Comprehensive Statistical Admission Control for Streaming Media Servers*

Roger Zimmermann, Kun Fu
Computer Science Department
University of Southern California
Los Angeles, California 90089
[rzimmerm, kfu]@usc.edu

ABSTRACT

Streaming media servers and digital continuous media recorders require the scheduling of I/O requests to disk drives in real time. There are two accepted paradigms to achieve this: deterministic or statistical. The deterministic approach must assume larger bounds on such disk parameters as the seek time, the rotational latency and the transfer rate, to guarantee the timely service of I/O requests. The statistical approach generally allows higher utilization of resources, in exchange for a residual probability of missed I/O request deadlines. We propose a novel statistical admission control algorithm called TRAC based on a comprehensive three random variable (3RV) model to support both reading and writing of multiple variable bit rate media streams on current generation disk drives. Its major distinctions from previous work include (1) a very realistic disk model which considers multi-zoning of disks, seek and rotational latency profiles, and unequal reading and writing data rate limits, (2) a dynamic bandwidth sharing mechanism between reading and writing, and (3) support for random placement of data blocks. We evaluate the TRAC algorithm through an extensive numerical analysis and real device measurements. The results show that it achieves a much more realistic resource utilization (up to 38% higher) as compared with the best, previously proposed algorithm based on a single random variable (1RV) model. Most impressive, in all the experiments the difference between the results generated by TRAC and the actual disk device measurements match closely.

Categories and Subject Descriptors

H.2.4 [**Information Systems**]: Database Management—*Multimedia Databases*

General Terms

Algorithms, Performance

Keywords

Admission control, statistical modeling, disk performance, streaming media

*This research has been funded in part by NSF grants EEC-9529152 (IMSC ERC), and IIS-0082826, and unrestricted cash/equipment gifts from Intel and Hewlett-Packard.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
MM'03, November 2–8, 2003, Berkeley, California, USA.
Copyright 2003 ACM 1-58113-722-2/03/0011 ...$5.00.

1. INTRODUCTION

Magnetic disk drives are increasingly being used in many digital media repositories that traditionally have been the domain of tape storage. Streaming media servers that handle digital audio and video content are one example, "digital hub" devices in the living room (e.g., TiVo) are another, and finally TV studio and film production equipment are a third. Disk drives are very cost effective and the continued increase in storage space per unit exceeds even Moore's Law. There are two generally accepted paradigms to assign data blocks to the magnetic disk drives that form the storage system: in a *round-robin* sequence [2], or in a *random* manner [12]. Traditionally, the round-robin placement utilizes a cycle-based approach to scheduling of resources to guarantee the service quality, while the random placement utilizes a deadline-driven approach. The latter provides a number of advantages such as support for multiple or variable delivery rates with a single storage data block size, easy support for interactive applications, and support for data reorganization during storage system scaling. All these features may be supported with cycle-based scheduling, however, it results in a complex implementation and – most importantly – many of the disk parameters must be assumed with their worst case values. Therefore, deterministic guarantees are obtained at the expense of efficiency.

Deadline-driven scheduling can be configured to be both very efficient and to incur a very low probability of disruptions. The most important task is to limit the number of streams to achieve a user or application defined, low probability for missed deadlines. This task is performed by the admission control algorithm and it is therefore crucial in the overall systems design [11, 10, 4].

In this paper we propose a novel statistical admission control algorithm called TRAC (Three Random variable Admission Control) that models a much more comprehensive set of features of real time storage and retrieval than previous work. Our approach targets multi-stream architectures as opposed to personal video recorders (PVR, e.g., TiVo or ReplayTV) which are restricted to one recording and one playback stream. In such a setting disk bandwidth resources are plentiful, especially if the video is compressed to a few Mb/s. Therefore, admission control procedures are not usually necessary. This situation will change in the future when a PVR unit may manage multiple video and audio streams via FireWire, USB and wireless connections. Our admission control procedure is designed for these resource constrained and other large-scale, multi-stream systems [21]. Specifically, our TRAC algorithm enables:

i Support for variable bit rate (VBR) streams (Fig. 1a illustrates the variability of a sample MPEG-2 movie).

ii Support for concurrent reading and writing of streams. The distinguishing issue for a mixed workload is that disk drives generally provide less write than read bandwidth (see Fig. 1b). Therefore, the combined available bandwidth is a function of

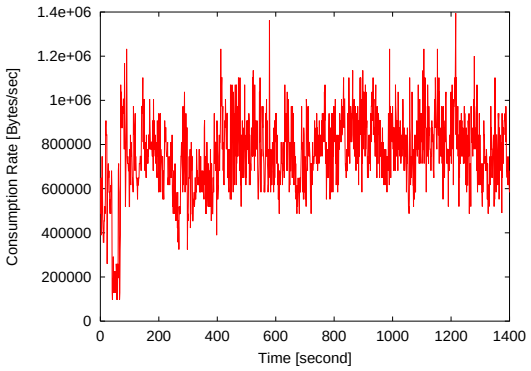

Fig. 1a: The consumption rate of a movie encoded with a VBR MPEG-2 algorithm ("Saving Private Ryan").

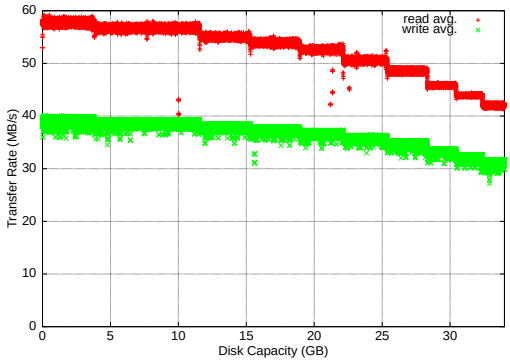

Fig. 1b: The read and write transfer rate profile of a Seagate Cheetah X15 disk drive.

Figure 1: Important modeling parameters that must be considered by the admission control algorithm.

the read/write mix. We propose a dynamic bandwidth sharing mechanism as part of the admission control.

iii Support for multi-zoned disks. Fig. 1b illustrates that the disk transfer rates of current generation drives is platter location dependent. The outermost zone provides up to 30% more bandwidth than the innermost one.

iv Modeling of the variable seek time and variable rotational latency that is naturally part of every data block read and write operation.

v Support for efficient random data placement [12].

To the best of our knowledge, no prior work has investigated such a comprehensive set of parameters. We feel that an integrated approach is essential when building large-scale, high performance real time storage systems and the preliminary evaluation results of the TRAC algorithm show that an increase in throughput of up to 38% may be achieved in retrieval only experiments.

The remainder of this paper is organized as follows. In Section 2 we review some of the related work. Section 3 describes our proposed TRAC algorithm and in Section 4 we present the results of our extensive performance evaluation through numerical analysis and real measurements. Finally, Section 5 concludes the paper and presents some future research directions of this work.

2. RELATED WORK

A number of studies have investigated admission control techniques in multimedia server designs. Fig. 2 classifies these techniques into two categories: *measurement-based* and *parameter-based*. The parameter-based approach can be further divided into *deterministic* and *statistical* algorithms.

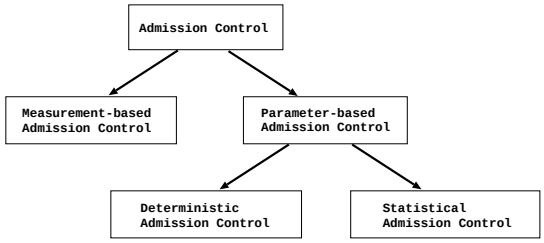

Figure 2: Taxonomy of different admission control algorithms.

With measurement-based algorithms [8, 1], the utilization of critical system resources is measured continually and the results are used in the admission control module. Measurement-based algorithms can only work online and cannot be used to offline configure a system or estimate its capacity. Furthermore, it is difficult to obtain an accurate estimation of dynamically changing system resources. For example, the time window during which the load is measured influences the result. A long time window smooths out load fluctuations but may overlap with several streams being started and stopped, while a short measurement interval may over or underestimate the current load. Deterministic admission control [13, 11, 10, 4] aims to provide guaranteed service, however it must assume the worst case for some of the system parameters and hence often under-utilizes available resources.

Statistical admission control has been studied in a number of papers [19, 3, 7, 14]. [19] exploits the variation in disk access times to media blocks as well as the VBR client load to provide statistical service guarantees for each client. Note that in [19], the distribution function for disk service time is obtained through exhaustive empirical measurements. [3] introduces three ways to estimate the disk overload probability while [7] proposes a probabilistic model that includes caching effects in the admission control. [14] introduces a stochastic model that considers VBR streams and the variable transfer rates of multi-zone disks.

Recently, the effects of user interaction on admission control has been studied [9, 5]. [9] proposed an optimization for the disk and cache utilization while reserving disk bandwidth for streams that are evicted from cache. [5] introduced a Continuous Time Markov Chains (CTMCs) model to predict the varying resource demands within an interactive session and incorporated it into the admission control algorithm.

Most of the previously proposed statistical admission control algorithms have adopted a very simple disk model. Only [14] considers the variable transfer rate of multi-zone disks. It differs from our TRAC algorithm in that (1) it assumes that all zones have the same number of tracks, (2) it did not consider the variance of the seek time, and (3) it is based on round-robin data placement and round-based disk scheduling. Additionally, no previous study has considered the difference in the disk transfer rate for reading and writing.

3. THE TRAC ALGORITHM

To address the shortcomings of the previous approaches we introduce a novel statistical admission control algorithm called TRAC.

Term	Definition	Units	Term	Definition	Units
B_{disk}	Block size on disk	MB	δ	Partition factor, the percentage of	
T_{svr}	Server observation time interval	second		disk bandwidth allocated for reading	
ξ	The number of disks in the system		n	The number of concurrent streams	
h	The maximum disk rotational latency	ms	n_{rs}	The number of retrieving streams	
$T_{seek}(i)$	Disk seek time for client i during T_{svr}	ms	n_{ws}	The number of recording streams	
$\widehat{R_{Dr}}$	Average disk bandwidth allocated for reading during a T_{svr}	MB/s	$D(i)$	The amount of data to read or write for client i during T_{svr}	MB
$\widehat{R_{Dw}}$	Average disk bandwidth allocated for writing during a T_{svr}	MB/s	p_{iodisk}	Probability of missed deadline by reading or writing	
$\overline{R_{Dr}}$	Average disk read bandwidth during T_{svr} (no bandwidth allocation for writing)	MB/s	p_{req}	The threshold of probability of missed deadline, it is the worse situation that client can endure.	
$\overline{R_{Dw}}$	Average disk write bandwidth during T_{svr} (no bandwidth allocation for reading)	MB/s	ε_{WRBS}	Percentage of disk bandwidth reserved for writing	
			ε_{RRBS}	Percentage of disk bandwidth reserved for reading	
$\overline{R_{Dio}}$	Average combined disk bandwidth during a T_{svr}	MB/s	μ_i	Mean value of random variable $D(i)$	MB
R_{Dr}	Maximum disk read bandwidth during T_{svr} (no bandwidth allocation for writing)	MB/s	σ_i	Standard deviation of random variable $D(i)$	MB
			m	The number of seeks during a T_{svr}	
R_{Dw}	Maximum disk write bandwidth during T_{svr} (no bandwidth allocation for reading)	MB/s	$t_{seek}(j)$	Seek time for disk access j, where j is an index for each disk access during a T_{svr}	ms
$R_{Dr}(j)$	Disk read bandwidth for disk access j (no bandwidth allocation for writing) where j is an index for each disk access during a T_{svr}	MB/s	$\mu_{t_{seek}}(j)$	Mean value of random variable $t_{seek}(j)$	ms
			$\sigma_{t_{seek}}(j)$	Standard deviation of random variable $t_{seek}(j)$	ms
			S_j	Seek distance for disk access j during a T_{svr}	
			U_j	Rotational latency for disk access j during a T_{svr}	ms
r_{Dr}	Current used disk read bandwidth	MB/s	β	Relationship factor between R_{Dr} and R_{Dw}	
r_{Dw}	Current used disk write bandwidth	MB/s	$\overline{t_{seek}}$	The average disk seek time during T_{svr}	ms
β_k	Ratio between write and read bandwidth for zone k		$\mu_{t_{seek}}$	Mean value of random variable $\overline{t_{seek}}$	ms
			$\sigma_{t_{seek}}$	Standard deviation of random variable $\overline{t_{seek}}$	ms
α	Mixed-load factor, the percentage of reading load in the system		a_1, b_1, a_2, b_2, r	Disk seek time modeling parameters	
			w, v_i, k_i	Disk transfer rate modeling parameters	

Table 1: List of terms used repeatedly in this study and their respective definitions.

We start by describing the algorithm in a single disk environment and then extend it to a multi-disk environment in Section 3.3. Table 1 lists all the parameters and their definitions used in this paper.

3.1 Three Random Variable (3RV) Model

Consider the following scenario. The system is servicing n variable bit rate clients using deadline-driven scheduling and data blocks are allocated to a disk using a random placement policy. The server activity is observed periodically, during a time interval T_{svr}. Hence, our model is characterized by three random variables: (1) $D(i)$ denotes the amount of data to be retrieved or recorded for client i during observation window T_{svr}, (2) $\overline{R_{Dr}}$ denotes the average disk read bandwidth during T_{svr} with no bandwidth allocation to writing, and (3) $\overline{t_{seek}}$ denotes the average disk seek time during each observation time interval T_{svr}.

Let $T_{seek}(i)$ denote the disk seek time for client i during T_{svr}[1]. Let n_{rs} and n_{ws} denote the number of retrieval and recording streams served respectively, i.e., $n = n_{rs} + n_{ws}$. Also, $\widehat{R_{Dw}}$ represents the average disk bandwidth (in MB/s) allocated for writing during T_{svr}, while $\widehat{R_{Dr}}$ represents the average bandwidth for reading. With such a mixed load of both retrieving and recording clients, the average combined disk bandwidth $\overline{R_{Dio}}$ is constrained by $\overline{R_{Dio}} = \widehat{R_{Dr}} + \widehat{R_{Dw}}$. Consequently, the maximum amount of data that can be read and written during each interval T_{svr} can be expressed by $\overline{R_{Dio}} \times \left(T_{svr} - \sum_{i=1}^{n_{rs}+n_{ws}} T_{seek}(i)\right)$. Furthermore, if $\sum_{i=1}^{n} D(i)$ represents the total read and write bandwidth requirement during T_{svr} from all streams n, then the probability of missed deadlines, p_{iodisk}, can be computed by Eq. 1.

$$p_{iodisk} = P\left[\sum_{i=1}^{n} D(i) > \left(\overline{R_{Dio}} \times \left(T_{svr} - \sum_{i=1}^{n} T_{seek}(i)\right)\right)\right] \quad (1)$$

Note that a missed deadline of a disk access does not necessarily cause a hiccup for the affected stream because data buffering may hide the delay. However, we consider the worst case scenario for our computations.

Recall that $\sum_{i=1}^{n} T_{seek}(i)$ denotes the total seek time spent for all n clients during T_{svr}. Let $t_{seek}(j)$ denote the seek time for disk access j, where j is an index for each disk access during T_{svr}. Thus, the total seek time can be computed as follows

$$\sum_{i=1}^{n} T_{seek}(i) = \sum_{j=1}^{m} t_{seek}(j) = m \times \overline{t_{seek}} \quad (2)$$

where m denotes the number of seeks and $\overline{t_{seek}}$ is the average seek time, both during T_{svr}. Because every seek operation is followed by a data block read or write, m can also be expressed by $m = \frac{\sum_{i=1}^{n} D(i)}{B_{disk}}$, where B_{disk} is the block size. With the appropriate substitutions we arrive at our final expression for the probability of over-committing the disk bandwidth, which may translate into missed I/O deadlines.

$$p_{iodisk} = P\left[\sum_{i=1}^{n} D(i) > \left(\frac{\overline{R_{Dio}} \times T_{svr}}{1 + \frac{\overline{t_{seek}} \times \overline{R_{Dio}}}{B_{disk}}}\right)\right] \leq p_{req} \quad (3)$$

Before we can proceed to evaluate Eq. 3 we need to focus our attention on the random variable $\overline{R_{Dio}}$ because of its interesting properties.

3.1.1 Dynamic Disk Bandwidth Sharing

Fig. 1b shows the measured disk transfer rate for reading and writing with a modern multi-zone disk drive. Let R_{Dr} denote the maximum disk read bandwidth without any bandwidth allocation for writing. Conversely, R_{Dw} denotes the maximum disk write bandwidth. Let $\overline{R_{Dr}}$ denote the *average* disk read bandwidth during T_{svr}. Similarly, $\overline{R_{Dw}}$ denotes the average disk write bandwidth. We observe that R_{Dr} is much higher than R_{Dw} and conclude that 1 MB/s of read bandwidth is *not* interchangeable with 1 MB/s of write bandwidth. Fig. 3 illustrates how the average combined bandwidth $\overline{R_{Dio}}$ changes depending on the mix of read versus write allocation. For our further discussion we introduce a par-

[1] $T_{seek}(i)$ includes rotational latency as well.

tition factor δ that defines the percentage of the disk bandwidth allocated for reading.

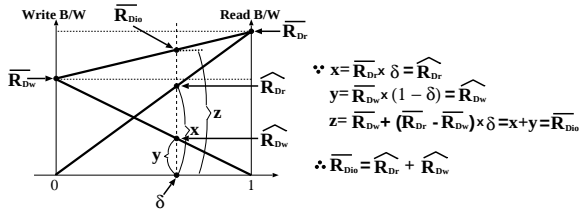

Figure 3: Relationships between the average read bandwidth $\overline{R_{Dr}}$, the average write bandwidth $\overline{R_{Dw}}$, the average combined bandwidth $\overline{R_{Dio}}$ and the disk bandwidth partition factor δ.

We can formulate the disk bandwidth sharing problem of how to effectively partition the disk bandwidth for reading and writing while maximizing resource utilization. We identify the following desirable **Design Goals** for the admission control algorithm:

DG1: Share the total disk bandwidth between read and write requests.

DG2: Dynamically allocate the available disk bandwidth to read or write requests on demand.

DG3: Support multiple bandwidth sharing policies (see definitions below).

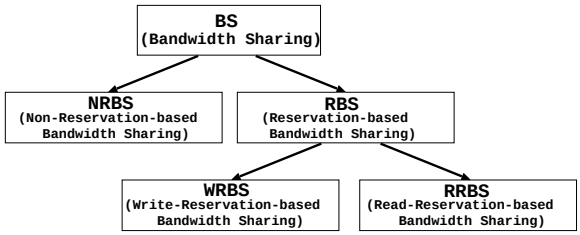

Figure 4: Taxonomy of different bandwidth sharing policies.

We introduce three bandwidth sharing policies, illustrated in Fig. 4, with the following properties:

DEFINITION 3.1**.** *The* Non-Reservation-based Bandwidth Sharing (NRBS) *policy is defined as: disk reading and writing requests are served with no preference, i.e., no bandwidth reservation for either reading or writing exists.* ∎

DEFINITION 3.2**.** *The* Reservation-based Bandwidth Sharing (RBS) *policy is defined as: a fraction of the disk bandwidth is reserved for disk reading or writing. When disk bandwidth is reserved for writing, it is termed* Write-Reservation-based Bandwidth Sharing policy (WRBS). *When disk bandwidth is reserved for reading, it is termed* Read-Reservation-based Bandwidth Sharing policy (RRBS). ∎

Fig. 3 illustrates all the possible configurations to partition the disk bandwidth: as δ moves from 0 to 1, more bandwidth is allocated for reading and $\overline{R_{Dio}}$ varies from $\overline{R_{Dw}}$ to $\overline{R_{Dr}}$. Thus, $\overline{R_{Dio}}$ can be expressed as:

$$\overline{R_{Dio}} = \delta\overline{R_{Dr}} + (1-\delta)\overline{R_{Dw}} \tag{4}$$

As suggested in Fig. 1b, $\overline{R_{Dr}}$ and $\overline{R_{Dw}}$ are two random variables. We model the relationship between the average read and the average write bandwidth with the parameter $\beta = \frac{\overline{R_{Dw}}}{\overline{R_{Dr}}}$, which can be obtained experimentally from disk profiling (see Section 4). For multi-zone disks, the ratio between write and read bandwidth will differ from zone to zone. Let β_k denote the ratio between write and read bandwidth for zone k. For example, for a Seagate Cheetah X15 disk, $\beta_0 = 0.668$, $\beta_1 = 0.679$, and $\beta_2 = 0.673$ (see Table 2). For this disk model, β_k varies about 13% ($\beta_k \in [0.668, 0.757]$) across all zones. We can prove that $\beta \in [\beta_{k_{min}}, \beta_{k_{max}}] = [0.668, 0.757]$ (see [20] for proof details), where $\beta_{k_{min}}$ and $\beta_{k_{max}}$ are the minimum and maximum values among all β_k respectively, and $k \in [1, w]$ and w is the total number of zones. It is generally true that β does not change much. Based on the *strong law of large numbers*, $\lim_{n\to\infty} \frac{\overline{R_{Dw}}}{\overline{R_{Dr}}} = \frac{\mu_{\overline{R_{Dw}}}}{\mu_{\overline{R_{Dr}}}}$

To simplify our model, we use the limit value as β in our further calculations (see Section 3.2.1). Therefore, Eq. 4 can be rewritten as:

$$\overline{R_{Dio}} = \delta\overline{R_{Dr}} + (1-\delta)\beta\overline{R_{Dr}} \tag{5}$$

To satisfy **DG1** and **DG2**, the partition factor δ must be dynamically adjusted according to the system conditions, i.e., the ratio between the read and write load. This behavior is modeled by the mixed-load factor α

$$\alpha = \frac{r_{Dr}}{r_{Dr} + \frac{r_{Dw}}{\beta}} \tag{6}$$

where r_{Dr} and r_{Dw} denote the current disk read and write bandwidth, respectively. For example, with $\alpha = 1$ only reading clients exist in the system. On the other hand, $\alpha = 0$ implies only recording clients are in the system. Next, we must compute δ under disk read load r_{Dr} and write load r_{Dw} using different bandwidth sharing policies, assuming that $r_{Dr} + r_{Dw} > 0$. We conjecture the following theorem.

THEOREM 3.3**.** *To satisfy the design goals **DG1**, **DG2**, and **DG3**,* $\delta = \alpha = \frac{r_{Dr}}{r_{Dr} + \frac{r_{Dw}}{\beta}}$.

We have omitted the proof here; for details see [20].

Different bandwidth sharing policies might be adopted for different applications. We first focus on NRBS to simplify the discussion. Based on *Theorem* 3.3 and Eq. 5, the disk bandwidth with a mixed-load factor α can be expressed as

$$\overline{R_{Dio}} = \alpha\overline{R_{Dr}} + (1-\alpha)\beta\overline{R_{Dr}} \tag{7}$$

This equation considers two extreme cases as well: (1) when there is only read load, i.e., $\alpha = 1$, $\overline{R_{Dio}} = \overline{R_{Dr}}$, and (2) when there is only write load, i.e., $\alpha = 0$, $\overline{R_{Dio}} = \beta\overline{R_{Dr}} = \overline{R_{Dw}}$. Based on Eq. 7, Eq. 3 can be further generalized to

$$\begin{aligned} & p_{iodisk} \\ &= P\left[\sum_{i=1}^{n} D(i) > \left(\frac{(\alpha\overline{R_{Dr}} + (1-\alpha)\beta\overline{R_{Dr}}) \times T_{svr}}{1 + \frac{\overline{t_{seek}} \times (\alpha\overline{R_{Dr}} + (1-\alpha)\beta\overline{R_{Dr}})}{B_{disk}}}\right)\right] \\ &\leq p_{req} \end{aligned} \tag{8}$$

where α can be approximated with Eq. 9, in which μ_i denotes the mean value of random variable $D(i)$.

$$\alpha \approx \frac{\sum_{i=1}^{n_{rs}} \mu_i}{\sum_{i=1}^{n_{rs}} \mu_i + \frac{\sum_{i=1}^{n_{ws}} \mu_i}{\beta}} \tag{9}$$

3.1.2 Probability Evaluation

We now have all the tools necessary to evaluate the probability of a possible disk bandwidth overcommitment. Let X, Y and Z denote $\sum_{i=1}^{n} D(i)$, $\overline{t_{seek}}$ and $\overline{R_{Dr}}$, respectively. The probability

p_{iodisk} in Eq. 8 can then be evaluated as follows

$$\begin{aligned} p_{iodisk} &= P\left[(X,Y,Z) \in \Re\right] \\ &= \iiint_{\Re} f_{XYZ}(x,y,z)dxdydz \end{aligned} \tag{10}$$

where

$$\Re = \left\{ (X,Y,Z) \mid X > \left(\frac{(\alpha Z + (1-\alpha)\beta Z) \times T_{svr}}{1 + \frac{Y \times (\alpha Z + (1-\alpha)\beta Z)}{B_{disk}}} \right) \right\} \tag{11}$$

and $f_{XYZ}(x,y,z)$ is the joint probability density function (pdf) of X,Y,Z. Fig. 5 shows the integration region $\Re$ of Eq. 11 with

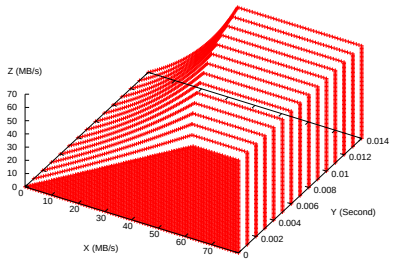

Figure 5: Example integration region $\Re$ for T_{svr} = 1 second, $B_{disk} = 1$ MB, $\alpha = 0.5$, and $\beta = 0.8$.

T_{svr} = 1 second, $B_{disk} = 1$ MB, $\alpha = 0.5$, and $\beta = 0.8$. Note that this figure only shows a small portion of the 3D space, where $0 < X < 80$ MB/s, $0 < Y < 14$ ms and $0 < Z < 70$ MB/s, which covers the operation parameters for most modern disk drives. Since the three random variables $\sum_{i=1}^{n} D(i)$, $\overline{t_{seek}}$ and $\overline{R_{Dr}}$ are independent, we obtain

$$f_{XYZ}(x,y,z) = f_X(x) f_Y(y) f_Z(z) \tag{12}$$

where $f_X(x)$,$f_Y(y)$ and $f_Z(z)$ are the pdf of X,Y and Z, respectively. Next, we will present an overview of how to derive $f_X(x)$, $f_Y(y)$, and $f_Z(z)$, and then continue with the actual admission control procedure.

3.1.3 Determination of $f_X(x)$: pdf of $\sum_{i=1}^{n} D(i)$

Recall that $D(i)$ denotes the amount of data that client i reads or writes during T_{svr}. Since $D(i)$ is only dependent on the stream bandwidth characteristics of each client, $D(1) \cdots D(n)$ are independent random variables. According to the *central limit theorem*, $\sum_{i=1}^{n} D(i)$ approaches a normal distribution [15] with mean $\sum_{i=1}^{n} \mu_i$ and variance $\sum_{i=1}^{n} \sigma_i^2$, where μ_i and σ_i^2 denote the mean value and variance of $D(i)$, respectively[2]. Therefore, we obtain the pdf of $\sum_{i=1}^{n} D(i)$ as:

$$f_X(x) = \frac{1}{\sqrt{2\pi \sum_{i=1}^{n} \sigma_i^2}} e^{-\frac{\left[x - \sum_{i=1}^{n} \mu_i\right]^2}{2 \times \sum_{i=1}^{n} \sigma_i^2}} \tag{13}$$

[2]It is relatively easy to obtain μ_i and σ_i^2 for a stream that is already stored on the server. For a new stream to be recorded–especially a live event–we have to rely on estimates. However, since such live streams usually use encoders or compressors at the source, it is often possible to obtain good estimates for μ_i and σ_i^2 from the configuration parameters of the corresponding encoder and compressor.

3.1.4 Determination of $f_Y(y)$: pdf of $\overline{t_{seek}}$

From Eq. 2 we obtained $\overline{t_{seek}} = \frac{\sum_{j=1}^{m} t_{seek}(j)}{m}$, which suggests that $\overline{t_{seek}}$ is dependent on m random variables $t_{seek}(j)$, with $j \in [1,m]$. Due to the random data placement, these m random variables are independently and identically distributed with mean value $\mu_{t_{seek}}(j)$ and variance $\sigma^2_{t_{seek}}(j)$. Assuming $m > 30$, by the central limit theorem $\overline{t_{seek}}$ also has a normal distribution with mean $\mu_{t_{seek}}(j)$ and variance $\frac{\sigma^2_{t_{seek}}(j)}{m}$. Recall that $m = \frac{\sum_{i=1}^{n} D(i)}{B_{disk}}$, since $\sum_{i=1}^{n} D(i)$ has a normal distribution and B_{disk} is a constant; furthermore m is normally distributed with mean $\frac{\sum_{i=1}^{n} \mu_i}{B_{disk}}$ and variance $\frac{\sum_{i=1}^{n} \sigma_i^2}{B_{disk}^2}$. To simplify the model, we approximate m with its mean value in later derivations. Thus, we obtain the pdf of $\overline{t_{seek}}$ as:

$$f_Y(y) \approx \frac{1}{\sqrt{2\pi \sigma^2_{t_{seek}}(j)}} e^{-\frac{\sum_{i=1}^{n} \mu_i}{2B_{disk}} \left[\frac{y - \mu_{t_{seek}}(j)}{\sigma_{t_{seek}}(j)}\right]^2} \tag{14}$$

3.1.4.1 Determination of $\mu_{t_{seek}}(j)$ and $\sigma_{t_{seek}}(j)$.

Let U_j denote the rotational latency for disk access j and let S_j denote the percentage value (between 0 and 100) of the total disk storage capacity. We consider the rotational latency part of the seek time $t_{seek}(j)$. We express the relationship among $t_{seek}(j)$, S_j and U_j through disk profiling and modeling [17] as

$$t_{seek}(j) = \begin{cases} a_1 + b_1\sqrt{S_j} + U_j & \text{if } 0 \le S_j \le r \\ a_2 + b_2 S_j + U_j & \text{if } r < S_j \le 100 \end{cases} \tag{15}$$

where a_1, b_1, a_2, b_2 and r are the disk seek modeling parameters. Because of the random data placement, both S_j and U_j follow uniform distributions with pdfs $f_{S_j}(s) = \frac{1}{100}(s \in [0,100])$ and $f_{U_j}(u) = \frac{1}{h}(u \in [0,h])$, where h denotes the maximum rotational latency. Then, we can derive the pdf of $t_{seek}(j)$ and compute $\mu_{t_{seek}}(j)$ and $\sigma_{t_{seek}}(j)$ (extensive details are contained in [20]). Figure 6(a) shows the seek time profile of a Seagate Cheetah X15 disk, which has the following parameters: $a_1 = 1$, $b_1 = 0.6$, $a_2 = 2.1$, $b_2 = 0.05$, $r = 5$, and $h = 4$ ms. Fig. 6(b) shows a good match of the derived pdf of $t_{seek}(j)$ with the empirically measured relative frequency histogram. Using the pdf of $t_{seek}(j)$, we obtain $\mu_{t_{seek}}(j) = 6.62$ ms, $\sigma_{t_{seek}}(j) = 1.85$ ms.

3.1.5 Determination of $f_Z(z)$: pdf of $\overline{R_{Dr}}$

Let $R_{Dr}(j)$ denote the disk read bandwidth for disk access j during T_{svr}. Then, the average read bandwidth $\overline{R_{Dr}}$ can be computed as $\overline{R_{Dr}} = \frac{\sum_{j=1}^{m} R_{Dr}(j)}{m}$, where these m random variables $R_{Dr}(j)$ are independently and identically distributed with mean value $\mu_{R_{Dr}}(j)$ and variance $\sigma^2_{R_{Dr}}(j)$. Following similar reasoning as in Section 3.1.4, $\overline{R_{Dr}}$ also approaches a normal distribution with pdf

$$f_Z(z) \approx \frac{1}{\sqrt{2\pi \sigma^2_{R_{Dr}}(j)}} e^{-\frac{\sum_{i=1}^{n} \mu_i}{2B_{disk}} \left[\frac{z - \mu_{R_{Dr}}(j)}{\sigma_{R_{Dr}}(j)}\right]^2} \tag{16}$$

Next, we will describe how to obtain $\mu_{R_{Dr}}(j)$ and $\sigma_{R_{Dr}}(j)$.

3.1.5.1 Determination of $\mu_{R_{Dr}}(j)$ and $\sigma_{R_{Dr}}(j)$.

Most magnetic disk drives feature variable transfer rates due to a technique called zone-bit recording (ZBR), which increases the amount of data being stored on a track as a function of its distance from the disk spindle. We model the variable zone transfer rates with $R_{Dr}(j)$. Let L denote the starting location of each disk access during T_{svr}. L can be quantified using the percentage value of the

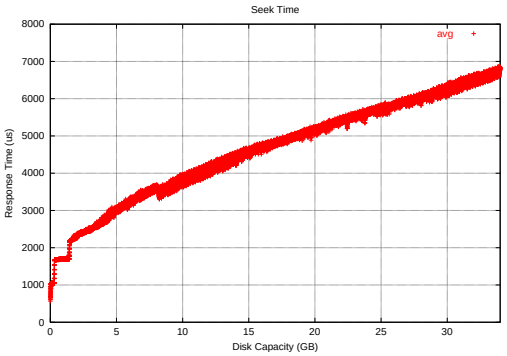

Fig. 6a: Seek time profile
(Note: rotational latency is not included)

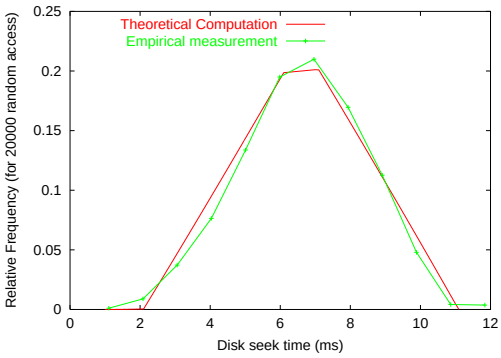

Fig. 6b: Probability density function $f_{t_{seek}(j)}(t)$ of $t_{seek}(j)$

Figure 6: Determination of $\mu_{t_{seek}}(j)$ and $\sigma_{t_{seek}}(j)$ for a Seagate Cheetah X15 disk drive.

total disk capacity, i.e., $L \in [0, 100]$. From the disk transfer rate profile (see Fig. 1b), the relationship between $R_{Dr}(j)$ and L is modeled as

$$R_{Dr}(j) = \begin{cases} v_1 & \text{if } 0 \leq L \leq k_1 \ (L \in \text{Zone 1}) \\ \vdots & \vdots \\ v_w & \text{if } k_{w-1} < L \leq k_w \ (L \in \text{Zone w}) \end{cases} \quad (17)$$

where w is the number of zones, and v_i and k_i model the multi-zone characteristics, where $i \in [1, w]$, $v_1 > \cdots > v_w$, $0 < k_1 < \cdots < k_w = 100$, and $[0, k_1]$, $[k_1, k_2]$, $\cdots$, $[k_{i-1}, k_i]$, $\cdots$, $[k_{w-1}, k_w]$ represent zones 1, 2, $\cdots$, i, $\cdots$, w, respectively. These w, v_i and k_i are termed disk transfer rate modeling parameters. Because of the random data placement, L is uniformly distributed with pdf $f_L(l) = \frac{1}{100}(l \in [0, 100])$. Consequently, with the introduction of Dirac delta functions [18] we can derive the pdf $f_{R_{Dr}}(r)$ (see [20] for details) as shown in Eq. 18.

$$f_{R_{Dr}}(r) = \sum_{i=1}^{w} \frac{k_i - k_{i-1}}{100} \delta(r - v_i) \quad (18)$$

Using the pdf of $R_{Dr}(j)$, we can obtain $\mu_{R_{Dr}}(j)$ and $\sigma_{R_{Dr}}(j)$. For example, for a Seagate Cheetah X15 disk, $\mu_{R_{Dr}}(j) = 52.26$ MB/s and $\sigma_{R_{Dr}}(j) = 5.33$ MB/s. The final step is to apply $\mu_{R_{Dr}}(j)$ and $\sigma_{R_{Dr}}(j)$ to Eq. 16 resulting in the pdf for $\overline{R_{Dr}}$.

We have now obtained all the necessary components to evaluate p_{iodisk}, the probability of cover-committing the disk bandwidth.

3.2 Admission Control Procedure with TRAC

After having presented the models to calculate the probability for missed deadlines, we will now outline how to incorporate them into a complete admission control procedure. The different bandwidth sharing policies will all be supported.

3.2.1 *Admission Control with the NRBS Policy*

Fig. 7 shows the admission control procedure as a flow chart. It can be divided into two components: the disk modeling module (steps 1 through 7) and the admission decision module (steps 8 through 15). The disk modeling component provides the parameters that describe the disk characteristics. It needs to be evaluated only when new disks are introduced into the system. Steps 1 and 2 model the maximum rotational latency h and the disk seek profile parameters a_1, b_1, a_2, b_2, r necessary for Eq. 15 [17]. With the model of Section 3.1.4.1 we obtain $\mu_{t_{seek}}(j)$ and $\sigma_{t_{seek}}(j)$ in step 3. Step 4 determines the optimal disk block size B_{disk}. At step 5, the disk transfer rate is profiled and the results produce the R_{Dr} modeling parameters $w, v_1, \ldots, v_w$ and $k_1, \ldots, k_w$ (see Eq. 17). The mean and variance of $R_{Dr}(j)$ is computed as described in Section 3.1.5.1 (step 6). Next, we obtain the fractional

Zone #	Size (MB)	Read Transfer Rate (MB/s)	Write Transfer Rate (MB/s)	β_k [a]	Start (MB)	End (MB)
0	12,000	57.5	38.4	0.668	0	12,288
1	3,500	55.4	37.6	0.679	12,289	15,872
2	3,000	54.7	36.8	0.673	15,873	18,944
3	4,000	52.7	36.2	0.687	18,945	23,040
4	3,000	50.6	35.3	0.698	23,041	26,112
5	2,500	48.1	34.5	0.717	26,113	28,672
6	3,000	45.6	33.1	0.726	28,673	31,744
7	2,500	43.6	32.2	0.739	31,745	34,304
8	2,500	41.9	31.7	0.757	34,305	36,864

[a] Ratio between write and read data transfer rate of a zone.

Table 2: Zoning information of a Seagate Cheetah X15 (model ST336752LC) disk.

factor β between $\overline{R_{Dw}}$ and $\overline{R_{Dr}}$ in step 7. To illustrate, assume the disk transfer rate profile of a Seagate Cheetah X15 disk as shown in Fig. 1b. Table 2 lists the β_k value for each of the disk's zones. For the Cheetah X15 disk drive β_k varies slightly between 0.668 and 0.757 (see Table 2). Recall that β is also bounded by the same range as β_k. Follow a similar derivation in section 3.1.5, we can obtain $\mu_{\overline{R_{Dw}}}$ with $\mu_{\overline{R_{Dw}}} = \mu_{R_{Dw}}(j)$, where $\mu_{R_{Dw}}(j)$ is the disk write bandwidth for disk access j during T_{svr} with no bandwidth allocation for reading. To simplify the model, we approximate β by its limit $\frac{\mu_{\overline{R_{Dw}}}}{\mu_{\overline{R_{Dr}}}}$. For example, the value for the Cheetah X15 is $\mu_\beta = 0.690337$. Note that these disk modeling procedures only need to be evaluated once for each disk drive. This could either be done offline (before the system is started) or online (when a new disk is introduced into a streaming server while it is running).

The admission decision module, on the other hand, is evaluated for every new stream request. It comprises of steps 8 through 15 shown in Fig. 7. At step 8, a new client request is received. If the request is for recording a stream, then estimates of the mean μ_i and variance σ_i^2 of the new VBR stream need to be provided. These values will be stored in a repository if the stream is admitted. Hence, for a retrieval stream request, μ_i and σ_i^2 are simply retrieved from the internal database. Step 9 implements the different bandwidth sharing policies. It is a pass-through in case of NRBS. At step 10, mix-load factor α is updated according to Eq. 9. Next, the pdfs of $\overline{t_{seek}}$, $\sum_{i=1}^{n} D(i)$ and $\overline{R_{Dr}}$ are computed at steps 11, 12 and 13, respectively, as outlined in Sections 3.1.4, 3.1.3 and 3.1.5. Note that steps 10, 11, 12 and 13 make use of the current session status

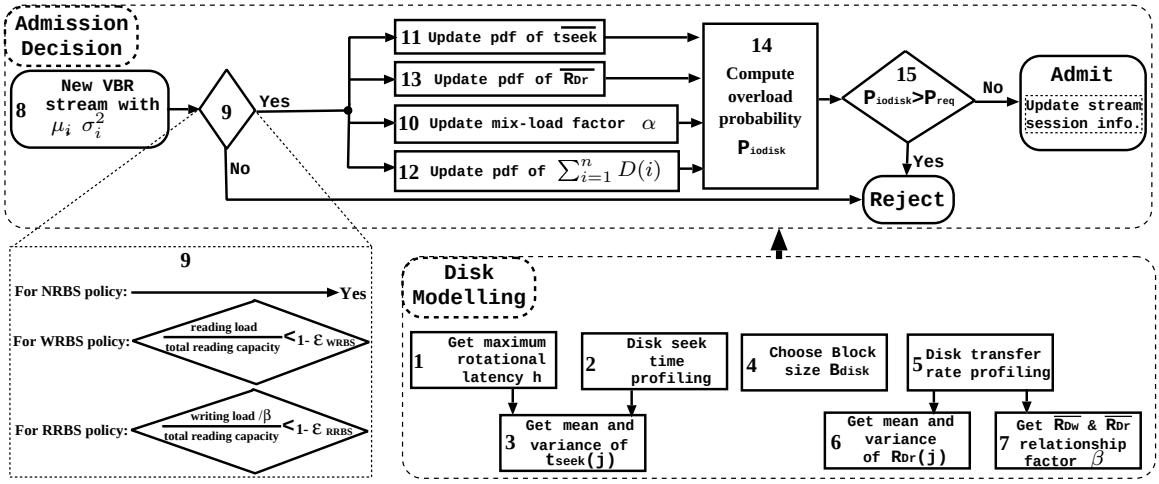

Figure 7: The admission control procedure with the three random variable model. It supports three different bandwidth sharing policies (step 9).

information. Finally, the disk overload probability p_{iodisk} is computed via Eq. 10 (see Section 3.1). The admission decision is made in step 14 by comparing the probability p_{iodisk} with the user provided threshold p_{req}. If $p_{iodisk} > p_{req}$, then the stream request is rejected. Otherwise, the client request is admitted and its information is added to the internal database.

3.2.2 Admission Control with Preferred Reading or Writing: RBS Policy

Step 9 in Fig. 7 can be used to give preferential treatment to either retrieval or recording requests. We term these Reservation Based Sharing (RBS) policies (see Definition 3.2 for details). With Write-RBS (WRBS), ε_{WRBS} denotes the fraction of a disk's R/W capacity that must be reserved for writing. In other words, the maximum read load should not exceed $1 - \varepsilon_{WRBS}$. Hence, at step 9 the current system read load plus the new stream are compared with this limit. The evaluation can be approximated by $\frac{\text{read load}}{\text{total read capacity}} \approx \frac{\sum_{i=1}^{n_{rs}} \mu_i}{\mu_{R_{Dr}}}$ where $\mu_{R_{Dr}}$ is the mean value of the random variable $R_{Dr}(j)$ (see Section 3.1.5.1) and μ_i ($i \in [1, n_{rs}]$) denote the mean value for each VBR retrieval stream. If the threshold is not exceeded then the stream is further evaluated equivalently to the non-reservation NRBS policy. Otherwise, the request is rejected.

Analogous, with Read-RBS (RRBS), ε_{RRBS} denotes the portion of a disk's R/W capacity reserved for reading, which means the maximum writing load should not exceed the upper bound $1 - \varepsilon_{RRBS}$. Therefore, the fraction of the write load is evaluated and compared with this limit. Like in the WRBS, the evaluation can be approximated by $\frac{\text{writing load}/\beta}{\text{total reading capacity}} \approx \frac{\sum_{i=1}^{n_{ws}} \mu_i/\beta}{\mu_{R_{Dr}}}$ where $\mu_{R_{Dr}}$ is the mean value of the random variable $R_{Dr}(j)$ and μ_i ($i \in [1, n_{ws}]$) denote the mean value for each VBR recording stream.

3.3 Disk Load Modeling for Multiple Homogeneous Disks

In the previous sections we assumed that only one disk existed in the system. However, the extension of our model for multiple disks is straightforward. If we denote the number of disks in the system with ξ then the R/W resource requirement to support n streams is scaled by a factor of $\frac{1}{\xi}$ on average for each individual disk. As stated previously, $\sum_{i=1}^{n} D(i)$ denotes the total R/W resource requirements during T_{svr}. Thus, the average amount of data to be stored to or retrieved from each disk during T_{svr} can be computed as $\lambda_{avg} = \frac{\sum_{i=1}^{n} D(i)}{\xi}$. Assuming a random data placement across the ξ disks, the R/W resource requirement λ_i for disk i during T_{svr} can be approximated with $\lambda_i \approx \lambda_{avg} = \frac{\sum_{j=1}^{n} D(j)}{\xi}$, where $i \in [1, \xi]$. Recall that $\sum_{i=1}^{n} D(i)$ follows a normal distribution (see Section 3.1.3) and since ξ is constant, λ_i also approaches a normal distribution with mean value $\mu_{\lambda_i} = \frac{\sum_{i=1}^{n} \mu_i}{\xi}$ and variance $\sigma^2_{\lambda_i} = \frac{\sum_{i=1}^{n} \sigma_i^2}{\xi^2}$. Consequently, following analogous reasoning as in the single disk case, the admission control criteria is modified to:

$$p_{iodisk} = P\left[\frac{\sum_{i=1}^{n} D(i)}{\xi} > \left(\frac{(\alpha\overline{R_{Dr}} + (1-\alpha)\beta\overline{R_{Dr}}) \times T_{svr}}{1 + \frac{\overline{t_{seek}} \times (\alpha R_{Dr} + (1-\alpha)\beta R_{Dr})}{B_{disk}}}\right)\right] \leq p_{req} \quad (19)$$

Note that the probability density functions need to be updated to reflect the decreased mean and variance values for the load on each disk. Furthermore, the number of seek operations will also be approximately evenly distributed across all disks. The detailed analysis is contained in [20].

4. PERFORMANCE EVALUATION

To evaluate the effectiveness of our TRAC algorithm we performed extensive comparisons between our analytical models and an actual system implementation. Additionally, we compared a previously proposed, single random variable admission control algorithm [3, 19] with our results. We termed this baseline algorithm 1RV-AC since it only considers the disk workload variability. There exists an approach that considers both the variability of the disk service time and the workload [19]. However, since the distribution function for the service time is obtained through exhaustive empirical measurements (and therefore difficult to reproduce), we felt that this method does not lend itself to a good analytical comparison. We expect that its performance would fall somewhere in between the 1RV-AC and TRAC algorithms. The performance measure used was the probability for missed deadlines p_{iodisk}. This probability directly translates into how many streams can be supported with a given miss-threshold.

4.1 Single Disk System Experiments

4.1.1 Experimental Setup

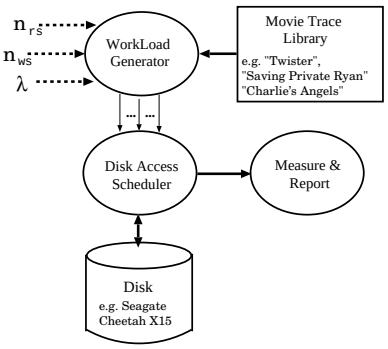

Figure 8: Experimental system setup.

The hardware platform used for our experiments was a Dell PowerEdge 1650 server with a Pentium III 1 GHz CPU, 256 MB of main memory and running RedHat Linux 7.0. This configuration can at the present time be considered midrange. We wanted to ensure that our measured results would be representative of a reasonable hardware configuration. Fig. 8 illustrates the structure of our experimental setup. Its five components are: a WorkLoad Generator, a Movie Trace Library, a Disk Access Scheduler, a Measure & Report module and a Disk. Note that we did not run a full fledged streaming server on our test system to reduce the number of parameters that would influence the results. As such, the results represent the best case scenario and other system bottlenecks – if present – might reduce the performance. The WorkLoad Generator produces stream requests based on a Poisson process with mean inter-arrival time of $\lambda = 5$ seconds. Each admitted stream produces data block requests with associated read/write deadlines according to movie bandwidth traces from the Movie Trace Library. The movie blocks are randomly placed onto disks and block requests are scheduled based on the deadline assigned to each block. Hence, the block with the earliest deadline is accessed first. The block requests are forward to the Disk by the Disk Access Scheduler at the set times. The Measure & Report module monitors the disk system performance and generates the result output. In this report, both the number of missed deadline requests and the total number of disk block requests are collected. Furthermore, the ratio between these two numbers, which represents the fraction of the missed deadline requests, is interpreted as the probability of missed deadlines p_{iodisk}.

The WorkLoad Generator has several configurable parameters: the mean inter-arrival time λ, the number of retrieval streams n_{rs} and the number of recording streams n_{ws}. We start with single media type experiments. Hence, the relationship between n_{rs}, n_{ws} and the mixed-load factor α can be expressed as $\alpha = \frac{n_{rs}}{n_{rs} + \frac{n_{ws}}{\beta}}$.

Table 3 summarizes the parameters used in the experiments and analysis and also lists the three movie traces: the DVD movie "Twister", the DVD movie "Saving Private Ryan" and the VCD movie "Charlie's Angels." The rate profile of "Saving Private Ryan" is shown in Fig. 1a as an example. The disk is a Seagate Cheetah X15 (Model ST336752LC). Fig. 6a shows the measured seek profile for the X15, while Fig. 1b illustrates the data transfer rate profile for both reading and writing, with reading being significantly faster than writing. Recall that β_k denotes the ratio between the writing and reading rate for zone k and Table 2 shows that this ratio is not constant, but varies between 0.668 and 0.757. We selected

Parameters	Configurations
Test movie "Twister"	MPEG-2 video, AC-3 audio
Average bandwidth	698594 Bytes/sec
Length	50 minutes
Throughput std. dev.	140456.8
Test movie "Saving Private Ryan"	MPEG-2 video, AC-3 audio
Average bandwidth	757258 Bytes/sec
Length	50 minutes
Throughput std. dev.	169743.6
Test movie "Charlie's Angels"	MPEG-1 video, Stereo audio
Average bandwidth	189129 Bytes/sec
Length	70 minutes
Throughput std. dev.	56044.1
Disk Model	Seagate Cheetah X15 (Model ST336752LC)
Mixed-load factor α	1.0 (retrieval only experiments) 0.0 (recording only experiments) 0.4094 (retrieval and recording mixed experiments)
Relationship factor β (between R_{Dr} and R_{Dw})	0.6934
Mean inter-arrival time λ of streaming request	5 seconds
Server observation window T_{svr}	1 second
Disk block size B_{disk}	1.0 MB
Number of disks (ξ)	1, 2, 4, 8, 16, . . . , 1024

Table 3: Parameters used in the experiments and analysis.

a $p_{req} = 1\%$ threshold for missed deadlines and the resulting maximum number of streams supported with different configurations is summarized in Table 4. Note that in all the experiments the disk read and write cache are turned on and hence their effects are included in the results.

4.1.2 Retrieval Only Experiments

To enable only retrieval streams in the system we set $n_{ws} = 0$ and the mixed-load factor $\alpha = 1$. Fig. 9(a) shows the measurement and theoretical results for the DVD movie "Twister." The y-axis shows the probability for missed deadlines of all block requests. When the number of streams $n \leq 55$, then the probability is very small ($< 1\%$). Above this threshold, the probability increases sharply and reaches 1 for $n = 62$. The analytical results based on our 3RV model follow the measurements very closely, except that the 1% transition is one stream higher at 55. The miss probability of the 1RV model is also shown and its transition point is 39. Consequently, not only does our 3RV model result in a 38% improvement over the simpler model (for $p_{req} = 1\%$), but it also tracks the physical disk performance much more accurately.

The results for the VCD movie "Charlie's Angels" are shown in Fig. 9(c). Because of the lower bandwidth requirement of this video, a much higher number of streams (150 resp. 200) can be supported. The improvement of 3RV over 1RV is similar to the "Twister" case and we have omitted the graphs for "Saving Private Ryan" because the results were comparable.

4.1.3 Recording Only Experiments

Next we performed recording only experiments with $n_{rs} = 0$ and the mixed-load factor $\alpha = 0$. Note that there is no comparison with any other technique, because to the best of our knowledge no prior work exists that investigated these issues.

Figs. 9(b) and (d) show the miss probabilities for our recording experiments. Analogous to the retrieval case, the 3RV curve very closely matches the measured values. Since the disk write bandwidth is significantly lower than the read band width (see Fig. 1(b)), the transition point for, say "Twister," is $n = 40$ instead of $n = 55$ in the stream retrieval experiment.

Parameters		Analysis	Measurements	
Mixed-load Factor α	Movie Name	$n_{max_{3rv}}$[a] 3RV Model	$n_{max_{true}}$[b] Measurements	Error[c]
1.0 (Retrieving Only)	"Twister"	54	55	1.82%
	"Saving Private Ryan"	49	51	3.92%
	"Charlie's Angels"	202	225	10.22%
0.0 (Recording Only)	"Twister"	40	40	0%
	"Saving Private Ryan"	36	38	5.26%
	"Charlie's Angels"	150	171	12.28%
0.4094 (Mix of Retrieving & Recording)	"Twister"	46	46	0%
	"Saving Private Ryan"	42	44	4.55%
	"Charlie's Angels"	172	194	11.34%
	"Charlie's Angels" and "Saving Private Ryan"	66	70	5.71%

[a] $n_{max_{3rv}}$ is the maximum number of supportable streams computed by the 3RV Model.
[b] $n_{max_{true}}$ is the maximum number of supportable streams obtained via measurements.
[c] The error is computed as $\frac{n_{max_{true}} - n_{max_{3rv}}}{n_{max_{true}}} \times 100\%$.

Table 4: Experimental results for different mixed-load factors α and different movie types, assuming $p_{req} = 1\%$.

4.1.4 Mix of Retrieval and Recording

A realistic workload for a large scale streaming media system is a mix of retrieval and recording load. As an example, we chose an equal number of retrieval and recording streams for our experiments (any other combination is also possible), i.e., $n_{rs} = n_{ws}$ and thus the mixed-load factor $\alpha = 0.4094 = \frac{n_{rs}}{n_{rs} + \frac{n_{ws}}{\beta}}$.

Fig. 11(a) shows an example graph for the movie "Twister" (the other media types produced analogous results). As expected, the 1% transition point at 46 streams lies between the pure retrieval (54) and recording (40) values. (See also Table 4 for summary information.)

As a final verification of the 3RV model, we performed an additional mixed workload experiment. However, we not only mixed retrieval and recording streams ($\alpha = 0.4094$), but also two different media types (the DVD movie "Saving Private Ryan" and the VCD movie "Charlie's Angels"). Fig. 11(b) shows the experimental results and once again, the miss probability computed by 3RV model closely matches the measured results.

4.2 Missed Deadline Probability Analysis for Multi-Disk System

We evaluated the TRAC algorithm through numerical analysis for a multi-disk storage systems, with the number of disks ξ ranging from 2 to 1024. We set α equal to 1.0 and chose the movie "Twister" as the candidate media type. The maximum number of supported streams for both the TRAC and 1RV-AC algorithms were computed for a user acceptable missed deadline probability p_{req} of 0.01. Table 5 summarizes the results. As expected, the benefits of the TRAC algorithm increase linearly with the number of disks. We have not included measurement results from disk arrays for the following reason. With only a few disks, the number of streams admitted is expected to scale linearly. However, with current generation, high-performance disks other parameters start to influence the results of disk arrays very quickly. For example, the Cheetah X15 disk used in our experiments has a read transfer rate in excess of 55 MB/s. Hence, the device interconnect starts to become a bottleneck (e.g, SCSI buses top out at 320 MB/s, but more importantly a regular PCI bus can only sustain 133 MB/s). Consequently, to obtain accurate results one needs to either test the disk array using machines that have faster, but less common I/O interfaces (e.g., PCI-X or PCI-Express [16]), or incorporate a complete system model into the admission control procedure. We are working on the latter approach as part of our future research.

Parameters	The number of supportable streams		
number of disks	TRAC algorithm	1RV-AC algorithm	Improvement
1	54	39	38.46%
2	110	81	35.8%
4	222	165	34.55%
8	445	324	37.35%
16	893	660	35.3%
32	1789	1337	33.81%
64	3582	2694	32.96%
128	7167	5460	31.26%
256	14334	10940	31.02%
512	28674	21905	30.9%
1024	57367	43858	30.8%

Table 5: The total number of supportable streams with a miss probability threshold of $p_{req} = 0.01$ for both the TRAC and the 1RV-AC algorithms (Cheetah X15 disk).

4.3 Computational Complexity Analysis

4.3.1 Theoretical Analysis

The admission control algorithm must be executed for each new stream arrival. Therefore, for practical purposes its computational complexity should be low. The key component that dominates the complexity is the calculation of p_{iodisk}, because it involves the multi-dimensional integration of a continuous function. Our goal is to find the minimum computational complexity of an approximation attaining a given level of error ϵ, denoted as ϵ-*approximation*. We assume that the approximation of the evaluation is based on an average case, and hence the expected error is at most ϵ and the computational complexity is the minimum expected cost. It has been shown that quasi-Monte Carlo algorithms are optimal for such averages [6]. Let $comp^{avg-det-unit}(\epsilon)$ denote the minimum cost to compute the integration with the domain of a d-dimensional unit cube, then $comp^{avg-det-unit}(\epsilon) = \Theta(\epsilon^{-1}[log\epsilon^{-1}]^{\frac{d-1}{2}})$ [6]. Consequently, let $comp^{avg-det}(\epsilon)$ denote the minimum cost to compute the integration with domain $\Re$ as defined in Eq. 11 (and illustrated in Fig. 5). Let D, T and R denote the range of the random variables $\sum_{i=1}^{n} D(i)$, $\overline{t_{seek}}$ and $\overline{R_{Dr}}$ respectively. Then $comp^{avg-det}(\epsilon)$ can be derived as follows:

$$comp^{avg-det}(\epsilon) = \begin{cases} \Theta(DTR\epsilon^{-1}[log\epsilon^{-1}]^{\frac{3}{2}}) & \text{3RV model} \\ \Theta(D\epsilon^{-1}) & \text{1RV model} \end{cases} \quad (20)$$

Eq. 20 shows that the computational complexity depends on the integration region and the level of error ϵ. Consequently, the TRAC algorithm has a higher computational complexity than the conventional 1RV-AC algorithm, based on a single random variable.

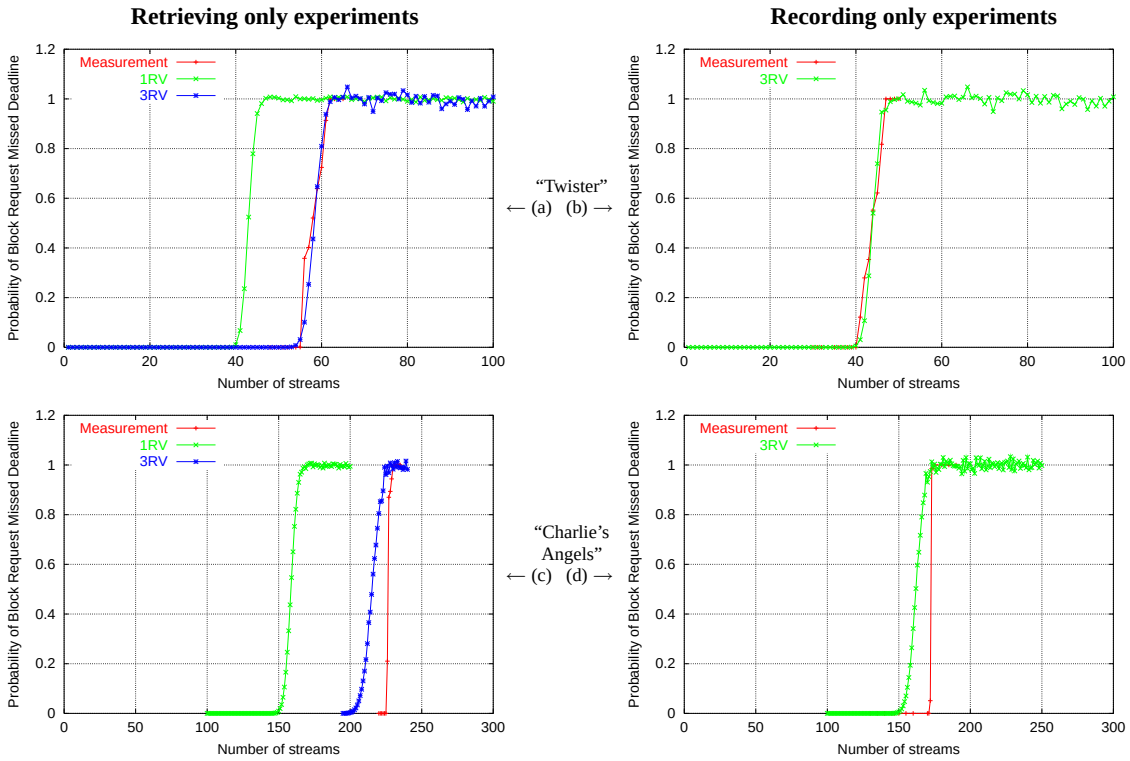

Figure 9: Retrieval or recording only experimental results.

4.3.2 Empirical Measurement

To verify the feasibility of integrating either the TRAC and 1RV-AC into a real system where the admission control procedure must be executed in real time, we implemented both models using the Plain Monte-Carlo integration method provided by GNU Scientific Library (GSL 1.3). Experiments were performed on a machine with an Intel P4 Xeon 2.0 GHz processor and 512 MB of memory.

The plain Monte-Carlo method samples points randomly from the integration region to estimate the integral and its error[3]. The accuracy of Monte-Carlo method depends upon the number of samples that are taken (i.e., number of library calls made). A higher number of calls results in more accuracy, but more time is spent for processing.

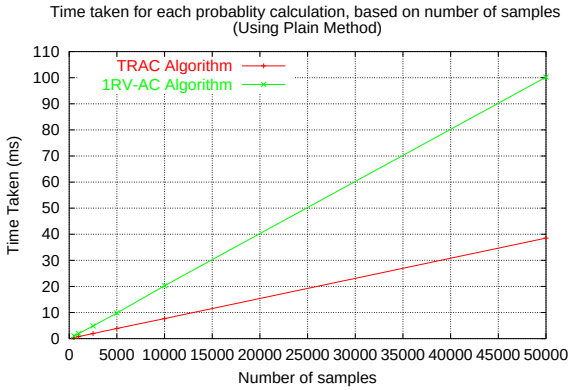

Figure 10: Comparison of the execution times for the TRAC and 1RV-AC algorithms.

[3]The error estimate should be taken as a guide rather than a strict error bound as it might be underestimated.

Fig. 10 compares the execution times of the TRAC and 1RV-AC algorithms. In both cases, the execution time increases linearly as a function of the number of samples. For the 1RV-AC algorithm, the execution time varies from less than 1 ms with 500 samples to 40 ms with 50,000 samples. For TRAC, the execution time is approximately 2.5 times higher than for 1RV-AC with the same number of samples. Fig. 12 shows the estimated integration error for the 3RV and 1RV model, respectively. For the same number of samples, the 1RV model usually generates a more accurate results than the 3RV model, which is intuitively clear, because more samples are needed to generate the same level of accuracy for higher dimensional integration. Fortunately, with 10,000 samples, we can obtain the results in 20 ms with approximately 5% error for the 3RV model. Hence, the TRAC algorithm is well suited for execution in a real time streaming media server.

5. CONCLUSIONS

We have presented a novel admission control algorithm called TRAC that considers a more realistic disk model and a dynamic disk bandwidth sharing scheme for both the retrieval and recording of streams. Our extensive measurement and analysis shows that the proposed algorithm can greatly increase the number of supportable streams at the minor expense of higher computational complexity. Hence, the disk bandwidth resources are used much more efficiently. We plan to implement the TRAC algorithm in our current project that aims to build a prototype Gigabit stream recorder [21]. We will evaluate its performance in conjunction with other components of the system, such as the buffer management, in a multi-node multi-disk environment.

6. REFERENCES

[1] Y. Bao and S. A.S. Performance-driven adaptive admission control for multimedia applications. In *IEEE International Conference on Communications, 1999 (ICC '99)*, pages 199–203 vol.1, 1999.

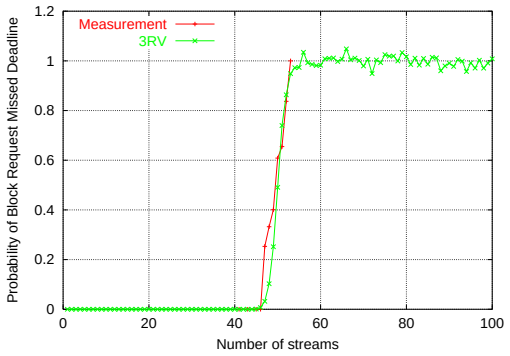

Fig. 11(a): "Twister" (retrieval and recording).

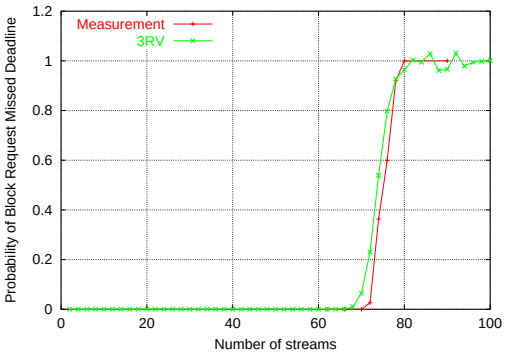

Fig. 11(b): A mix of "Saving Private Ryan" and "Charlie's Angels" (retrieval and recording).

Figure 11: Mixed workload recording and retrieval experiments.

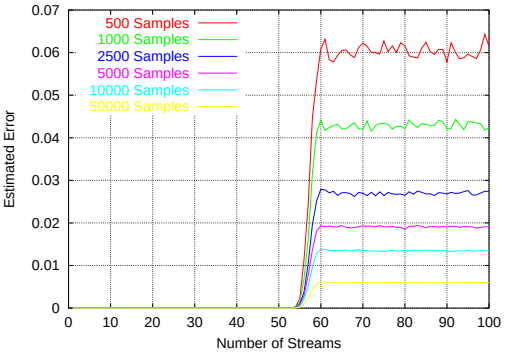

Fig. 12a: 1RV Using Plain Method

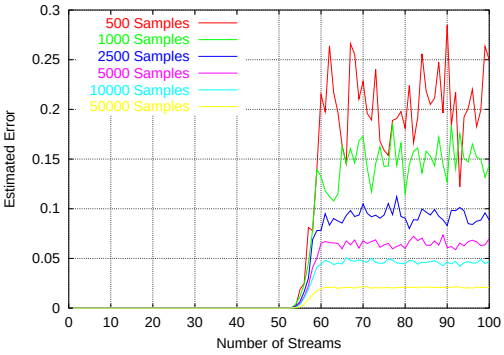

Fig. 12b: 3RV Using Plain Method

Figure 12: Estimated integration error for multidimensional Monte Carlo integration.

[2] S. Berson, S. Ghandeharizadeh, R. Muntz, and X. Ju. Staggered Striping in Multimedia Information Systems. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 1994.

[3] E. Chang and A. Zakhor. Variable bit rate mpeg video storage on parallel disk arrays. In *First International Workshop on Community Networking, San Francisco, CA*, pages 127–137, July 1994.

[4] S.-T. Cheng, C.-M. Chen, and I.-R. Chen. Dynamic quota-based admission control with sub-rating in multimedia servers. *Multimedia Systems*, 8(2):83–91, 2000.

[5] M. Friedrich, S. Hollfelder, and K. Aberer. Stochastic resource prediction and admission for interactive sessions on multimedia servers. In *ACM Multimedia*, pages 117–126, 2000.

[6] H.Wozniakowski. Average case complexity of multivariate integration. *Bull. Amer. Math. Soc. (New Ser.), 24(1)*, pages 185–194, 1991.

[7] S. Kang and H. Y. Yeom. Statistical admission control for soft real-time vod servers. In *ACM Symposium on Applied Computing (SAC 2000)*, March, 2000.

[8] I.-H. Kim, J.-W. Kim, S.-W. Lee, and K.-D. Chung. Measurement-based adaptive statistical admission control scheme for video-on-demand servers. In *The 15th International Conference on Information Networking (ICOIN'01)*, pages 472–478, 31 January - 2 February 2001, Beppu City, Oita, Japan.

[9] S.-E. Kim and C. Das. A reliable statistical admission control strategy for interactive video-on-demand servers with interval caching. In *Proceedings of the 2000 International Conference on Parallel Processing, Toronto, Canada*, August 21-24, 2000.

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

[11] D. J. Makaroff, G. W. Neufeld, and N. C. Hutchinson. An evaluation of VBR disk admission algorithms for continuous media file servers. In *ACM Multimedia*, pages 143–154, 1997.

[12] R. Muntz, J. R. Santos, and S. Berson. Rio: a real-time multimedia object server. In *ACM Sigmetrics Performance Evaluation Review, Volume 25, Issue 2*, pages 29–35, September 1997.

[13] A. R. Narasimha and J. C. Wyllie. I/O issues in a multimedia system. *IEEE Computer*, 27(3):69–74, 1994.

[14] G. Nerjes, P. Muth, and G. Weikum. Stochastic service guarantees for continuous data on multi-zone disks. In *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 1997)*, pages 154–160, May 12-14, 1997, Tucson, Arizona.

[15] A. Papoulis and S. U. Pillai. *Probability, Random Variables and Stochastic Processes*. McGraw-Hill Companies, ISBN: 0-07366-0116, October 2001, 864 pages, 4th Edition, 2002.

[16] L. D. Paulson. The Ins and Outs of New Local I/O Trends. *IEEE Computer*, 36(7):13–16, July 2003.

[17] C. Ruemmler and J. Wilkes. An introduction to disk drive modeling. *IEEE Computer*, 27(3):17–28, 1994.

[18] H. Stark and J. W. Woods. *Probability and Random Processes with Applications to Signal Processing*. Prentice-Hall, Inc. Upper Saddle River, New Jersey 07458, 2002.

[19] H. M. Vin, P. Goyal, and A. Goyal. A statistical admission control algorithm for multimedia servers. In *ACM Multimedia*, pages 33–40, 1994.

[20] R. Zimmermann and K. Fu. Pushing the limit of resource utilization: Statistical admission control for streaming media servers, USC technical report, university of southern california, 2003.

[21] R. Zimmermann, K. Fu, and W.-S. Ku. Design of a Large Scale Data Stream Recorder. In *Proceedings of the 5th International Conference on Enterprise Information Systems (ICEIS 2003), Angers - France*, April 23-26 2003.