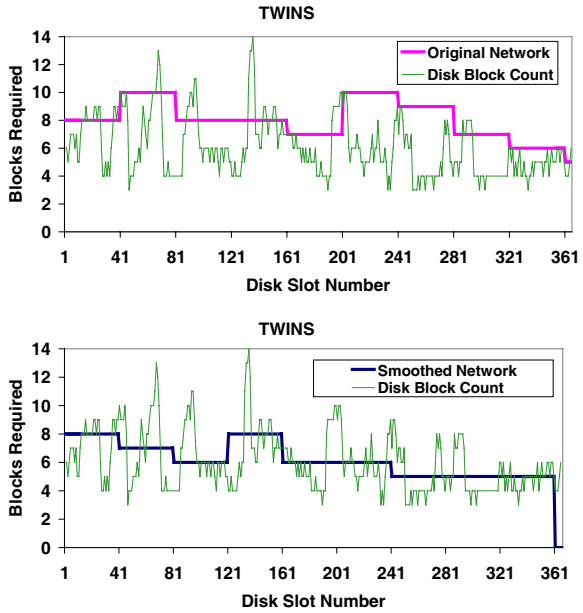

Figure 2. Network bandwidth schedule (minimum client buffer space).

The network and disk decisions could both be made on a disk-slot granularity. Unfortunately, this would require frequent network renegotiation. Grossglauser *et al.* [11] indicate that renegotiation intervals should be relatively long. With longer network slots, the amount of client buffer space for send-ahead shrinks in relative importance. We choose the *Instantaneous Maximum* algorithm as it outperforms *Simple Maximum* in all cases, and by definition *Average* will oversubscribe the network interface.

Network bandwidth smoothing techniques

Work on smoothing the network requirements [14–16] has been extensive, attempting to modify data traffic so as to reduce the network signaling and peak bandwidth usage. Three algorithms for constructing a network bandwidth allocation schedule are considered in the experiments. The first algorithm (denoted *Peak*) uses the peak disk bandwidth value in each network slot for the network schedule.

The second algorithm (denoted *Original*) considers the number of bytes that are required to be sent in each network slot independent of every other network slot. Each network slot is processed in order. The cumulative average bandwidth required for the network slot is calculated. The maximum of the cumulative averages is chosen as the bandwidth value for the network slot, enabling some peaks to be smoothed within the slot. Peaks which occur late in the network slot have marginally less influence in the cumulative average and are absorbed easily, as shown in Figure 2. The first three large peaks are at

Copyright © 2004 John Wiley & Sons, Ltd.

slots 68, 94, and 136 and have no effect. Unfortunately, if a peak in disk bandwidth occurs early in a network slot, then the maximum cumulative average for the slot is affected.

The final network bandwidth characterization algorithm improves on the second by explicitly accounting for the fact that sending excess data to the client reduces the amount of data that must be sent during the next network slot. Each slot 'carries forward' a *credit* of the number of bytes already in the client buffer at the beginning of a slot. Thus, this is called the *Smoothed* algorithm. Variations of this strategy have been presented in other work [17]. The second chart in Figure 2 shows the smoothed network bandwidth schedule for the same stream.

DISK ADMISSION PERFORMANCE RESULTS

A few general observations can be made about the disk admission experiments. Each disk was loaded with video streams that were between 1.7 and 7.2 Mbps in average bandwidth. The number of streams required to fill the disk ranged from 9 to 11. At these bandwidths, up to seven streams (for a total bandwidth of between 28 and 30 Mbps, depending on which streams were selected) can be supported simultaneously from a single disk without regard for admission control of any kind. Very few requests of seven simultaneous streams could be supported, but a stagger of as little as 10 s greatly increased the achieved bandwidth. Since each scenario used different disk resources (tracks, sectors, seeks), they were grouped in bands corresponding to the percentage of the disk bandwidth requested. The request band is the sum of the average bandwidth of each stream in the scenario divided by the actual disk bandwidth achieved during the execution of the scenario.

In particular, there are a few scenarios that request over 100% of the average bandwidth achieved, yet are still valid scenarios and are accepted by some algorithms. The highest actual bandwidth accepted was 37.2 Mbps, which is 117% of the achieved bandwidth in that particular disk scenario. This appears to be impossible on first glance. A closer look at the execution of the scenario shows that the bandwidth achieved varies over time. The cumulative average bandwidth measures disk performance more accurately. This often steadily decreases over time. There are two factors contributing to the decrease. First, as more streams become active, seeks increase, reducing the number of blocks that can be read. Second, the disk bandwidth decreases as the blocks requested are closer to the inside of the drive. The value used for achieved bandwidth is the minimum cumulative average.

Streams were selected according to their coefficient of variation and stored on separate disks for experiments which investigated the effect of variability in the streams. Table III divides the streams chosen into low-variability and high-variability streams, respectively.

Disk admission: algorithm comparison

Figure 3 shows the results of the first set of experiments with all four algorithms. Our goal with these experiments was to see how close each algorithm could come to accepting all of the valid scenarios supported by the disk (i.e. 100% of the disk capacity).

The results for the *Simple Maximum* algorithm were disappointing as expected. No scenarios requesting 50% or more were accepted. An anomaly appears in the 30–34% request range. Only three scenarios had that request range, and all three were rejected. Due to the dismal performance of *Simple Maximum*, it is not considered in any further experiments.

Copyright © 2004 John Wiley & Sons, Ltd.

Table III. Stream groupings.

Low variability		High variability	
Stream	Variability	Stream	Variability
Aretha	0.184	Football	0.290
Coaches	0.202	YES30	0.251
Rescue	0.209	Maproom	0.340
Joe Greene	0.185	Bloop93	0.287
Ray Charles	0.119	Laettner	0.312
FBI Men	0.201	Snowstorm	0.430
Plan-9	0.200	Twins	0.350
Country Music	0.224	Basketball	0.354
Fires	0.224	Dallas Cowboys	0.340
Tom Connors	0.154	Akira	0.260
Clinton	0.186	John Elway	0.299

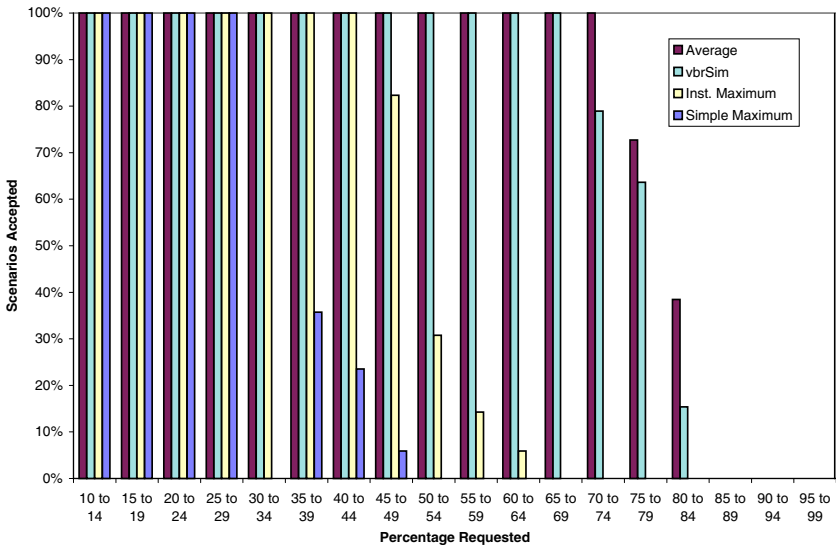

Figure 3. Acceptance rate—all algorithms.

More scenarios were accepted by *Instantaneous Maximum* than by *Simple Maximum*. The actual highest peak bandwidth value for *Instantaneous Maximum* was 20–30% lower than the value for *Simple Maximum* in corresponding scenarios. This almost always permitted an extra stream to be accepted, and this increased the utilization of the disk by 30% or more, as a three-stream scenario could be accepted by *Instantaneous Maximum*, while only two streams would be accepted by *Simple Maximum*. Since all

Copyright © 2004 John Wiley & Sons, Ltd.

requests that had cumulative bandwidth below *minRead* were accepted by the *Average* algorithm, regardless of their time-varying requirements, the acceptance should drop off suddenly immediately above *minRead*. All requests up to 75% were accepted. The graph does not completely go down to zero in the next request range, due to the disk bandwidth range.

The *vbrSim* algorithm performed much better than *Instantaneous Maximum* and was comparable to *Average*. Almost all scenario requests below 75% were accepted and a steady drop-off was observed as the requested level of disk utilization was increased. As seen in the previous paragraph, the range of *minRead/averageRead* is 0.76–0.85, showing that *vbrSim* does accept scenarios with cumulative average bandwidth very near the level of *minRead/averageRead*. For some of these scenarios, the cumulative average bandwidth required exceeded the value of *minRead*. This initial test shows that the *vbrSim* algorithm is quantitatively superior to the *Instantaneous Maximum* algorithm.

Using *minRead* as an estimate with large bandwidth video objects was a conservative decision, because *minRead* was calculated assuming streams requiring a physical seek for every block in every slot. In the case of video streams, fewer seeks were required, so the access time was dominated more by the transfer time than the seek time. As well, the initial portions of each stream were located further to the outside of the disk, where transfer rates are faster, A fairer comparison between *vbrSim* and *Average* would be to allow *Average* to use some observed average performance value as the estimate of disk bandwidth. As an appropriate value to use for this estimate is not easily determined, this is left as future work, which can use the extensive literature on statistical and measurement-based admission control. Thus, we would have a quantitative basis for the claim that *Average* accepts invalid scenarios. We expect that using a calibrated average value of disk performance for the *Average* algorithm would substantially outperform *vbrSim* for the simultaneous arrival case, but we feel that intuitively, this would also accept a significant percentage of invalid scenarios.

To determine the sensitivity of the three remaining disk admission control algorithms to arrival patterns of the streams, the next experiment selected a number of scenarios and presented them to each algorithm with three values of stagger: 0, 5, and 10 s. The results are shown for the low-variability streams in Figure 4.

For simultaneous arrivals, *Average* performs the best. It does not take into account the specific peaks in bandwidth, but since the disk performance is well above *minRead*, this does not cause any problems. It is interesting to note the degradation in performance for both *Instantaneous Maximum* and *Average* when stagger is introduced to the arrival pattern. These two algorithms do not correspondingly increase their estimate of disk guarantee due to the benefit of contiguous reading, and so accept fewer scenarios at the same relative amount of bandwidth. The *vbrSim* algorithm does not increase its estimate of the disk performance either, but takes advantage of the past disk performance and the amount of data read ahead when making its admission decisions, thus accepting a higher relative bandwidth.

Each scenario is longer with stagger between arrivals (i.e. 60 s longer for a 7-stream scenario with 10 s stagger), and thus the disk is reading the same number of blocks in a longer period of time. The cumulative bandwidth needed to supply these streams with data decreases in a manner proportional to the increase in scenario duration. By the time the first stream in the scenario is finished reading in the scenario with stagger, the number of required bytes is over 95% of the number of bytes needed in the simultaneous arrival case. Thus, the lengthening of the schedule does not account for the increase in acceptance rates.

Copyright © 2004 John Wiley & Sons, Ltd.

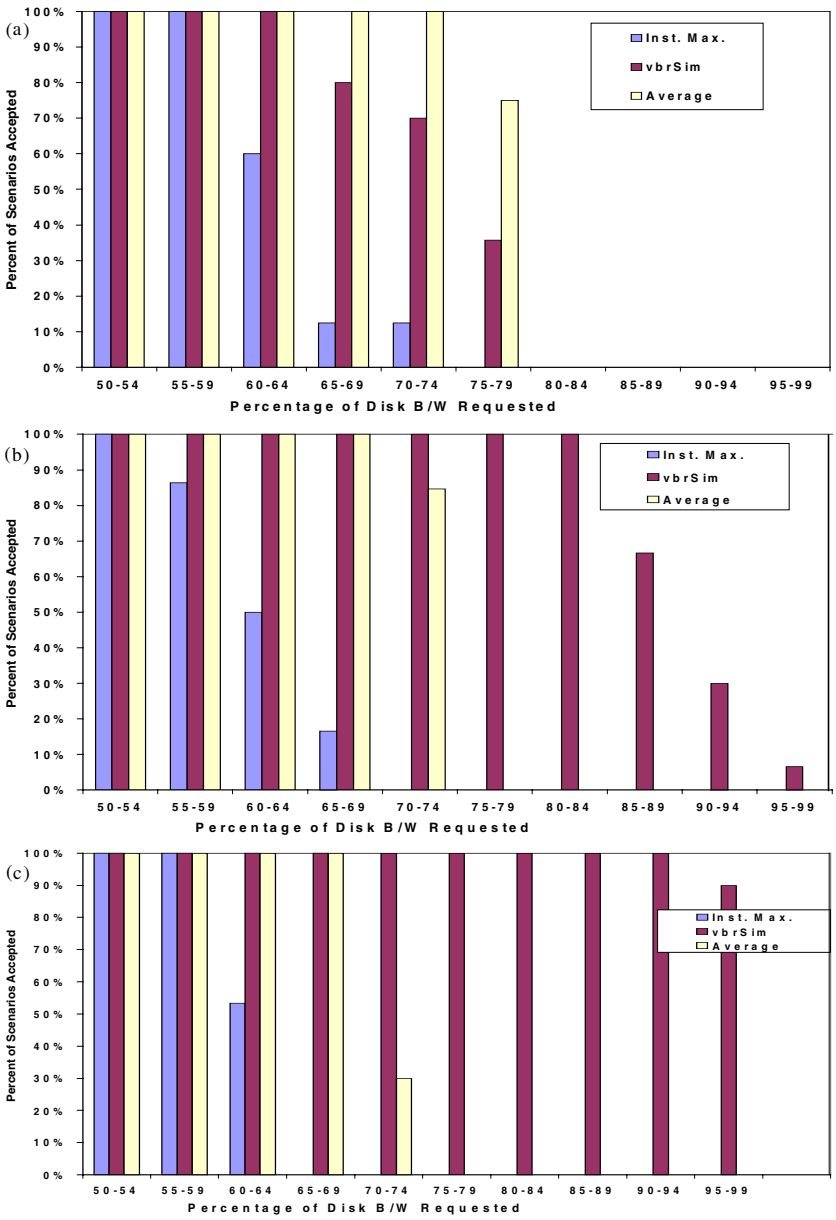

Figure 4. Low variability streams admission performance: (a) no stagger; (b) 5 s stagger; (c) 10 s stagger.

Copyright © 2004 John Wiley & Sons, Ltd.

vbrSim admission performance

Since *vbrSim* outperforms both of the other deterministic guarantee algorithms, the remaining performance tests were performed for the *vbrSim* algorithm only. We tested sensitivity to stream variability and server buffer space.

The first experiment was designed to determine the sensitivity of the *vbrSim* algorithm to the variability of the stream profiles. Therefore, three disks were loaded with different types of streams and the admission results were compared. One disk contained synthetically generated constant-bit-rate streams. The other two disks contained low-variability streams and high-variability streams, respectively, as in Table III.

The results for simultaneous arrivals of stream requests are shown in Figure 5(a). All scenarios of every variability factor that request more than 80% of the disk bandwidth are rejected. With constant bit-rate streams, all requests below 80% are accepted. This is reasonable to expect, since *minRead* is approximately 80% of the achieved bandwidth in the execution of these scenarios. Scenarios of low-variability streams are accepted at lower rates with requests close to 80%. Approximately 50% of the requests can be supported in the 70–74% range. Even fewer of the high-variability requests are accepted. Approximately 15% of the scenarios of high-variability streams in that request range are accepted. In the 60–65% range and the 65–70% range, there are still a significant number of scenarios of high-variability streams rejected. Only below 60% are virtually all scenarios of high-variability streams accepted. This confirms the intuition that *vbrSim* is sensitive to stream variability.

The acceptance rates of the same scenarios under staggers of 5 and 10 s are shown in Figures 5(b) and (c), respectively. With stagger equal to 5 s, all of the low-variability stream scenarios below 85% of the disk capability are accepted, and over half of those between 85% and 89%. This indicates that the admission algorithm is effective in making use of the request delay. While the peaks are generally of the same size in the simultaneous arrival and the staggered arrival case, they occur later in the scenario when more read-ahead has occurred. Since both the percentage accepted and the achieved bandwidth are increased significantly, the total bandwidth supportable increases to a level beyond *minRead* in many cases.

For the CBR streams and a 5 s stagger, many scenarios are accepted in the 95–100% range. The achieved read-ahead in the past reduces the overall level of bandwidth required for the remainder of the scenario. By the time the last stream is admitted, enough buffer space has been used to reduce the maximum slot to below *minRead*. Very few stream scenarios are rejected by the *vbrSim* algorithm with a stagger value of 10 s, since the high disk performance enabled a very large amount of data to be read ahead. The *vbrSim* algorithm effectively uses nearly all available disk bandwidth.

Some scenarios requesting more than 100% of the bandwidth are accepted. There were no CBR scenarios accepted, but some low variability and high variability scenarios were accepted. The VBR scenarios sustained their high-bandwidth request pattern for a smaller amount of the total scenario time (due to the difference in length), at an earlier point in the scenario. The number of streams actively being read off disk decreased much sooner, maintaining the bandwidth achieved.

When long staggers are used, the variability of individual streams appears to have a tiny effect on the acceptance rate of scenarios. A lot of smoothing takes place with buffering many slots' worth of data for the earlier streams, but using a large amount of buffer space.

Copyright © 2004 John Wiley & Sons, Ltd.

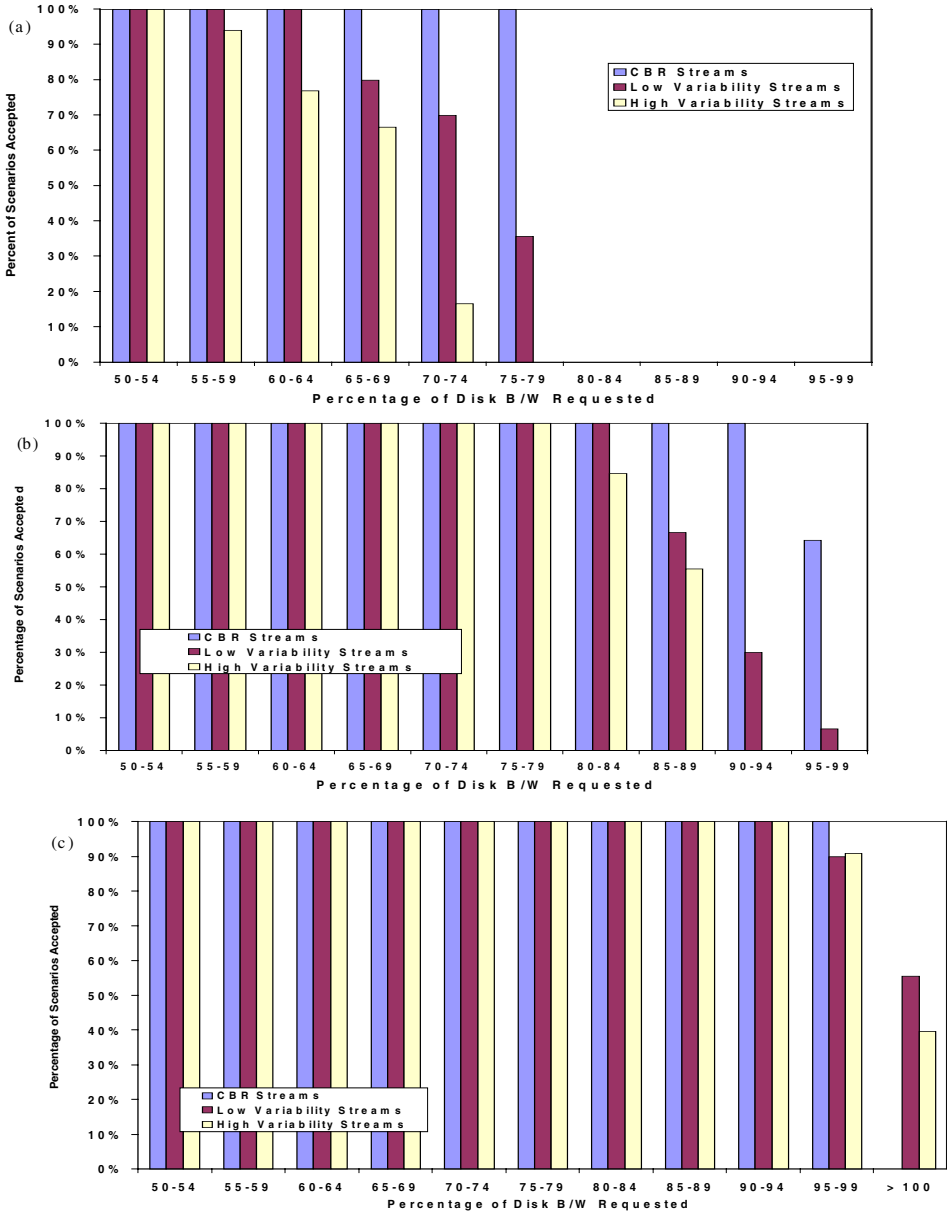

Figure 5. *vbrSim* Acceptance Rates: (a) no stagger; (b) 5 s stagger; (c) 10 s stagger.

Copyright © 2004 John Wiley & Sons, Ltd.

Buffer space

In order to take advantage of the read-ahead in any of the preceding scenarios, there must be sufficient server buffer space. It is expected to be greater for the high-variability streams, because of peaks which are above *minRead*. As the previous results show, a small amount of stagger with moderately short video streams is enough to increase the accepted bandwidth to the achieved disk bandwidth when the server is modeled with unlimited buffer space. Most of the increase in bandwidth is due to contiguous reading when only one stream is actively reading (i.e. immediately after stream acceptance).

The buffer space required to accept a scenario was calculated by static examination. The largest contiguous area of the scenario's requirements above *minRead* is found. The blocks referred to by the area in the scenario schedule above *minRead* must be in server buffers. Otherwise, the server cannot guarantee the delivery of the blocks to the clients, because there may not be sufficient disk bandwidth in the future.

The buffer space required for scenarios with CBR, low-variability and high-variability streams is shown in Figure 6. For requests of bandwidth below *minRead*, a small amount of buffer space was needed. The largest number of buffers needed for simultaneous arrivals of low-variability streams was 75 buffers (5 MB), when an average of 22 Mbps (or 97% of *minRead*) was requested. For scenarios of high-variability streams, the largest buffer request required 160 buffers (12 MB) and had an average bandwidth of approximately 20 Mbps.

With staggered arrivals, the pattern of buffer usage and buffer requirements is much different. The system can admit more streams due to contiguous reading, under the assumption that there is sufficient buffer space. Figures 6(b) and (c) show that most requests for bandwidth below *minRead* require a modest amount of buffer space. As the request bandwidth increases, the buffer space required increases somewhat linearly. Requests with bandwidth greater than *minRead* require steadily more buffer space, with the maximum buffer space needed being 4491 buffers (287 MB) for a scenario which had a staggered arrival interval of 10 s and requested 33.5 Mbps (104% of the achieved bandwidth and 150% of *minRead*). This is a substantial amount of memory. This scenario was comprised of the seven longest streams from the high-variability streams, and there was a long substantial peak in the bandwidth required.

Most of the requests with a 5 s stagger can be satisfied with fewer than 1500 buffers (96 MB). These requests use close to 100% of the achieved disk bandwidth. For the requests with 10 s stagger, 3000 buffers (192 MB) is enough for nearly all the scenarios which can be accepted and request less than 100% of the disk bandwidth. The largest buffer requirement for a scenario that requested less than 100% of the disk bandwidth was 2699 buffers (173 MB). Substantial, but not exorbitant, memory can accommodate requests that require a very high percentage of the disk bandwidth achievable.

NETWORK INTERFACE ADMISSION RESULTS

In this section, we report on the results of admission experiments that used the different smoothing characteristics of the bandwidth schedule creation algorithms. These are then compared with scenarios containing streams with different characteristics and finally with different network slot sizes. We conclude this section with a description on how the network interface and disk bandwidth algorithms can be combined to configure a realistic system.

Copyright © 2004 John Wiley & Sons, Ltd.

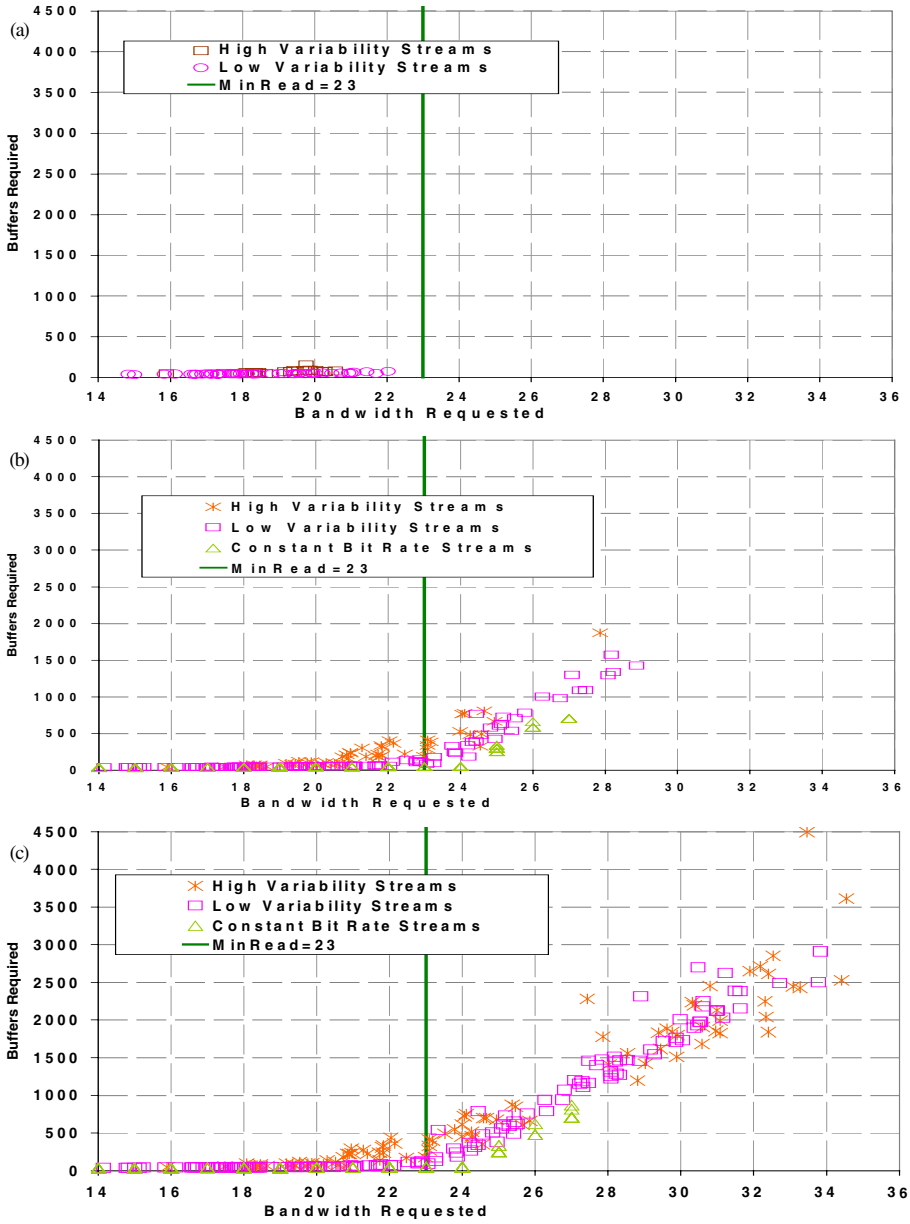

Figure 6. Buffer space requirements: (a) no stagger; (b) 5 s stagger; (c) 10 s stagger.

Copyright © 2004 John Wiley & Sons, Ltd.

Network bandwidth characterization results

The initial performance experiments were implemented with single disk servers and the results were combined in simulation as though they were executed on a multi-disk server, due to limitations in the hardware environment. The scenarios were grouped according to the sum of the average bit rates of each stream. The value for *maxXmit* was set at 100 Mbps, corresponding to the network interface cards that were installed in the server nodes. The experiments could not be done on the system itself, since in every environment at our disposal, either the host CPU or the device driver could not deliver the bandwidth from the server.

The results of admission are shown in Table IV. A total of 193 scenarios were generated and submitted to a CMFS with one server node and four disks. Only those scenarios that were valid from both the network and the disk subsystem point of view were considered. Only 92 scenarios were accepted by the disk subsystem; 184 scenarios were accepted by the disk system when stagger was introduced. Even though 184 scenarios with staggered arrivals were accepted by the disk, only 130 requested less than 100 Mbps and only those scenarios are shown in the results. It is clear that smoothing did change the number of streams and bandwidth that could be accepted by the network admission algorithm. A maximum of 80% of the network bandwidth was accepted by the *Original* algorithm on simultaneous arrivals, although about 60% of the scenarios in the range immediately below 80% (i.e. 75–79 Mbps) were accepted. The smoothing operation allows almost all scenarios below 85% request to be accepted, along with a small number with slightly greater bandwidth. As expected, the *Peak* algorithm accepted very few scenarios, none over 70%.

On the right side of Table IV, combining smoothing with staggered arrivals has a compounding effect on increasing the bandwidth supportable by the server. A few scenarios with a request range of between 80 and 90% can be accepted with the *Original* algorithm, which is a slight improvement over the simultaneous arrivals case. The *Smoothed* algorithm accepts nearly all requests below 90% of the network bandwidth, since only a few streams are reading and transmitting their first network slot simultaneously. The first network slot is the only one that cannot benefit from pre-sending data and cannot be smoothed. Thus, with simultaneous arrivals, it is more likely that the peaks occur in the first network slot. With staggered arrivals, the existing streams are sending at smoothed rates at arrival time, meaning lower peaks for the entire scenario.

Network stream variability results

In this section, the influence of stream variability on the performance of the bandwidth schedule creation algorithms is examined. Since the peak characterization performs so poorly, we do not consider it in the remainder of the experiments. To evaluate the stream variability factor, three configurations of streams were utilized: mixed-variability streams, low-variability streams, and high-variability streams. The first configuration consisted of 44 unique streams that had a mix of variability in the scenarios (denoted Mix). The second configuration contained streams with low variability (denoted Low). The 25 lowest variability streams were used with 19 replications to complete the 44 streams. The same process was used to obtain the third configuration (denoted High), with the 25 highest variability streams. Scenarios were generated and submitted to the simulation of a multi-disk, single-node CMFS. These scenarios generated between 58 and 135 Mbps of bandwidth (sum of the average bit-rates).

Copyright © 2004 John Wiley & Sons, Ltd.

Table IV. Network admission performance: stream characterization.

% of network	Characterization algorithm	Simultaneous arrivals	Staggered arrivals
90–94	Peak	0/5	0/19
	Original	0/5	0/19
	Smoothed	0/5	2/19
85–89	Peak	0/4	0/15
	Original	0/4	2/15
	Smoothed	2/4	14/15
80–84	Peak	0/18	0/27
	Original	1/18	3/27
	Smoothed	17/18	27/27
75–79	Peak	0/32	0/29
	Original	19/32	18/29
	Smoothed	32/32	29/29
70–74	Peak	0/19	0/22
	Original	18/19	22/22
	Smoothed	19/19	22/22
65–69	Peak	6/11	5/11
	Original	11/11	11/11
	Smoothed	11/11	11/11

The admission results are shown in Table V. For the low-variability streams, the acceptance rate of complete scenarios in the 80–84% range increased from 13/25 to 18/25 by using the *Smoothed* algorithm. The high-variability streams did not have any scenarios accepted in the 80–84% range with the *Original* algorithm, but this increased to 6/21 with the *Smoothed* algorithm. In the 75–79% range, the acceptance rate increased from 5/27 to 25/27 for high-variability streams. The network could admit scenarios at a higher network utilization overall, but smoothing had a more drastic effect on the acceptance rate with high-variability streams. The right side of Table V shows the admission performance for arrivals which are staggered by 10 s. The effect of smoothing is greater for the high-variability streams than for the low-variability streams.

Network slot size

While the disk admission control is based on relatively short disk-reading slots, the network admission is based on slots which are significantly longer. Selecting an appropriate slot length may substantially affect the network bandwidth schedule and subsequently the admission performance of the network admission control algorithm.

The network admission experiments in the section 'Network bandwidth characterization results' used 40 disk slots (20 s) as the length of the network slot. To establish the validity of the selection of 20 s as an optimal (or at least reasonable) choice for the network slot size, the admission performance

Copyright © 2004 John Wiley & Sons, Ltd.

Table V. Network admission performance: stream variability.

Request B/W %	Algorithm	Simultaneous			Staggered		
		Mixed	Low	High	Mixed	Low	High
90–94	Original	0/19	0/10	0/10	0/19	0/11	0/10
90–94	Smoothed	0/19	0/10	0/10	4/19	0/11	5/10
85–89	Original	0/15	0/3	0/17	3/15	0/3	0/17
85–89	Smoothed	4/15	0/3	0/17	13/15	2/3	9/17
80–84	Original	3/27	13/25	0/21	9/27	14/25	1/21
80–84	Smoothed	27/27	18/25	6/21	27/27	25/25	20/21
75–79	Original	18/29	24/25	5/27	20/29	25/25	10/27
75–79	Smoothed	29/29	24/25	25/27	29/29	25/25	27/27
70–74	Original	21/22	21/21	16/22	22/22	21/21	16/22
70–74	Smoothed	22/22	21/21	22/22	22/22	21/21	22/22
65–69	Original	11/11	22/22	25/27	11/11	22/27	26/27
65–69	Smoothed	11/11	22/22	27/27	11/11	22/27	27/27

Table VI. Network admission granularity: simultaneous arrivals.

B/W %	1/2 s	10 s	20 s	30 s	600 s
Low variability					
90–94	0/10	0/10	0/10	0/10	0/10
85–89	2/3	0/3	0/3	0/3	0/3
80–84	25/25	20/25	18/25	16/25	15/25
75–79	21/21	21/21	21/21	21/21	21/21
Mixed variability					
90–94	6/19	1/19	0/19	0/19	0/19
85–89	13/15	9/15	4/15	5/15	0/15
80–84	27/27	26/27	27/27	25/27	16/27
75–79	29/29	29/29	29/29	29/29	29/29
High variability					
90–94	0/10	0/10	0/10	0/10	0/10
85–89	5/17	3/17	0/17	0/17	0/17
80–84	11/21	14/21	5/21	4/21	1/21
75–79	27/27	27/27	25/27	26/27	17/27

is compared for network slot sizes of 1/2, 10, 30, and 600 s in addition to the original choice of 20 s. The 1/2 second slot is identical to the slot in the disk admission control algorithm. The 600 s slot case is very similar to allocating bandwidth on an average bit-rate basis.

The effect of the difference in slot sizes on the admission results for simultaneous arrivals is shown in Table VI. The worst admission performance occurs for the 600 s slot because of peaks early in the schedule. The best admission performance is for 1/2 s slots and that acceptance gets steadily worse as

Copyright © 2004 John Wiley & Sons, Ltd.

Table VII. Network admission granularity: staggered arrivals.

B/W %	1/2 s	10 s	20 s	30 s	600 s
Low variability					
90–94	0/10	7/10	0/10	0/10	0/10
85–89	2/3	3/3	3/3	2/3	0/3
80–84	25/25	25/25	25/25	25/25	15/25
75–79	21/21	21/21	21/21	21/21	21/21
Mixed variability					
90–94	6/19	13/19	4/19	6/19	0/19
85–89	13/15	15/15	13/15	13/15	0/15
80–84	27/27	26/27	27/27	26/27	16/27
75–79	29/29	29/29	29/29	29/29	29/29
High variability					
90–94	0/10	5/10	5/10	1/10	0/10
85–89	5/17	14/17	9/17	9/17	0/17
80–84	11/21	20/21	20/21	21/21	1/21
75–79	27/27	27/27	27/27	26/27	17/27

the network slot size is lengthened. The major reason for this behavior is precisely the simultaneous arrivals of the streams. With 1/2 s slots, there is a possibility that the peaks in some of the first few disk slots will be offset by valleys in others, which will reduce the peaks. Any peak in bandwidth near the beginning of a stream with a long network slot influences a longer amount of time within the scenario.

Staggered arrivals enhance the benefit of long network slots. With 1/2 s slots, approximately the same occurrence of peaks and valleys would occur regardless of the stagger. As network slot length increases, smoothing gives better acceptance rates, as shown in Table VII. The only request bracket with significant variation in all types of streams is the 90–94% range. In this range, the 10 s network slot performs significantly better than the 20 s network slot and the 30 s network slot in all but the high-variability streams. This provides a good balance between acceptance rate with both simultaneous arrivals and staggered arrivals and shows the need to minimize the overhead of re-negotiating network bandwidth reservations.

Network and disk scalability results

The results of this section enable the manner in which components can be added together to be evaluated. It is desirable that the disk and network bandwidth scale together.

In the configuration tested, 4 disks that could guarantee 23 blocks per slot time provided 92 Mbps of guaranteed bandwidth off of the disk with a network interface of 100 Mbps. At this level of analysis, it would seem a perfect match, but the tests with simultaneous arrivals did not support this conjecture. With simultaneous arrivals, a system configured with guaranteed cumulative disk bandwidth approximately equal to nominal network bandwidth was unable to accept enough streams at the disk in order to use the network resource fully. There were no scenarios accepted by the disk that

Copyright © 2004 John Wiley & Sons, Ltd.

requested more than 94% of the network bandwidth and only four scenarios of simultaneous arrivals in the 85–89% request range accepted by the disk system, whereas such scenarios of staggered arrivals could be supported.

When staggered arrivals were simulated, the network admission control became the performance limitation, as more of the scenarios were accepted by the disk. No scenarios requesting less than 100 Mbps were rejected by the disk. Thus, equating disk bandwidth with network bandwidth is an appropriate design point which maximizes resource usage for moderate bandwidth video streams of short duration with staggered arrival patterns.

RELATED WORK

Admission control, bandwidth evaluation, and the transmission characteristics of the communication between client and server machines have received significant attention in the literature. Early scalability work has been simulated to assess the resource needs of systems with hundreds of disks [18,19]. These show the levels of bandwidth required to support large numbers of users, but do not address the difficulties in building a system of that size.

Most of the previous work in this area considers the network only or the disk only and has not taken the opportunity to consider the interactions between the two. Much current work focuses on the details of network delivery of video packets, but ignores the disk aspects to its peril. The performance bottleneck will continue to shift between the disk and the network.

Servers with heterogeneous disks have been studied by Santos and Muntz [20] in the RIO multimedia storage server. Their focus was load balancing policies with selected replication to reduce the delay bounds on start-up for interactive requests, based solely on the disk bandwidth requirements. They also addressed performance of random-replication of data blocks in comparison with data striping techniques [21].

One of the first explicit considerations of variability was made by Dey-Sircar *et al*. [22] with planned bandwidth allocation when supporting multiple streams, but they did not provide a mechanism for allocating the bandwidth. Lau and Lui [18] also considered variable bit-rate retrieval to provide data for the client application. Their algorithms utilize a client-provided time bound on start-up latency to determine stream schedulability, given a limited set of resources. Peak rate is used in the admission test and startup is delayed to minimize other measures of resource usage. This approach explicitly considers the anticipated length of time required for disk-reading tasks, which may be variable.

Chang and Zakhor are also among those who have more directly experimented with more complicated versions of algorithms based on average bit rates and distribution of frame sizes. A more complex version of the *Average* algorithm for Constant Time Length (CTL) video data retrieval [8] was developed. They also investigated Constant Data Length retrieval methods which introduce buffering for the purposes of prefetching portions of the stream and incorporate a start-up latency period. In further work [23], they showed via simulation that a variation of deterministic admission control admits 20% more users than their statistical method for a small probability of overload. Unfortunately, their work is primarily theoretical and assumes that each stream considered has a similar profile, equal buffer sizes, and thus, is suited to a system with homogeneous data streams. They showed the difference in the number of streams that can be supported for different overload probabilities, but did not compare it to the actual capacity of the disk system. Our work utilizes the server buffer space on

Copyright © 2004 John Wiley & Sons, Ltd.

a finer granularity, accommodating different orders of magnitude of bandwidth, while not requiring prefetch latency under any circumstances, and approaches the total utilization of the disk.

Knightly *et al*. [10] performed a comparison of different admission control tests in order to determine trade-offs between deterministic and statistical guarantees. This is then combined with different packet transfer schemes and the results do not particularly isolate each subsystem. This has inspired other approximations [24] which are less accurate and less expensive to compute. In particular, Wrege and Liebeherr [24] utilized a prefix of the video trace as an aid in characterizing the traffic pattern. When combined with statistical multiplexing in the network, high levels of network utilization can be achieved [10]. Measurement-based algorithms, such as in Jamin *et al.* [25], are more appropriate for network delivery and need substantial modification to work in server environments.

Server buffer space is utilized significantly in the CMFS to allow additional bandwidth to be supported. Similar goals were addressed by Feng *et al*. [26] to reduce the variability in frame rate experienced by the end user, by using both server buffer and client pre-fetch buffers.

One technique for improving performance of on-demand delivery of video data is to share the information amongst many users that are reasonably correlated in time and use multicast transmission schemes. This has been analyzed and simulated by several researchers [15,27,28,30]. The approaches range from batching to patching. One other approach is adaptive piggybacking [29], which alters the frame display rate at the client to enable merging of requests. Further analysis on sharing [30] examines bandwidth savings and system capacity planning issues. Some of these approaches replicate video files across striped storage media for hight performance in the normal case, while providing fault tolerance and graceful degradation in the case of failures.

CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a substantial performance study of a CMFS with respect to the workload that it can support in a guaranteed manner to the set of client applications that request delivery of the media data. This server has been implemented and tested on a variety of hardware platforms. The performance testing has utilized a wide range of video streams that are typical of a news-on-demand environment. The abstract disk model permits streams of widely differing VBR profiles and varying formats to be stored on the same server node without adverse effect on the guaranteed performance of the server.

The most significant contribution is the analysis of the disk admission control algorithm that explicitly utilizes a detailed bit-rate profile for each stream, simulating the use of server buffer space for the reading ahead of data. The *vbrSim* algorithm significantly outperforms the other deterministic algorithms in terms of admission performance. The performance tests evaluated the algorithms for scenarios of large bandwidth video streams of differing variability and arrival patterns. The admission performance of the *vbrSim* algorithm improved with staggered requests, due to the incorporation of achieved read-ahead and guaranteed future read-ahead. With reasonably small values of stagger, scenarios that request a cumulative bandwidth which is greater than *minRead* and close to the actual bandwidth achieved by the disk system can be accepted. We also showed that the *vbrSim* algorithm is somewhat sensitive to the variability of the bit rate within the stream.

Buffer space requirements for the *vbrSim* algorithm were found to be large, but not excessive. The scenarios which requested more than *minRead* blocks per slot required significant buffer space

Copyright © 2004 John Wiley & Sons, Ltd.

to guarantee delivery from the server for every stream. Most scenarios that were acceptable in terms of bandwidth required less than 200 MB of server buffer space. The required space appeared to grow linearly with the cumulative bandwidth of the scenario for requests above *minRead*.

The network admission control algorithm and network bandwidth smoothing technique provided an efficient use of the network bandwidth available from the network interface of a server node. A slightly modified *Instantaneous Maximum* algorithm with the *Smoothed* network bandwidth characterization can accept scenarios with over 90% of the network interface limit requested. This is 10–15% better than using the *Original* network characterization method and far superior to the *Peak* characterization. With respect to different stream types, the *Smoothed* algorithm showed more performance improvements for the high-variability streams, because there are more peaks to smooth.

The value of *minRead* is set conservatively to ensure that there are no accepted requests that exceed the disk capacity. We found that in many cases, the disk capacity was much greater than *minRead*. If the pattern of requests is known, it may be possible to reschedule disk reads to avoid seeks, or at least predict the number of seeks required per slot to have an increased value of *minRead* in some environments. This does take a risk, however, because there is no guarantee that logically contiguous blocks will not require arbitrary seeks. Part of the future work in analyzing this system is to see if greater bandwidth can actually be supported by relaxing *minRead* in those cases.

REFERENCES

1. Burchard L-O, Luling R. An architecture for a scalable video-on-demand server network with quality-of-service guarantees. *Interactive Distributed Multimedia Systems and Services (IDMS) 2000*, Enschede, Netherlands, October 2000. Springer: Berlin/New York, 2000; 132–143.
2. Bernhardt C, Biersack E. The server array: A scalable video server architecture. *High Speed Networking for Multimedia Applications*, Spaniol O, Effelsberg W, Danthine A, Ferrari D (eds.). Kluwer: Dordrecht, 1996; 103–125.
3. Kumar M, Kouloheris JL, McHugh MJ, Kasera S. A high performance video server for broadband network environment. *IST&SPIE Multimedia Computing and Networking*, San Jose, CA, January 1996. SPIE Press: Bellingham, WA, 1996; 410–421.
4. Makaroff D, Neufeld G, Hutchinson N. Design and implementation of a VBR continuous media file server. *IEEE Transactions on Software Engineering* 2001; **27**(1):13–28.
5. McManus JM, Ross KW. Video on demand over ATM: Constant-rate transmission and transport. *IEEE INFOCOM*, San Francisco, CA, October 1996. IEEE Computer Society Press: Los Alamitos, CA, 1996; 1357–1362.
6. Biersack E, Thiesse F. Statistical admission control for video servers with variable bit rate streams and constant time length retrieval. *Euromicro '96*, Prague, Czech Republic, September 1996. IEEE Computer Society Press: Los Alamitos, CA, 1996; 633–642.
7. Vin HM, Goyal P, Goyal A, Goyal A. A statistical admission control algorithm for multimedia servers. *ACM Multimedia*, San Francisco, CA, October 1994. ACM Press: New York, 1994; 33–40.
8. Chang E, Zakhor A. Admissions control and data placement for VBR video servers. *1st IEEE International Conference on Image Processing*, Austin, TX, November 1994. IEEE Signal Processing Society: Piscataway, NJ, 1994; 278–282.
9. Neufeld G, Makaroff D, Hutchinson N. Design of a variable bit rate continuous media file server for an ATM network. *IST&SPIE Multimedia Computing and Networking*, San Jose, CA, January 1996. SPIE Press: Bellingham, WA, 1996; 370–380.
10. Knightly EW, Wrege DE, Liebeherr J, Zhang H. Fundamental limits and tradeoffs of providing deterministic guarantees to VBR video traffic. *ACM SIGMETRICS '95*, Ottawa, Canada, May 1995. ACM Press: New York, 1995; 98–107.
11. Grossglauser M, Keshav S, Tse D. RCBR: A simple and efficient service for multiple time-scale traffic. *ACM SIGCOMM*, Boston, MA, August 1995. ACM Press: New York, 1995; 219–230.
12. Zhang H, Knightly EW. A new approach to support delay-sensitive VBR video in packet-switched networks. *5th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, Durham NH, April 1995. Springer: Berlin/New York, 1995; 381–397.
13. ISO/IEC JTC1/SC29/WG 11 Editorial Group. Video generic coding of moving pictures and associated audio information. *Technical Report MPEG-2 DIS 13818-7*, International Standards Organization, Geneva, Switzerland, 1996.

Copyright © 2004 John Wiley & Sons, Ltd.

14. Feng W. Video-on-demand services: Efficient transportation and decompression of variable-bit-rate video. *PhD Thesis*, University of Michigan, 1997.
15. Sen S, Gao L, Rexford J, Towsley D. Optimal patching schemes for efficient multimedia streaming. *Network and Operating Systems Support for Digital Audio and Video 99*, Baskin Ridge, NJ, June 1999. Academic Press: San Diego, CA, 1999; 265–277.
16. Sen S, Towsley D, Zhang Z, Dey JK. Optimal multicast smoothing of streaming video over an internetwork. *IEEE INFOCOM*, New York, NY, March 1999. IEEE Computer Society Press: Los Alamitos, CA, 1999; 455–463.
17. Zhang Z, Kurose J, Salehi JD, Towsley D. Smoothing, statistical multiplexing and call admission control for stored video. *Special Issue on Real-Time Video Services in Multimedia Networks. IEEE Journal of Selected Areas in Communications* 1997; **15**(6):1148–1166.
18. Lau SW, Lui JCS. A novel video-on-demand storage architecture for supporting constant frame rate with variable bit rate retrieval. *5th International Workshop on Network and Operating System Support for Digital Audio and Video*, Durham, NH, April 1995. Springer: Berlin/New York, 1995; 294–305.
19. Rangan PV, Vin HM, Ramanathan S. Designing an on-demand multimedia service. *IEEE Communications Magazine* 1992; **30**(7):56–64.
20. Santos JR, Muntz R. Performance analysis of the RIO multimedia storage system with heterogeneous disk configurations. *ACM Multimedia 98*, Bristol, UK, September 1998. ACM Press: New York, 1998; 303–308.
21. Santos JR, Muntz R, Ribiero-Neto B. Comparing random data allocation and data striping in multimedia servers. *ACM Sigmetrics 2000*, Santa Clara, CA, June 2000. ACM Press: New York, 2000; 44–55.
22. Dey-Sircar JK, Salehi J, Kurose J, Towsley D. Providing VCR capabilities in large-scale video servers. *ACM Multimedia*, San Francisco, CA, October 1994. ACM Press: New York, 1994; 25–32.
23. Chang E, Zakhor A. Cost analyses for VBR video servers. *IST&SPIE Multimedia Computing and Networking*, San Jose, CA, January 1996. SPIE Press: Bellingham, WA, 1996; 381–397.
24. Wrege DE, Liebeherr J. Video traffic characterization for multimedia networks with a deterministic service. *IEEE INFOCOM*, San Francisco, CA, March 1996. IEEE Computer Society Press: Los Alamitos, CA, 1996; 537–544.
25. Jamin S, Danzig P, Shenker S, Zhang L. A measurement-based admission control algorithm for integrated services packet networks. *ACM SIGCOMM Computer Communication Review*, Boston, MA, August 1995. ACM Press: New York, 1995; 2–13.
26. Feng W, Krishnaswami B, Prabhudev A. Proactive buffer management for the streamed delivery of stored video. *ACM Multimedia 98*, Bristol, UK, September 1998. ACM Press: New York, 1998; 285–290.
27. Eager D, Vernon M, Zahorjan J. Optimal and efficient merging schedules for video-on-demand servers. *ACM Multimedia*, Orlando, FL, November 1999. ACM Press: New York, 1999; 199–202.
28. Sen S, Rexford J, Towsley D. Proxy prefetch caching for multimedia streams. *IEEE INFOCOM*, New York, NY, March 1999. IEEE Computer Society Press: Los Alamitos, CA, 1999; 1310–1319.
29. Golubchik L, Lui JCS, Muntz RR. Reducing I/O demands in video-on-demand storage servers. *ACM SIGMETRICS '95*, Ottawa, Canada, May 1995. ACM Press: New York, 1995; 25–36.
30. Leung MYY, Lui JCS, Golubchik L. Use of analytical performance models for system sizing and resource allocation in interactive video-on-demand systems employing data sharing techniques. *IEEE Transactions on Knowledge and Data Engineering* 2002; **14**(3):615–637.

Copyright © 2004 John Wiley & Sons, Ltd.