Disk Performance and VBR Admission Control for Media Servers

Dwight Makaroff Jason Coutu Fujian Liu

Department of Computer Science
University of Saskatchewan
Saskatoon, SK, S7N 5C9, Canada
{makaroff, jdc135, ful113}@cs.usask.ca

Abstract

Streaming applications have used many different approaches to provide delay-sensitive data to clients across networks. Previous work [4] showed that a detailed deterministic-guarantee admission algorithm can achieve close to optimal results with little overhead on commodity hardware. Changes in hardware technology may influence the nature of admission control results.

We evaluate admission performance based on bandwidth and buffer space limitations on multiple deployments of the server. The admission control algorithm used in this work can accept scenarios that utilize a high percentage of the available disk bandwidth when the average disk performance is close to the worst case performance. Increasing variability in disk bandwidth makes admission control decisions more conservative, but modifying a parameter of the algorithm helps achieve high acceptance rates.

Keywords*: admission control, variable bit-rate video, disk performance, system support*

1. Introduction

The delivery of stored video content has received much attention in the research and commercial communities. Client concerns have been CPU resources for decoding, as well as download bandwidth. Early systems supported a fixed number of constant bit-rate (CBR) streams [9]. RAID disk subsystems were necessary to get sufficient bandwidth for video. Over-provisioning both the disk and network to support variable bit-rate (VBR) was also one early technique.

There have also been research projects concerning admission control, ranging from statistical admission control [10, 11] to more advanced deterministic guarantees [2, 5]. None of these have been widely deployed, due to the scope/nature of the projects and the risk in having complicated mechanisms that may not be economically beneficial.

In this work, we consider high-quality, variable-bit-rate video streams that are of short playback duration, such as for news and/or music-video-on-demand environments that can be supported by a system composed of commodity personal computers. Substantial changes have taken place in the resource capabilities of this class of computers since the first generation of such systems. For a given cost, disk capacity, RAM size and CPU speeds have increased by an order of magnitude or more.

The increased variability of the disk subsystem has substantial influence on the total bandwidth that can be guaranteed from the server. If the average performance of the disk is less than 50% better than the worst case, our admission control has good results. With highly variable disk performance and substantial sequential reading, the performance guarantee is reduced. When the *minimum average past performance* is used as the worst case bandwidth estimation, however, the system achieves high utilization of the disk.

The remainder of this paper is organized as follows. Section 2 describes the system model. In Section 3, we provide the context of the experiments measurements. The results obtained for MPEG-2 video data clips are given in in Section 4. Section 5 describes the related work, while we conclude in Section 6 with indications for future work.

2. System Model

The Continuous Media File Server (CMFS) [5] is organized as a distributed system, with an administrator node and several server nodes. Server nodes can be added until the bandwidth of a particular network interface is saturated. Our admission control algorithm is called *vbrSim* and has been described and evaluated in other previous work [4, 6]. The main parameter to *vbrSim* is an empirically-derived measure [5] of the guaranteed disk bandwidth, hereafter called *minRead*. Data is read in a periodic fashion. Dur-

ing each time period (called a *slot*[1]), the disk is guaranteed to read at least *minRead* blocks in an earliest-deadline-first manner.

This assumption has the effect of reducing requirements in future slots, should a present slot's requirements be less than *minRead*. The CMFS will read faster than required, and store the data in server buffers. This uses slack bandwidth to smooth even more peaks in the future requirements. In the CMFS, the user can choose starting points (*start*/*stop* parameters) and fast motion or slow motion (*speed* parameter). High-speed scan delivers a subset of the frames (*skip* parameter). Each parameter can be independently modified by the user. The interaction of these parameters changes the VBR profile of the stream, and thus, the disk requirements, so that delivery schedules cannot be precomputed.

Buffers hold blocks that are read in advance of their deadline. A modest amount of buffer space can be allocated that corresponds to non-trivial playback time per stream. For example a 500 MB buffer could hold 20 MB for each of 25 streams[2]. Any stream requests arriving during steady state would read contiguously to catch up. This sequential reading provides great bandwidth improvement.

3. Experimental Environment and Design

The server nodes have AMD-Athlon processors (dual-booting FreeBSD 1.6 and Intel Solaris 2.8), with SCSI disks attached via SCSI-Ultra-160 buses (see Table 1). There are 4 basic hardware configurations for the experiments: Solaris-small disk, Solaris-big disk, FreeBSD-small disk and FreeBSD-big disk. The experiments were performed once per configuration. A small number of scenarios were repeated multiple times to confirm the stability of the achieved bandwidth.

Characteristic	Big Disk	Small Disk
Manufacturer	Seagate	Quantum
Brand	ST318406LW	Atlas IV
Capacity (GBytes)	18	9
RPM	10000	5400

Table 1: Disk Characteristics

Each disk was loaded with medium-length video clips (2 to 12 minutes), typical of news-on-demand or music-on-demand scenarios. The average bit rate varied from 1.46 Mbps to 8.55 Mbps, with an average of 5.55 Mbps. All streams were encoded using a software MPEG-2 encoder, with VBR encoding at 30 fps at a resolution of 640×480.

Request scenarios are randomly generated. First, the file to be requested is selected. Then, the portion of the file, the direction of desired playback, *speed* and *skip* are selected modeling varying modes of user requests.

Four arrival patterns are examined for the small disk: unique requests with mean inter-arrival time of 500 ms and 1000 ms, and repeated requests at 500 ms arrivals and 1000 ms. The arrival pattern of the requests is Poisson, with the inter-arrival time set to capture different traffic intensities. With this traffic intensity range, all requests arrive within the first minute and no more can be accepted until at least one stream has completed service.

The experiments use a server with unlimited buffer space in order to isolate the bandwidth effects from the limitations imposed by buffer space. Thus, longer delays between arrivals (i.e. lower traffic intensity) will result in the server buffering large amounts of each stream before the next request arrives, leading to decreased parallel reading. In the extreme, this becomes downloading. If enough *buffer space* is provided to enable acceptance of rapidly occurring requests, then scenarios in which requests arrive more slowly will have the *bandwidth* to support the request.

We consider the Cumulative Average Bandwidth Requested (CABR) as a percentage of the bandwidth achieved for that particular scenario (BA) The number of streams is a meaningless quantity; different streams have different total/average bandwidth requirements. Total bandwidth requested is also unsatisfactory, because a request for B Mbps may or may not be supported by the disk system, depending on the number of seeks and the data location. Scenarios with the same percentage of disk requests (CABR/BA) are hereafter referred to as a *request band*.

Bandwidth measures are estimates. *Request* bandwidth is the sum of the average bandwidth of each stream. This is an *overestimate* of the actual bandwidth, due to the varying stream playback length, but is accurate when all streams are actively using the disk. *Achieved* bandwidth is also an estimate. The minimum stable value of the average bandwidth is used as the measure of achieved disk performance.

High accept rates for scenarios in request bands near 100% of disk bandwidth capacity is a desirable outcome. Acceptances above 100% are possible, due to the high bandwidth available at the beginning of the scenario.

4. Results

4.1. Disk Bandwidth

The results for minimum bandwidth calculations and minimum observed behaviour is shown in Table 2. A set of calibration programs were run on each configuration to determine *minRead*. These programs measured read times from pathological disk block locations. FreeBSD provides a guaranteed minimum bandwidth slightly less than 10% better than Solaris. The situation is reversed for the large disk,

1 In the experiments, the slot time is 500 ms

2 25 seconds of 6 Mbps MPEG-2 video

where Solaris exceeds FreeBSD by 16%. Since the exact same disks were used, this difference is perplexing.

For the small disk on Solaris, the bandwidth achieved was on average 142% of *minRead* (64 blocks per slot), but the range varied due to block location and amount of sequential reading. The average performance on FreeBSD is similar. The difference between observed minimum and guaranteed minimum is much greater on the large disk. The lowest observed value on FreeBSD is 2.2 times *minRead*. The difference in performance between operating systems is part of future work. This will determine if there are OS-level optimizations that can significantly improve performance.

Disk	FreeBSD		Solaris	
	min	Observed	min	Observed
9 GB Quantum	50	60 to 82	46	54 to 73
18 GB Seagate	78	172 to 205	91	161 to 206

Table 2: Average Disk Performance

4.2. Small Disk

Figure 1 shows the admission results for Solaris and FreeBSD with unique stream requests. The results for the scenarios with repeated requests are similar. We can see that the percentage of the scenarios accepted decreases as percentage of bandwidth requested approaches 100%. Since relatively few scenarios could be supported by the disk at a rate greater than 100%, the results for the high request bands are not as reliable. Neither FreeBSD nor Solaris could accept many scenarios which requested over 85% of the disk bandwidth capacity at either arrival rate.

When an additional stream is requested, it is most often the case that the disk system cannot support the request. A new request increases the request percentage nearly 10%. Moving from 80% to 90% shows a decrease in acceptance rates, but moving from 90% to 100% shows a increase in the failure of the disk system to deliver the data.

Recall that the minimum of observed performance for these scenarios was 54 blocks/slot on Solaris and 60 on FreeBSD. If we set *minRead* to the minimum observed value of disk performance in the recent past, there is an increase in acceptance rates. These configurations accept nearly all scenarios requesting below 92% of disk bandwidth capacity.

Buffer space usage results are fairly consistent with the previous work. All the accepted scenarios require fewer than 10000 buffers (or 640 MBytes). This amount of memory to dedicate for buffers is available on desktop machines, but is rather large. Over 99% of the scenarios can be accepted with 6000 or fewer buffers (less than 400 MBytes), a reduction of 40% from the worst case amount of buffer space, with a small decrease in acceptance rates.

4.3. Large Disk

On the larger disk, the results are not as encouraging. The same shape of acceptance rates is observed. Most streams below 50% of disk bandwidth capacity are accepted, with a steady decrease between 50% and 60%, with only a few acceptances beyond 60%, even with extended inter-arrival times of 1500 ms and 2000 ms. This greatly under-utilizes the disk bandwidth and is not shown.

When *minRead* is set to the minimum *observed* bandwidth, the admission control results improve dramatically. On FreeBSD, the value used was 149, based on the lowest observed bandwidth in any *individual* slot in any of the scenarios, and therefore lower than any of the minimum cumulative averages. This is also almost twice *minRead*. The results are shown in Figure 2. Note that this is very similar to the small disk results with *minRead* equal to the observed minimum, in that almost all scenarios requesting less than 92% of disk capacity are accepted. Most scenarios with more than 100% of disk capacity are rejected as they cannot be supported.

The use of buffer space on the large disk is similar to that for the small disk. It is not shown, due to space constraints. The overall shape is the same, but the range is larger. Most scenarios between 5000 and 20000 buffers (320 MBytes to 1.3 GByte) of buffer space for a single disk. This is large, but not excessive for a single disk video server.

5. Related Work

One method of providing increased robustness in the bandwidth and increased flexibility in the scalability of the server has used data layout techniques, such as randomized placement on heterogeneous disks [7]. The Yima Continuous Media Server [8] uses pseudo-random placement [3] and statistical admission control [11] to increase the flexibility for variable delivery rates and interactivity. The admission control mechanism used by Zimmerman and Fu [11] explicitly considers writing and reading at the same time, but involves very complex mathematical formulations and complex disk modeling to get the most benefit out of the disk system. This disk modeling needs only be done once per disk, which is similar to our calibration method.

A similar system to the CMFS described in this paper is implemented and evaluated by Dimitrijevic *et al.* [2]. They have 3 components, an I/O Scheduler, a Request Scheduler, and an Admission Controller. The system uses conservative and aggressive versions of the admission control and achieves close to optimal results as well.

Server-based smoothing [1] shows an increase in the number of streams can be supported over non-smoothed streams, but such optimized algorithms require tens of seconds to smooth 30 minute clips. Our linear algorithm works on-line in less than a millisecond for 1 hour clips.

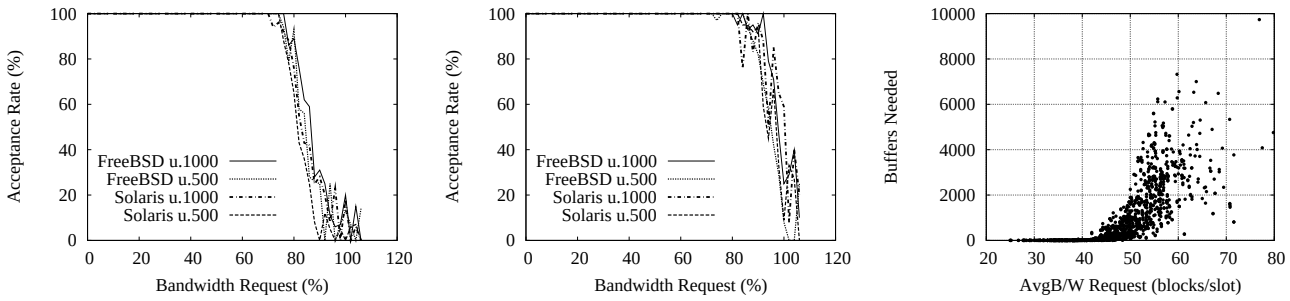

Figure 1: Small Disk Results

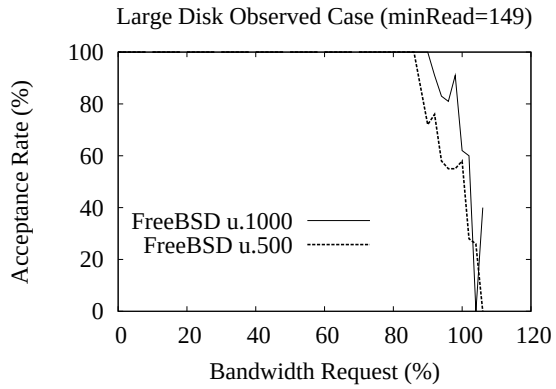

Figure 2: Admission for Large Disk

6. Conclusions and Future Work

The advances in performance of desktop disks and increased availability of memory for buffers have made it possible to provide video service with commodity components that can scale down to a modest single-disk video server.

The *vbrSim* admission control algorithm provided encouraging results several years ago. Experiments with more modern disks show that good admission results are still achieved when average and worst-case bandwidth guarantee are less than a factor of 1.5 apart. When the variability of the disk bandwidth is increased and the worst-case guarantee is used, the system greatly under-utilizes the available resources. Relaxing the value of *minRead* to the *observed* minimum provides good results in this case.

Future work will consider the differences in admission behaviour for feature-length streams and increased arrival rates. More detailed instrumentation of the server is necessary so that it can measure the resource usage of bandwidth and buffer space simultaneously.

Another issue to (re)examine is the data layout of streams on the disk. Our architecture does not make assumptions about the location on the disk, but contiguous allocation of media data for each object permits a great increase in bandwidth. Disk capacity is now another order of magnitude larger than the devices used in this study, with likely even more variability.

References

[1] S. V. Anastasiadis, K. C. Sevcik, and M. Stumm. Server-Based Smoothing of Variable Bit-Rate Streams. In *ACM Multimedia*, pages 147–158, Ottawa, ON, Canada, Sept. 2001.

[2] Z. Dimitrijevic, R. Rangaswami, and E. Chang. The xtream multimedia system. In *IEEE International Conference on Multimedia and Expo*, pages 545–548, Lausanne, Switzerland, Aug. 2002.

[3] A. Goel, C. Shahabi, S.-Y. Yao, and R. Zimmermann. SCADDAR: An Efficient Randomized Technique to Reorganize Continuous Media Blocks. In *Proceedings of the International Conference on Data Engineering*, pages 473–482, San Jose, CA, 2002.

[4] D. Makaroff, G. Neufeld, and N. Hutchinson. An Evaluation of VBR Disk Admission Algorithms for Continuous Media File Servers. In *ACM Multimedia*, pages 143–154, Seattle, WA, Nov. 1997.

[5] D. Makaroff, G. Neufeld, and N. Hutchinson. Design and Implementation of a VBR Continuous Media File Server. *IEEE Transactions on Software Engineering*, 27(1):13–28, Jan. 2001.

[6] D. Makaroff, G. Neufeld, and N. Hutchinson. Performance Evaluation for VBR Continuous Media File Server Admission Control. *Software Practise and Experience*, 34:1187–1210, July 2004.

[7] J. Santos, R. Muntz, and B. Ribeiro-Neto. Comparing Random Data Allocation and Data Striping in Multimedia Servers. In *ACM Sigmetrics*, pages 44–55, Santa Clara, CA, May 2000.

[8] C. Shahabi, R. Zimmermann, K. Fu, and S. Yao. Yima: A Second-Generation Continuous Media Server. *IEEE Computer*, 35(7):56–64, 2002.

[9] F. A. Tobagi, J. Pang, R. Baird, and M. Gang. Streaming RAID - A Disk Array Management System For Video Files. In *ACM Multimedia*, pages 393–400, Anaheim, CA, June 1993.

[10] H. M. Vin, P. Goyal, A. Goyal, and A. Goyal. A Statistical Admission Control Algorithm for Multimedia Servers. In *ACM Multimedia*, pages 33–40, San Francisco, CA, Oct. 1994.

[11] R. Zimmermann and K. Fu. Comprehensive Statistical Admission Control for Streaming Media Servers. In *ACM Multimedia*, pages 75–85, Berkeley, CA, 2003.