BOSTON UNIVERSITY
COLLEGE OF ENGINEERING

Dissertation

MODELING AND DESIGN OF CONTINUOUS MEDIA STORAGE ARCHITECTURES

by
DINESH VENKATESH

B.E., Bangalore University, 1990
M.S., Boston University, 1992

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
1998

Approved by

First Reader: _____

Thomas D.C. Little, Ph.D.
Associate Professor of Electrical and Computer Engineering
Boston University

Second Reader: _____

David A. Perreault, Ph.D.
Professor of Electrical and Computer Engineering
Boston University

Third Reader: _____

Percy Tzelnic, Ph.D.
Director, Network Storage Group
EMC Corporation

Fourth Reader: _____

Roscoe Giles, Ph.D.
Associate Professor of Electrical and Computer Engineering
Boston University

Dedication

To My Parents

For their love, sacrifice, encouragement, and patience

and

Ajji

Whose memories shall always be with me

Acknowledgments

My advisor, Prof. Tom Little, has constantly inspired me to set high goals for myself and helped me attain them. Ever since I became a member of the Multimedia Communications Laboratory, Tom has enthusiastically guided my research and was patient as I meandered through many topics and ideas. I thank Tom for his guidance, support, encouragement, and friendship.

Wayne Duso, my manager at EMC constantly encouraged me in my endeavor and provided an excellent atmosphere at work that let me balance my educational and professional commitments. I thank Wayne for his support and for helping me maintain my focus as I tried to complete this dissertation.

Dr. Percy Tzelnic of EMC served as a reader on my committee and was instrumental in initiating the collaboration between EMC and the MCL that supported much of this work. I would like to thank Percy for his wisdom and valuable guidance.

I would like to thank Profs. David Perreault and Roscoe Giles for their encouragement and guidance. Prof. David Castañon agreed to chair my dissertation committee at short notice and provided valuable insight for which I am indebted to him. Prof. Azer Bestavros provided valuable input on several research problems for which I would like to thank him.

My colleagues at the MCL have constantly encouraged me and my interactions with them have enabled me to broaden my knowledge. I would like to thank John Gibbon, Huang-Jen Chen, Anand Krishnamurthy, Gulrukh Ahanger and Rajesh Krishnan for the many stimulating discussions and collaborations that made my stay at the MCL very enjoyable.

My colleagues at EMC have constantly encouraged me. John Forecast and Lev Vaitzblit were always willing to discuss new ideas and issues. Bill Conway wrote the "voltest" utility which enabled me to measure disk performance.

Abhi, Anand, Ashvin, Madhuri, Reshma, and Ximena have been my family away from home and constantly encouraged me for which I am indebted to them.

Last but not the least, I thank my parents for their love and encouragement, Peli for letting me be a big brother, and Sowmya for inspiring me to finish this dissertation.

MODELING AND DESIGN OF CONTINUOUS MEDIA STORAGE ARCHITECTURES

(Order No.)

DINESH VENKATESH

Boston University, College of Engineering, 1998

Major Professor: Thomas D.C. Little,
Associate Professor of Electrical and Computer Engineering

Abstract

The design of large-scale continuous media (CM) storage servers must balance the conflicting goals of achieving rapid response times and high server utilization. This dissertation addresses these issues in the context of disk-based server architectures and proposes a server design approach that scales across a spectrum of application bandwidth requirements.

A data streaming model is considered to evaluate latency/bandwidth tradeoffs in disk systems. The model is subsequently extended to consider disks with Zone Bit Recording (ZBR) to exploit disk geometry to trade capacity for bandwidth by an optimal grouping of disk zones. Application of the design technique is shown to reduce disk counts compared with traditional data placement for several capacity/bandwidth requirements.

A server cost-performance model is developed to determine disk parameters that yield the minimum cost per unit bandwidth for a given stream bandwidth, device capacity, and memory cost. This work demonstrates the inefficiencies of building server architectures based on streaming data for low bandwidth applications. The cost model also illustrates the need to constrain the number of drives in a disk array to prevent inefficient server operation.

The server cost and disk streaming models form the basis of a design procedure that can be used to create large-scale CM servers given data describing user behavior, system component costs, media types and the desired levels of interaction. These results are applicable to the design of large-scale CM servers are are supported by extensive analysis, simulation, and measurement. A simulation study based on a workload derived from observations of user behavior in movie theaters and object size distributions based on data from the Internet Movie Database is used to demonstrate the design procedure.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The past decade has seen much interest in the design and implementation of multimedia communication systems. Multimedia on the desktop has become a reality, due to powerful and inexpensive computer systems. Multimedia systems have also benefited from the rapid growth in the communication infrastructure that has enabled a wide range of applications to take advantage of the technology as is evident from the rapid proliferation of the World Wide Web (WWW). Digital systems are the preferred means of information interchange in the near future as digital data are easily stored, processed, and transmitted with little human intervention. Furthermore, digitization of data offers the added benefits of scalability, reliability, and consistent quality.

Despite these significant improvements, most widely deployed applications are optimized for low-bandwidth communication and utilize only text and graphic media types. Applications that deliver continuous media data types such as audio and video are typically of low-quality (e.g., monaural audio and video with small aspect ratios and low frame rates). While useful in low-end applications (e.g., corporate education and distance learning) such systems are unsuitable for deployment in a large number

of commercial scenarios, especially those requiring high quality CM data delivery.

This observation is interesting when one considers that much of the initial hype surrounding continuous media (CM) applications was created by the prospect of delivering entertainment services to consumer homes via video-on-demand (VOD) [29, 44]. However, VOD as an application is yet to mature and is proving significantly difficult to implement on a large scale despite the many user trials that have been conducted to date [9, 41, 50, 64].

This delay can be attributed primarily to the lack of an existing infrastructure that can handle the bandwidth requirements of high quality CM applications. Despite the gains in video compression technology, high quality video requires a significant amount of storage and communication bandwidth. The high cost of building this infrastructure has been a deterrent to these applications. In contrast, the World Wide Web is built over existing networking and computer infrastructures.

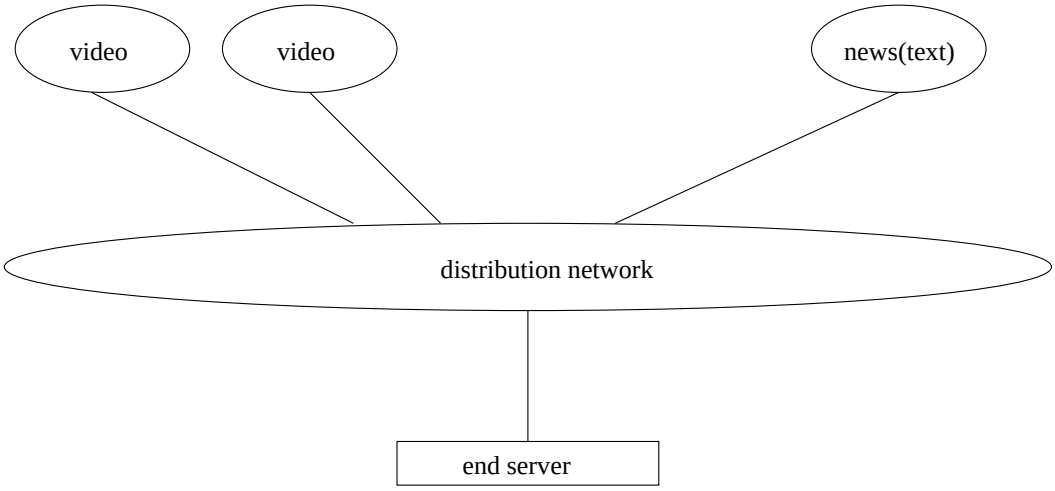


Figure 1.1: A Server Hierarchy

It is clear from this discussion that a widespread deployment of high quality CM applications will require a new information delivery infrastructure. Fig. 1.1

presents a simplified view of such an architecture [29]. Several application-specific servers are interconnected via a delivery network to many user-end servers. The end-user servers act as user access points and may cache information locally for improving access speeds [44].

Such an information delivery infrastructure will require a variety of components including storage servers, communication networks, end-user systems and the protocols and software to control their operation. In this dissertation, our focus is on the storage-server component of such CM systems.

1.1 CM Storage Servers

The server component in an information system is required to perform several activities. Fig. 1.2 illustrates a functional view of a storage server. We summarize the functionalities provided by each component below:

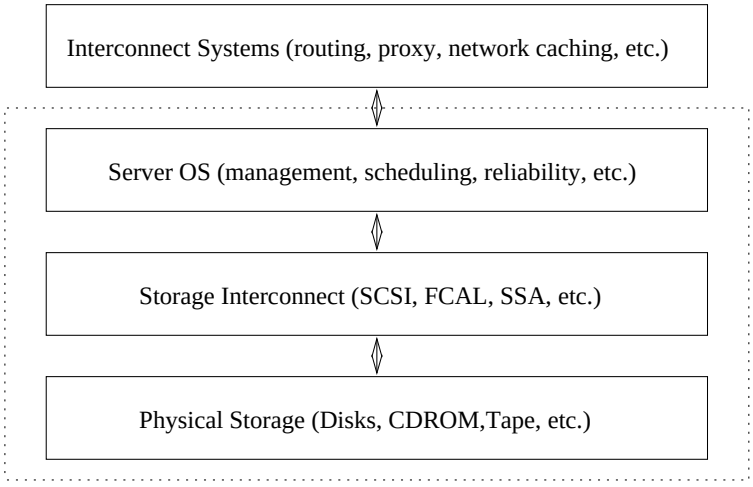


Figure 1.2: A Storage Server System Hierarchy

- **The network interface** provides the functions necessary to transfer data from/to the server. The performance of this component is affected primarily by the choice of the network protocol used to interface the server to the external world.

- **The server operating system** provides services to translate data between network and storage formats and mechanisms to manage server resources. In some cases, it may also provide a browsing mechanism to access the stored data.

- **The storage interconnect** is a second-level network that connects multiple storage devices to the server. The choice of the interconnect affects the speed at which data can be retrieved from the physical storage into the server operating system.

- **The physical storage** consists of the devices (tape, disk, memory, etc.) used to store data. The physical storage represents the data-store in the server from which data are delivered to the client.

The bulk of previous studies in CM server design focus primarily on one aspect: maximizing the number of concurrent streams of a given bandwidth that a server can support.[1] Furthermore, most studies are limited to a single bandwidth or specific encoding schemes such as JPEG (Joint Pictures Expert Group) or MPEG (Moving Pictures Expert Group). These studies aim to establish policies within the server or disk operating systems that efficiently utilize limited server resources to satisfy the delivery needs of a large client population. They attempt to overcome the limitations of conventional storage technologies (disks and tapes) that cater primarily to conventional data types that do not have the notion of timeliness associated with them.

[1] Chapter 2 provides an extensively survey of related work.

Optimizing server streaming, while important, addresses only one of the many issues that must be considered when designing CM server systems. We list some others below:

- Support for diverse media types.
- Support for data deletion, addition, and reorganization.
- Cost of streaming data to the end-user.
- Server reliability and failure recovery.
- Communication (bandwidth) and storage (capacity) scalability.

The need for supporting diverse media types is evident from the proliferation of the WWW. Future information delivery infrastructures will be required to support a multitude of applications and a variety of media types. This is clearly to the users' benefit as many applications are supported via the same infrastructure. However, this requirement poses a challenge to the server designer who must meet the delivery requirements of a variety of media types.

Efficient techniques for content replacement and reorganization are critical in application domains that experience frequent updates (e.g., news delivery). The server must not only support the delivery requirements of a large user population, but must also enable new content to be stored on the server. Content creation and deletion are necessary actions of normal server operation. However, there is an excessive overhead in management complexity when traditional file management and retrieval techniques are applied to continuous media systems that results in poor system performance.

Similarly, cost is the main factor in determining the success of a multimedia server. Since cost plays a major role in user acceptance, minimizing the cost of building a server is critical. However, a designer must be aware of the tradeoffs that result when a system is used in different ways (e.g., different media types). Reliability and scalability are more important to the service providers who implement the servers as it allows them to expand and provide better service to their customers.

While several studies have considered storage server design for CM systems from the perspective of a system that supports few tens of users, the solutions do not scale well to more complex scenarios. Few studies have addressed the design of a CM server supporting hundreds of users and have also considered the issues of scalability. Those studies that address the issues of scale often do so from the perspective of a single media type. It is not apparent whether similar techniques work efficiently with different media characteristics.

In this dissertation, we focus on the design tradeoffs that a server designer must consider when building a storage server. Our study is motivated in part by the desire to provide solutions for building inexpensive servers that can reduce the cost of an information infrastructure. Consequently, the work in this dissertation can be classified into three broad categories:

- System models and performance analysis
- Models of user accesses and media characteristics
- Algorithms and techniques for efficient server operation

Modeling the system components allows us to understand the hardware limitations in supporting a desired load. CM servers are usually built using disk drives

that have non-linear response times. Understanding these limitations allows us to determine the feasibility of supporting a given load from a given system as determined by current component costs (e.g., disks, RAM).

Modeling user access behavior and the characteristics of the data being accessed allows us to define the system requirements. Given a user population and the characteristics of the media types, we can predict the storage and bandwidth requirements of a system along with any related response time requirements.

The server operation algorithms determine how best to partition server resources when the requirements of multiple clients must be met simultaneously. In reality each of these issues cannot be dealt in isolation. However, separating them allows us to iteratively address each issue and design practical solutions.

In this work, we focus on data with periodic delivery requirements. This notion of data delivery when data must be delivered periodically or *streamed* over a reasonably long interval (called a *session*) is a central theme in our work. To this end, we conduct an in-depth investigation into the requirements for resource allocation and management in systems where data must be streamed from the server to the end-user.

1.2 Contributions

The results and accomplishments from this work are summarized below:

In the first part of our work, we analyze disk system performance and the resulting performance tradeoffs for supporting CM data. Specifically, we consider the application of a given disk scheduling algorithm to derive an estimate for the number of concurrent streams that a disk can support [67]. We use this estimate to study

disk utilization and latency constraints for different bandwidths. On-disk buffering and the resulting constraints on maximum disk utilization are used to derive a bound on the maximum disk transfer size. This limit on disk utilization is used to constrain the original disk model and to bound the concurrent number of sessions a disk can support. The disk model is then expanded to consider Zone Bit Recording (ZBR). Here, we propose a new technique for selectively grouping zones to trade capacity for bandwidth in a CM server. These studies are validated against measurements from real disk systems. Finally, we describe a system cost model and discuss techniques for its application in server design.

In the next phase of the work, we consider the effects of user access behavior on server performance. We describe user access behavior in a WWW server and in movie theaters [66]. This work allows us to build models that describe changing user preferences and describe the fluctuations in load a server can experience. Next, we evaluate techniques for load balancing by minimizing session blocking probability in replicated servers [45] and derive bandwidth allocation criteria for estimating content reorganization times in multi-disk storage servers.

Finally, we consider a model to describe a video object size distribution that is based on movie running times as described in the Internet Movie Database (IMDB) [68]. This data is used in conjunction with the access models to build a simulation model to demonstrate the process of CM server design.

Collectively this work comprises a set of guidelines that can be used to configure large scale CM servers given data describing user behavior, system component costs, media types and the desired levels of interaction. This work also demonstrates the inefficiencies of streaming data from disks for low-bandwidth applications as well as the need to constrain the number of drives in a disk-array for efficient server oper-

ation.

The solutions proposed in this dissertation are extensively evaluated using simulations, analysis and measurements on real systems. This demonstrates the applicability of the proposed mechanisms in real systems and illustrates their utility. Finally, we applied the aforementioned techniques and developed a storage server model. The model can be applied to analyze and evaluate the effects of data partitioning approaches and memory hierarchy design on server performance. Additionally, the model can be used to evaluate data allocation policies to balance I/O and storage bandwidths in a manner that maximizes the availability of information to the user. This tool is a valuable aid to system designers who can evaluate and enhance CM storage architectures.

1.3 Organization of the Dissertation

The remainder of this dissertation is organized as follows: In Chapter 2 we survey existing work for building CM storage servers and describe their advantages and shortcomings. In Chapter 3 we describe studies on disk systems and their performance. We also derive a cost model for disk based servers and evaluate the resulting tradeoffs from the different data layout policies. In Chapter 4 data layout and retrieval policies for multiuser server systems are described. We also describe studies of user access behavior and media characteristics. In Chapter 5 we derive a video-size distribution using data from the IMDB and present a system simulation model. Conclusions and directions for future work are presented in Chapter 6.

Chapter 2

Background and Related Work

Synopsis

In this chapter we discuss issues related to the design and implementation of CM servers and describe related work to date. We begin by considering traditional memory hierarchies and their influence on CM server architectures. Next, we illustrate the data-streaming paradigm and derive the basic criteria that must be addressed in server design. We subsequently describe server architectures based on rotating electro-magnetic disks in great detail. Relevant issues such as session scheduling, data-layout, user access behavior and storage management are described.

2.1 Storage Hierarchies and Server Design

Historically, storage technologies have been classified by their importance within a computer system. Consequently, storage is categorized as either primary, secondary, or tertiary based on the proximity (as measured in access latencies) to the processing unit [34, 35, 63]. Primary storage is usually built using solid-state components and can be accessed at very high speeds ($< 1\mu s$). Secondary storage typically consists of rotating magnetic disks with reasonable access latency ($< 100ms$) and tertiary storage typically consists of tape storage with relatively large access times ($> 1s$). This partitioning is well suited for computer systems where program access patterns demonstrate a strong locality of reference (both in space and time).

Much of the initial research in building memory hierarchies was aimed at reducing the execution time of computer programs. Computer programs involve operations on a data set and the execution time is dependent on data accessibility. As a result, memory management techniques attempt to minimize the time for data to manifest itself at the processing unit. Furthermore, computer programs exhibit spatial and temporal locality with regards to memory access which is exploited to design efficient memory hierarchies [34, 35]. Typically, the memory is organized into blocks (pages) and the design considerations include determining the optimum page size, cache replacement and update policies, and mechanisms for ensuring data consistency [61].

In contrast, I/O intensive applications (e.g., databases), are mainly concerned with increasing transaction throughput. Within this information-centric view, the main performance measure is the ability to quickly retrieve specified data. Since magnetic-disks are an established technology and provide the best price-performance ratio for I/O, they form the basis for most studies on optimizing storage performance. The main focus of these studies has been the derivation of allocation techniques that

place data such that access times are minimized. Some of these techniques include contiguous layout, linked lists, clustered systems, and indexing [26]. It has been demonstrated that contiguous allocations are optimal from a retrieval perspective and an indexed approach is preferable for a transaction-oriented system [26, 71].

Access patterns to the contents of a database also display large temporal and spatial localities that can be exploited by using a memory hierarchy. A memory hierarchy is a design that balances the costs of building a system with the desire to provide the fastest access to content. This is a fundamental tradeoff and in later sections we examine the design of such a memory hierarchy for the delivery of continuous media. Clearly, the characteristics of CM data significantly affect the choice of a memory architecture.

2.2 CM Server Storage Hierarchies

The requirements for CM storage servers are similar to that of database systems with the additional constraint that the data are streamed over reasonably long periods. Additionally, CM data types such as video and audio require significant I/O and storage resources unlike the small text or graphics objects that traditional databases usually manage. Technologies available for building CM servers include electro-mechanical disks, solid-state memory, tape, and optical storage.

Most existing CM server architectures use electro-mechanical rotating storage or magnetic-disks in their design. Disks are generally much cheaper than solid-state architectures such as flash memory and RAM, have reasonable access latencies, and provide the high throughput necessary to support the massive storage requirements of interactive CM systems. Solid-state storage devices have larger data transfer rates

and can support more concurrent sessions. Optical-disk technology is currently about ten times slower than disk storage and, while the cost of the media are low, the cost-performance ratio is much lower than that of drives when considered in conjunction with the cost of the access mechanism [5]. Alternate technologies such as holographic memories are promising but are not yet commercially viable. Thus, magnetic disks are currently the preferred media for building CM servers.

We now study the basic design criteria that a CM server designer must address that provides a basis for the work in this dissertation. We begin this discussion by understanding the characteristics of CM data.

2.3 Characteristics of Continuous Media

CM data types are characterized by the requirement for periodic display unlike static data types such as text and graphics. For example, NTSC video must be displayed at 30 f/s for the desired visual experience. Video data are the most bandwidth intensive amongst all CM data types. Displaying raw video data requires enormous system bandwidth (e.g., 270 Mb/s for HDTV quality video). Consequently, compression techniques can be applied to the CM data to decrease their bandwidth requirements. Two popular compression schemes are specified in the JPEG and MPEG recommendations of the International Standards Union (ISO) [49]. However, compression schemes that yield significant reductions can result in some degradation of visual quality.

Several compression schemes have been devised for supporting a gamut of video streaming applications. These include the high-quality/high-bandwidth (e.g., MPEG-II) and low-quality/low-bandwidth schemes (e.g., H-261). The high quality compression schemes are used in applications such as editing and broadcasting while

the low quality schemes are preferable for streaming video over the Internet or for real-time videoconferencing.

Compression can result in bandwidth requirements that vary over time and the data are characterized as having a VBR (variable bit rate) characteristic. For example, when video is compressed using a technique that takes advantage of both the spatial and temporal correlations in video data (e.g., MPEG), the resulting stream has a variable bit rate (VBR) characteristic. Fig. 2.1 illustrates the video frame sizes over a $5s$ interval for a hardware encoded MPEG-I video stream.

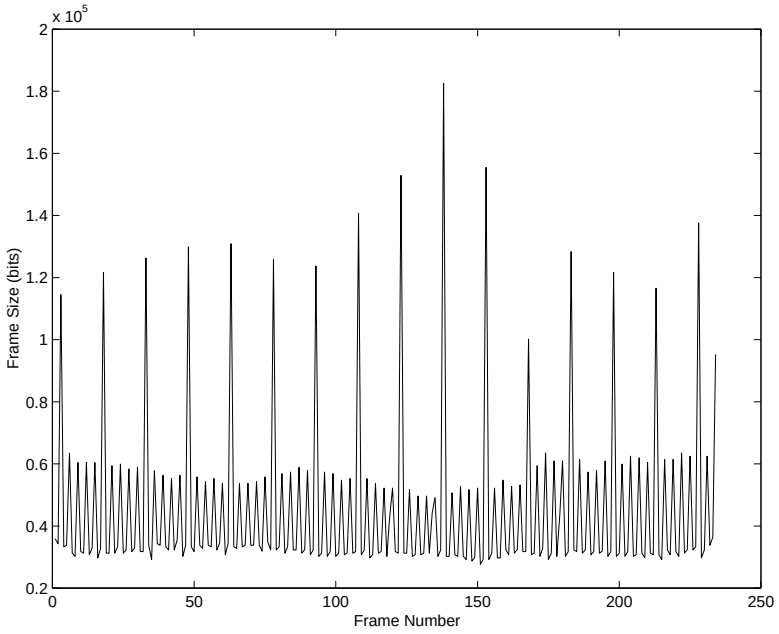


Figure 2.1: MPEG-I Video Frame Size Characteristics

In MPEG encoding, video frames are grouped into independent units called group-of-pictures (GOP) at encoding time. Each GOP represents a logical unit from the decoder's perspective. Fig. 2.2 illustrates the change in GOP sizes (for a GOP size of 15 frames) vs. time for an MPEG-I encoded video stream that is approximately 400 seconds long.

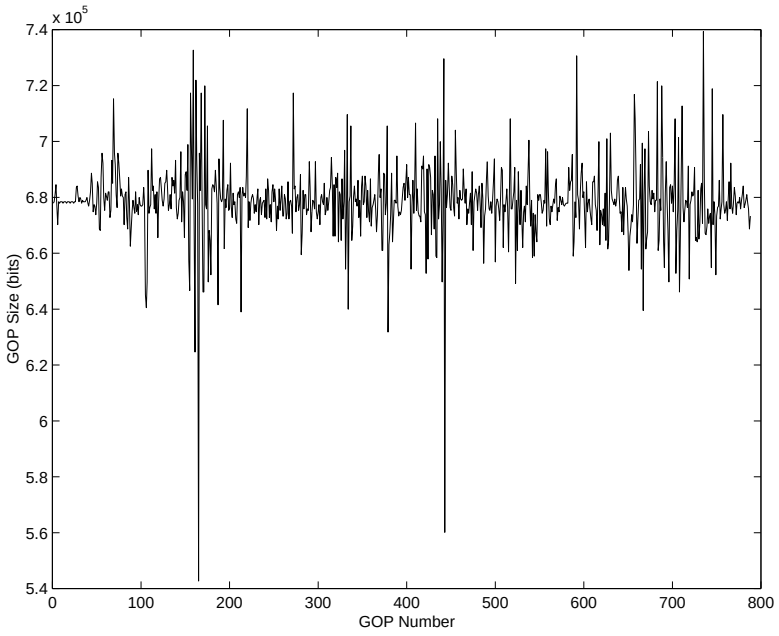


Figure 2.2: GOP Size Characteristics

Several approaches can be used to transmit VBR data from a server. One approach is to transmit the video stream to the user at a fixed data rate. Fixed rate transmission ultimately requires a smoothing buffer to absorb the variability in data rates. The minimal buffer size necessary to support a video stream is a function of the stream consumption rate R_c, the service rate, and the stream characteristics.

Let vector G represent the group sizes of the MPEG video stream. Consider a retrieval policy that retrieves data periodically from the disk and sends them to the user at a fixed rate. If k represents the number of groups retrieved during each service interval T_c, the amount of data that must be served during each service period i is given by

$$b(i) = \beta(i-1) + \sum_{j=i*(k-1)+1}^{i*k} g(j), \tag{2.1}$$

where the buffer occupancy at the end of each period $\beta(i)$ is given by:

$$\beta(i) = max[(b(i) - R_c T_c), 0], \qquad \beta(0) = 0. \tag{2.2}$$

The minimum buffer space necessary to support the playout of the stream at the client end is then given by $max(b(i))$. Fig. 2.3 illustrates $b(i)$ as a function of time for a cycle time $T_c = 1s$. The variation of $max(b(i))$ with respect to T_c is shown in Fig. 2.4 for the clip whose characteristics are given in Fig. 2.2. Using this analysis, we can compute the buffering requirements at the client in order to absorb the variation in data delivery rates depending on the characteristics of the bit stream.

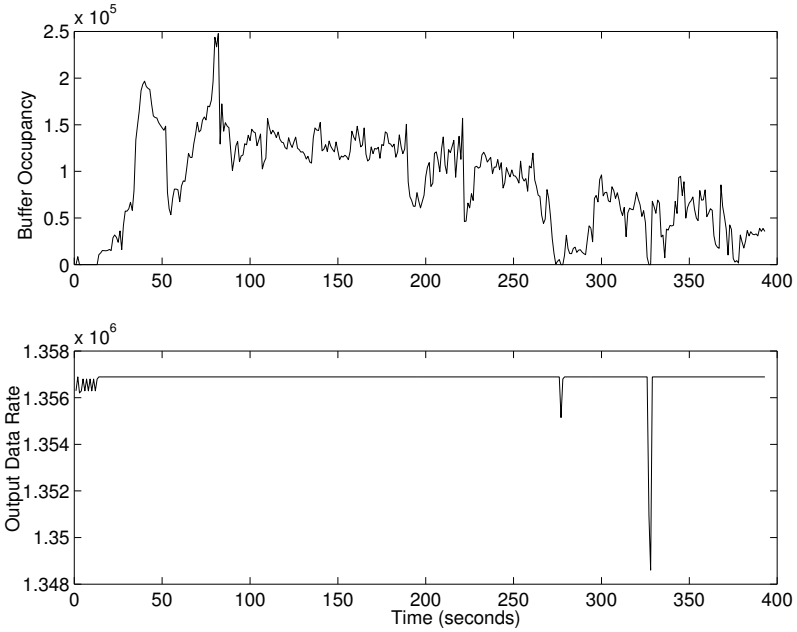


Figure 2.3: Buffer Dynamics for $T_c = 1s$

It is clear from the analysis that buffering the data to $max(b(i))$ will ensure jitter free playout of video at the client end. This also implies that the server can serve data in a CBR fashion, simplifying the task of scheduling the data retrieval

mechanism.

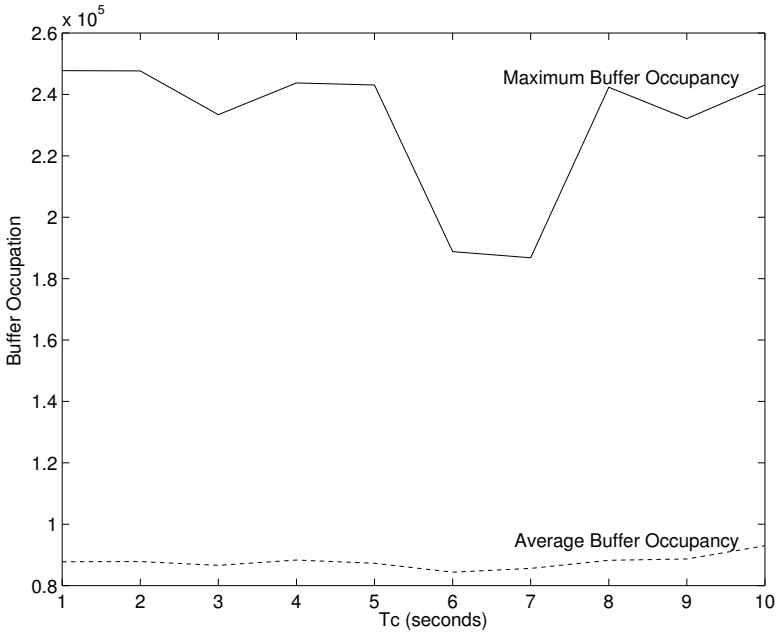


Figure 2.4: The Maximum Buffer Size vs. T_c for an MPEG Encoded Stream

A second approach to serving data from the server is to deliver data to the client at the rate of consumption. Such a policy requires the server to make guarantees that only the data necessary to satisfy a client's requirements are retrieved during each scheduling interval. This scenario requires a server admission control mechanism to prevent data overflow or underflow at the client.

However, relatively constant bit rate traffic can be generated with very little buffering overhead as illustrated in Fig. 2.3. Additionally, most practical encoders generate CBR streams because they are easy to manage, both for storage and transmission. Several studies address the requirements for multiplexing VBR streams and their delivery across the network based on stream characteristics at the granularity of a single video frame. In reality, as was demonstrated earlier, the MPEG stream from the hardware encoder is essentially CBR when considered at the GOP level. In

this dissertation, we focus on the performance of servers when serving CBR streams.

In the following sections, we consider the CM server architectures and define bounds for their operation.

2.4 A Basic Data Pipeline

All storage servers are required to deliver data within a reasonable delay-bound (latency) to the client requesting data. What make CM data different is the notion of "timeliness" imposed on the delivery of information. Not only must data be delivered accurately, but they must also be delivered within a certain time to ensure the continuous display of data at the client. To understand this simple concept, consider the data-pipeline illustrated in Fig. 2.5 which is a very general overview of the problem.

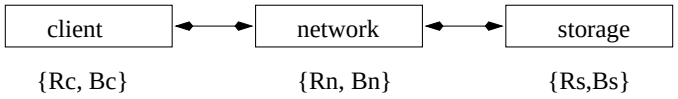


Figure 2.5: A Basic Pipeline for Data Delivery

The basic pipeline is a producer-consumer system that consists of a three node path that data traverse between the server (producer) and the client (consumer). This relationship can be inverted without any loss in generality. Each component in the pipeline may or may not have buffering. Much of the research in server design focuses on solving the constraints imposed by each component in the pipeline. Also, the notion of *rate* of consumption is fundamental to CM systems. While media types such as text and graphics do not have a notion of time, a rate can be associated to them by estimating the maximum delay before they must be displayed (display latency).

Let $\{R_c, B_c\}$ represent the rate of data consumption and buffering availability at the client and $\{R_n, B_n\}$ and $\{R_s, B_s\}$ represent the bandwidth and buffering available at the network and server respectively. Two basic delivery paradigms can be associated with this general architecture.

- **Store and Forward:** The store and forward paradigm is applicable when $R_c > min\{R_n, R_s\}$. Since data are continuously displayed at the client, they must be downloaded and stored at the client's buffer before being displayed. Clearly, there is a delay (startup latency $\mathcal{L}$) before data can be displayed at the client as sufficient data must be collected to ensure that the entire content can be displayed in a smooth manner. The startup latency $\mathcal{L}$ can be estimated as

$$\mathcal{L} = \frac{\mathcal{S}}{R_c - \min\{R_n, R_s\}} \tag{2.3}$$

  where $\mathcal{S}$ is the size of the media object being displayed. Thus, the latency in displaying an object is inversely proportional to the rate differential in the system.

  When $R_c \leq min\{R_n, R_s\}$, data can be displayed with a minimum latency at the client. Buffering at the client is necessary only when the server transmits data at a rate higher than R_c and is given by

$$B_s \geq \frac{\mathcal{S}}{\min\{R_n, R_s\}} - \frac{\mathcal{S}}{R_c}. \tag{2.4}$$

  Thus the two tradeoffs in server design are balancing $\mathcal{L}$ with B_s and is fundamental to the design of all CM systems.

- **Streaming:** The interesting and most studied case for the delivery of CM data is one in which all elements along the delivery pipe have a bandwidth

approximately equal to the client consumption rate (or data is *streamed* between the client and the server). In other words, data are streamed when $R_c = R_n = R_s$. Streaming requires guarantees of delivery made along each component in the data path and must ensure the delivery of data at the client at the exact times specified. Any variance in the delivery times leads to *jitter* and degradation of service quality [28].

Store-forward techniques can be implemented in conjunction with streaming, especially when buffering is available at the client. However such a policy is not desirable especially when the data must be displayed with a minimal latency or when the buffering at the client is limited.

2.5 CM Server Architectures and Server Scheduling

The term *server* is usually associated with a computer system configured to meet the computing requirements of many users or processes. The server system must provide a multiplexing (multitasking) mechanism that efficiently partitions available server resources among many users. This mechanism is usually called a *scheduler* in the context of a server operating system (OS) and the process by which resources are multiplexed is called *scheduling* [61]. Efficient scheduling is extremely important to server operation.

Most mechanisms for scheduling multiple requests in a CM server take advantage of the following assumption: *sessions are usually long-lived and data are streamed.* Consequently, the scheduler periodically retrieves session data from storage

at R_s, buffers them, and delivers data to the client at a rate R_c. Usually, $R_c << R_s$ and this rate differential allows for the scheduling of additional sessions within the period in which the data is consumed [4, 31, 54, 55]. If the transfer size is B, the interval before the next segment of data for a session must be retrieved from storage is given by $T_c = \frac{B}{R_c}$. Typically T_c is fixed, and for a constant R_c, one can easily determine the size of the buffer that must be retrieved to meet the client streaming requirements. Fig. 2.6 illustrates the scheduling process.

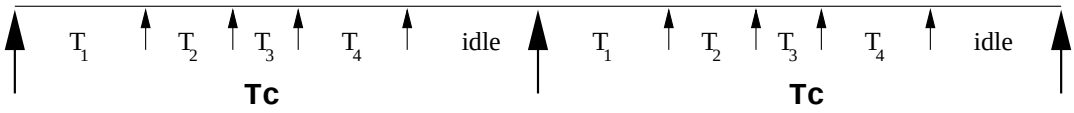


Figure 2.6: A Simple Scheduling Example

In the scheduling example of Fig. 2.6, T_c represents the *scheduling interval*. T_1, T_2, T_3, and T_4 represent the durations during which data are retrieved from storage for sessions 1, 2, 3, and 4 respectively. *idle* is the excess time when no CM data are retrieved for any session. Even though the spare time is denoted as an idle state, the server is active as data are being transmitted to the client. However, the storage system is idle during this interval, representing an under-utilization of storage resources.

The ordering of sessions within a scheduling interval can be structured to efficiently utilize server resources. Because data read in a scheduling interval are consumed (delivered) in a subsequent interval, the data must be buffered until they are delivered in their entirety. The amount of buffering that must be provided per stream is at least twice the amount of data consumed in any period [4, 31, 55]. In other words, the amount of buffering necessary for a stream is a function of its rate $\mathcal{R}$ and the scheduling interval T_c and is given by

$$\mathcal{B} = 2 * \mathcal{R} * T_c. \tag{2.5}$$

In practice, variance in the I/O subsystem can increase the buffering requirement [47]. Ensuring the continuity of display can require a new session request to be delayed by a finite duration before data are streamed. This delay, also known as a server-startup latency $\mathcal{L}_s$ is usually given by

$$\mathcal{L}_s = \frac{T_c}{2}. \tag{2.6}$$

$\mathcal{L}_s$ is in addition to the basic pipeline latency $\mathcal{L}$ and represents a initial latency at the server when the data are streamed.

Due to finite resources, the server cam support only a finite number of sessions. Consequently, the scheduler must enforce an *admission control* mechanism to ensure that the server does not accept a new request whose bandwidth requirements cannot be satisfied [54]. As a result, a user request for a new session can be denied (or delayed) by the operating system and the system is said to *block* when its request is denied.

Thus, any server design must:

- Maximize the number of sessions (i.e., minimize the probability of blocking) and
- Minimize $\mathcal{L}_s$ (i.e., provide fast response times).

Clearly, reducing $\mathcal{L}_s$ also reduces $\mathcal{B}$ which results in a net reduction in system cost and is desirable. Thus, any server design must balance the two important issues:

(i) The availability of sufficient resources to meet the requirements of multiple requests and (ii) the design of an efficient scheduler to maximize the utilization of existing resources.

We now enumerate the two important issues that must be considered when developing a CM storage server architecture:

- *Capacity and Bandwidth Estimation:* Due to the limited I/O bandwidth and storage capacity of a single storage device, supporting the access demand for a large number of users requires aggregating the bandwidths of multiple storage devices. The design of a CM server must consider the storage device types and organization in addition to other factors including expected system load, physical storage, media characteristics (such as data rates and timing requirements) and cost.

- *Resource Allocation and Replacement:* In addition to the hardware components, appropriate mechanisms for data management are necessary. Resources must be allocated to storage devices for maximizing the availability of information to the user. If an imperfect object-storage or user-storage mapping is used, the limitations in I/O bandwidth prevent perfect system utilization. As a result, it is important to develop mechanisms that can intelligently redistribute data so they are available to the users who need them the most. Additionally, the policies to replace obsolete objects when new ones manifest themselves must also be designed with care.

In other words, a design must address the design of an efficient storage architecture as well as its management given the system limitations. Often, these requirements are in conflict and complicate the design process.

We now survey research activity related to the design of CM servers. This survey provides a basis for the work described in the remainder of this dissertation.

2.6 Related Work

There has been significant interest in the design of CM servers in the past decade. Much of this work can be classified into the two broad categories of micro issues and macro issues. Micro issues in server design concern themselves with I/O and device level management for supporting CM applications. Macro issues address issues such as file management and are concerned with overall server operation and resource management.

The bulk of server designs for CM data are based on disk storage. A single disk system can support multiple user sessions when the disk I/O bandwidth exceeds the per session bandwidth. Multiple user sessions are supported on a single disk by multiplexing the disk bandwidth among the user sessions. Perfect disk bandwidth utilization is difficult due to the electro-mechanical nature of disk accesses that prevent instantaneous data accesses. This latency is primarily due to *switching, seek*, and *rotational* overheads a disk incurs when positioning the heads over the physical location of the desired data [57, 62]. Moreover, additional overheads are incurred in the disk-host interface that diminish disk performance [72].

The primary focus of studies on disk architectures for continuous media are disk scheduling and data placement with the primary objective of improving disk utilization. Several disk scheduling techniques have been proposed for minimizing disk switching overheads and efficiently utilize available disk bandwidth. These approaches are built upon traditional disk scheduling mechanisms such as First Come First Served

(FCFS), Shortest Seek Time First (SSTF), SCAN and C-SCAN [62]. Some of the new techniques that have been developed for retrieving continuous media from disks include Earliest Deadline First (EDF) and SCAN-EDF [56], Group Sweep Scheduling (GSS) [14], and preseeking [30]. These techniques exploit the observation that seek-times dominate disk overheads. They reorder disk requests so that the disk heads move a minimal distance within a scheduling interval [62].

Similarly, it is possible to place CM data such that the layout minimizes disk seek overheads [16, 48, 69]. This includes storing CM data contiguously to prevent multiple disk seeks within a single data transfer [16], computing the appropriate block size for storage necessary to minimize the seek times [48], or interleaving data from multiple streams [69].

Zone Bit Recording (ZBR) complicates the design of the storage subsystem. ZBR makes better use of available disk surface by using a constant recording density on all the disk tracks. This results in greater storage capacity but varying disk transfer rates for different cylinders of the disk. Some schemes take advantage of this variability to optimize data placement and increase the disk throughput [8, 21, 32]. In *track pairing*, tracks are paired to achieve transfer units with identical capacity and bandwidth properties. This allows the scheduling of user sessions at the average disk transfer rate and results in better disk utilization [8]. In the interleaved annular layout (IAL) scheme, segments are laid out such that distances among segments involved in a read cycle are short to achieve a higher throughput [21]. An alternative scheme is to distribute the data among the zones at a rate that is proportional to the zone transfer speed [32].

A second class of studies address the issues of providing statistical guarantees of disk performance for variable bit rate (VBR) encoded video [12, 17, 51, 70]. These

studies include techniques for selectively dropping JPEG encoded video in disk servers [60], data reorganization and layout techniques for MPEG video [17] and selective dropping of MPEG data [15].

Several researchers have also addressed the design of storage servers based on disk arrays for streaming CM data [6, 9, 10, 36, 32]. For disk arrays, the performance of the storage system is enhanced by aggregating the combined bandwidths of many devices. There have been several approaches that support the interactive playout of video data from disk based architectures. These approaches typically reorganize the video stream before storing it such that scheduling and bandwidth constraints are not violated during periods of interaction [10, 13, 15, 18, 39].

From a macro perspective, the main problems that must be addressed by a server architecture are related to data management. A server has finite resources as measured by its capacity and bandwidth and the replicated and striped disk arrays represent systems in which all data are stored and delivered from a single access point. An alternative approach is to satisfy the capacity and I/O bandwidth constraints by including a diverse set of storage devices and partitioning the storage into a hierarchy [25, 52, 53]. The partitioning of the storage components depends on the distribution of object popularities as well as the cost of the individual storage components.

Instead of providing a dedicated connection to each user, an alternative approach to building a CM server is reduce the number of interactive capabilities offered to the user. This can be achieved by not fully supporting interactive features or by scaling down sessions gracefully during periods of overload [17, 60]. Approaches for building scalable servers include batching techniques and service scaling during periods of overload:

- **Batching:** In batching, user sessions are delayed for a finite duration so that

requests for a given media object can be accumulated and then satisfied en masse [23]. The desired information is subsequently simulcast to all users reducing the I/O requirements. The batching delay depends on several factors including user behavior such as reneging and available system bandwidth. Interactivity can be provided via *contingency channels* that are reserved to handle interactivity. However, the availability of such a channel cannot be guaranteed to every user in the system [65].

- **Near Video-on-Demand:** In Near Video-on-Demand (N-VOD), time is slotted and users receive movies only at the beginning of a slot. Users are allowed to perform interactive VCR functions by switching to contingency channels. Upon the completion of an interaction, the system attempts to merge the user to a N-VOD channel that is the closest in time. Smooth merging of the interactive stream is achieved via buffering. Division of the time into slots allows the system to bound the merges, thereby simplifying its operation [1, 2]. However, just as in the delayed batching scheme, there are no guarantees that all users who desire an interactive channel will receive one.

- **Adaptive Piggybacking:** In adaptive piggybacking, the playout rates of the media streams are varied by speedup or slowdown so that they can be merged with other sessions that are close in time, thus conserving I/O bandwidth [33]. The rate of merging must be limited to within 5–10% of the original bandwidth for the effect on the user to be tolerable. This requirement places a lower bound on the time difference between two streams that can be merged. An alternate approach is look-ahead scheduling that provides users with pause and resume functionality [73]. This is achieved by reserving sufficient channels for each media stream and providing a buffer to absorb the delays introduced by interaction.

Most of the studies described here only consider MPEG-I data and constant object-sizes and user access patterns. In reality, servers will be required to deliver many data types with diverse bandwidth and capacity requirements. Under these circumstances, it becomes critical for a server designer to be aware of the performance tradeoffs that are feasible to design the correct architecture. We summarize these observations in the next section and outline the issues addressed in the remainder of this dissertation.

2.7 Summary

In this chapter, we have conducted an in-depth survey into the requirements for a CM storage server. It is clear that the measures of server performance are throughput (the number of concurrent streams supported), response time (the startup latency), and cost (buffering). Much of the related work described in this area addresses the design of a CM server from the perspective of static storage and data types (e.g., MPEG-I). In contrast, practical servers are expected to support multiple data types and are affected by additional system constraints.

For example, the performance of the disk subsystem is limited by available buffering on the disk surface. Due to the I/O rate mismatch between the disk transfer rates and the I/O bus, data must be buffered on the drive before they can be transferred in a burst to the host. This requirement limits the maximum throughput a drive can achieve which in turn limits its streaming capability. The I/O bus characteristics also limit the number of physical drives that can be interconnected.

Disk arrays perform in a superior manner for streams with high bandwidth requirements. This constraint follows from the observation that the data retrieved

for individual disks increases, offsetting the seek time overheads. When the data rate per stream is small, data retrieved per disk decreases, increasing the seek overhead. However, when the same technique is applied to a low bandwidth stream, it results in an excessive startup latency.

These tradeoffs must be quantified in the context of different data types and changing user requirements. Additional issues that must be addressed when building a CM server include:

- **Replication:** A storage device has a limited capacity for supporting concurrent CM streams. To support several users, a media object may have to be replicated across several devices.
- **Load Balancing:** It is impractical to expect all the resources in the CM server to be equally popular. Observation based schemes that predict usage patterns are necessary to balance I/O bandwidth against access demands.
- **Data Reorganization:** As media objects have a finite lifetime, techniques for replacing them when database updates occur are necessary. Content on a server is replaced over time and it may be necessary to replace content while streams are active.
- **Cost:** The critical factor that affects the usability of a CM server is the cost per unit bandwidth and must be minimized.

In this dissertation, we address each of the issues mentioned here and provide efficient solutions for them.

Chapter 3

Storage and I/O Constraints in Server Design

Synopsis

In this chapter, we describe the mechanics of streaming CM data from disk storage and evaluate the performance of several scheduling algorithms. A scheme is derived that maps disk scheduling constraints to evaluate the number of concurrent streams a disk can support. This scheme is used to evaluate bandwidth-latency tradeoffs for a simple disk model. The model is subsequently expanded to consider capacity-bandwidth tradeoffs in ZBR drives. Measurement studies that validate this approach are described for real disk systems. Finally, a cost model for evaluating bandwidth-latency tradeoffs is considered along with its implications on the design of a CM server architecture.

3.1 Introduction

Existing approaches to CM server design typically select a specific media type (e.g., MPEG-I) to evaluate the performance of a disk systems. A given scheduling mechanism is used to perform session admission control at a disk. A random workload is applied to the disk and the resulting disk throughput is used to evaluate the CM capabilities of the device. Unfortunately, this approach provides little insight into disk performance for CM data when the characteristics of the CM data types are altered. From a designer's perspective a model that can be used to evaluate server performance for a range of data types and user requirements would be more useful. We consider such a model in this chapter and describe several refinements to it.

The approach to solve this design issue is to consider disk scheduling constraints and apply them in the derivation of a disk model that determines the streaming capabilities of the drive. The model is subsequently expanded in Section 3.3 to include ZBR disks. Table 3.1 describes the symbols introduced in this section.

3.2 A Disk Performance Model

The efficiency of disk throughput utilization is affected primarily by the disk characteristics (e.g., average transfer rate and mechanical properties) and the choice of the scheduling mechanism. Other parameters that influence disk utilization include the availability of on-disk buffering and the choice of the disk-host interconnect protocol. Any disk performance model must consider all these factors to evaluate a given disk system.

For ease of management and to simplify disk operation, most CM server

Table 3.1: Disk Parameters

Symbol	**Parameter**
N	number of user sessions
R_d	average disk transfer rate
T_{min}	one-track seek time
T_{max}	end-end seek time
T_{seek}	seek time
T_{data}	time to transfer data into the disk buffer
T_d	average time to transfer a disk block
T_{avg}	average seek time
R_c	client data consumption rate
B	disk transfer size unit
T_c	scheduling interval
T_{rot}	rotational latency
$T_{seek}(x)$	seek time to cross x tracks
a	disk seek constant
b	disk seek constant
c	disk seek constant
Θ	total number of disk cylinders
ϕ	disk constant
μ	system constant

schemes retrieve data from the drive in fixed size units or chunks. Furthermore, physical disk organization always retrieves units in discrete size blocks called *sectors*. The size of a disk sector depends on the host operating system and is typically 512 bytes for most filesystems. Thus disk transfers are always integral multiples of the sector size. For the remainder of our discussion, we assume that data from the drive is always read in fixed sizes and call this unit a *block*.

For a client consumption rate of R_c B/s and a constant disk transfer unit of B bytes, the maximum disk scheduling interval is $T_c = \frac{B}{R_c}$. Exceeding this interval will result in data underruns at the client if we assume that a session is served only once in a scheduling interval and all sessions have an identical bandwidth requirement. Multiple requests for a session can be scheduled within a scheduling interval. However, since we are considering CBR streams, this situation can be reduced to the the single bandwidth case in a straightforward manner [69].

A typical disk transfer is a for phase process consisting of (i) a seek-time T_{seek} during which the disk-head is positioned on the proper track, (ii) a rotational latency T_{rot} for the disk head to reach the correct starting block and (iii) the time to transfer data into the disk buffer T_{data}, and (iv) the time to transfer the data to the host T_h. Of these, T_h is external to the drive and we postpone its consideration to Section 3.2.1. However, as will be shown later, T_h only constrains the maximum disk transfer size and has little impact on raw disk performance. Fig. 3.1 illustrates the three phases in disk transfer [20].

Of the disk latencies, T_{seek} and T_{rot} represent necessary overheads that translate to wasted disk bandwidth. Consequently, most scheduling mechanisms aim to minimize their effects. T_{seek} is usually the most expensive of the disk overheads. T_{data} is also a function of the disk zone from which the data are retrieved. However, in

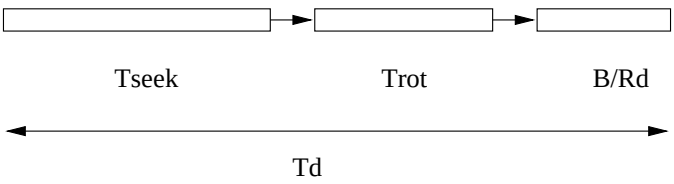


Figure 3.1: The Three Phases of Disk Transfer

the absence of a priori knowledge concerning user access patterns, we approximate this parameter by an average value R_d and therefore $T_{data} = \frac{B}{R_d}$. The average time to transfer a block of data into the disk buffer is then approximated by:

$$T_d = T_{seek} + T_{rot} + \frac{B}{R_d}. \tag{3.1}$$

T_{seek} is a function of the number of tracks spanned by the disk head before it positions itself on the proper track. The seek profiles of a disk are dependent on the disk's geometry and mechanics of operation and are unique to each vendor. As such, each disk has a unique seek profile and extensive studies must be conducted on the physical disk to derive an accurate seek profile for a disk [72]. In the absence of disk availability, the seek times must be approximated from available disk parameters. Disk manufacturers typically supply the average, one-track, and end-end seek times for a drive. When this information is available, the seek times as a function of the seek distance can be estimated using the formula [42]:

$$T_{seek}(x) = a\sqrt{x-1} + b(x-1) + c \tag{3.2}$$

where

$$\begin{aligned} a &= \frac{-10T_{min}+15T_{avg}-5T_{max}}{3\sqrt{\Theta}} \\ b &= \frac{7T_{min}-15T_{avg}+8T_{max}}{3\Theta} \\ c &= T_{min}. \end{aligned}$$

In Eq. 3.2, x is the seek distance measured in number of tracks spanned by the head between consecutive requests. If we assume that sessions are independent (i.e., user requests for content are from independent locations on the disk surface) then the number of concurrent sessions that the disk can support is given by:

$$\left\lfloor \frac{T_c}{T_d} \right\rfloor \qquad \text{or}$$

$$N = \left\lfloor \frac{B}{R_c(T_{seek} + \frac{B}{R_d} + T_{rot})} \right\rfloor, \qquad N \leq \left\lfloor \frac{R_d}{R_c} \right\rfloor. \tag{3.3}$$

Of all the parameters in Eq. 3.3, T_{seek} is unknown. If a simple scheduling scheme such as FCFS is applied to the drive, T_{seek} can be replaced with T_{avg} to determine the streaming capabilities of the drive. For a more efficient scheme such as C-SCAN the disk serves one half of the sessions in one scan and the other half in the reverse scan [62]. In the worst case scenario, sessions are uniformly spread out in the disk surface. Substituting for T_{seek} using Eq. 3.2, the number of concurrent sessions supported by the drive using a C-SCAN approach is given by the solution to

$$N\left(\frac{B}{R_d} + a\sqrt{\frac{\Theta}{N} - 1} + b\left(\frac{\Theta}{N} - 1\right) + c + T_{rot}\right) \leq \frac{B}{R_c}. \tag{3.4}$$

Reorganizing, we obtain

$$aN\sqrt{\frac{\Theta}{N} - 1} \leq \frac{B}{R_c} - b\Theta - N\left(\frac{B}{R_d} - b + c + T_{rot}\right).$$

Squaring both sides and reorganizing, the number of concurrent sessions that the disk can support is obtained by solving for the roots of the quadratic equation:

$$N^2(\phi^2 + a^2) - N(2\phi\mu + a^2\Theta) + \mu^2 = 0 \tag{3.5}$$

where

$$\phi = \frac{B}{R_d} - b + c + T_{rot} \quad \text{and}$$

$$\mu = \frac{B}{R_c} - b\Theta.$$

In deriving Eq. 3.5, we have converted N, which is always an integer into a continuous variable. However, empirical evaluation of Eq. 3.5 demonstrates that the square root term in the solution of the quadratic equation has little influence on the value of N. Consequently, the two roots of the quadratic equation are within ± 1 of each other for a wide range of practical bandwidth values. Mapping N to the integer portion of the lower root always yields a solution that satisfies the constraint in Eq. 3.4. Consequently, we use this observation in determining N.

For small values of R_c, and large values of B, the roots to Eq. 3.5 can be imaginary. This occurs when $\frac{\Theta}{N} < 1$ and the seek times are computed to be negative. In this case, ignoring the square-root term is sufficient to compute the value of N that satisfies the given constraint.

We can approximate N as:

$$N \approx \left\lfloor \frac{2\phi\mu + a^2\Theta}{2 * (\phi^2 + a^2)} \right\rfloor \tag{3.6}$$

when N is large as a consequence of the bandwidth being small. The resulting

value of N is validated against the inequality in Eq. 3.4 to ensure its correctness. Depending on the characteristics of the drive, the value of N as estimated here is greater than the correct value by at most 1 when N is large. We can ignore this error when it is necessary to directly relate B with N. Appendix A2 describes the behavior of N in greater detail.

Fig. 3.2 illustrates the behavior of the scheduling schemes described in Eqs. 3.3 and 3.5 for a reference disk whose parameters are described in Table 3.2. Clearly, Eq. 3.5 represents a better scheduling scheme as evidenced by the figure. However, due to the constraint on N being an integer, both schemes perform similarly for a wide range of scheduling intervals. The gains are more significant for the lowest bandwidth considered (i.e., 1.5 Mb/s).

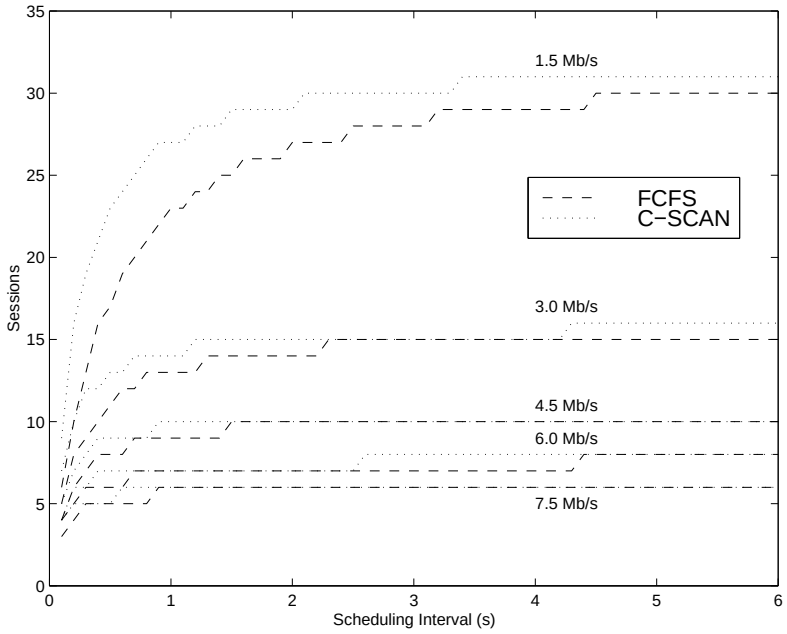


Figure 3.2: T_c vs. N for a Single Disk

Fig. 3.3 illustrates the effective disk bandwidth used by the scheduling mechanism described by Eq. 3.5 as a function of session bandwidth and the scheduling

Table 3.2: Reference Disk Parameters

Symbol	**Parameter**
R_d	6.1 MB/s
T_{min}	1 ms
T_{max}	18 ms
T_{avg}	8.5 ms
T_{rot}	4.17 ms
B_d	1 MB

interval.

We can draw many conclusions from these observations:

- Scheduling is more critical for low bandwidth streams because of the smaller transfer units. Similarly, scheduling is more critical in conventional filesystems where retrievals are typically small.
- Increasing the scheduling interval (or transfer size) does not necessarily increase the number of concurrent sessions supported. Such an increase results in the added penalties of increased system cost due to the cost of additional buffering and increased startup latencies.

In a heavily loaded system with C-SCAN scheduling, the first block of data for a session is delivered to the disk buffer with an average latency $\mathcal{L} = \frac{T_c}{2}$. This delay is because the request for a new data is from an unpredictable location on the disk surface and the insertion of a new request, on average, occurs in the middle of the scheduling interval. For the FCFS scheme, this latency is always equal to $T_c - \hat{N}T_d$

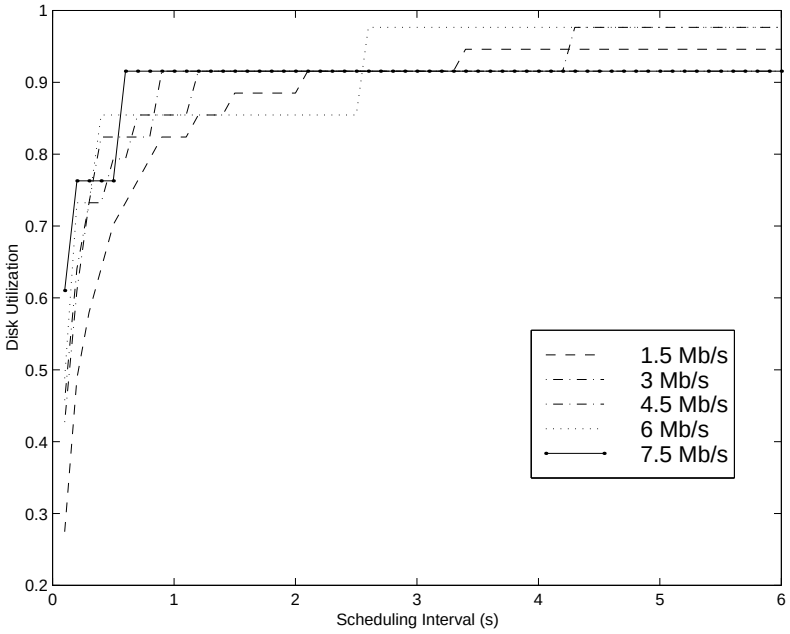


Figure 3.3: Disk Bandwidth Utilization

where $\hat{N}$ is the number of sessions currently in the system. For a client using a dual buffer for rate matching, the average startup latency is given by $\frac{T_c}{2} + T_c$ or

$$\mathcal{L} = \frac{3 * T_c}{2}. \tag{3.7}$$

Similarly, the amount of buffering necessary to support a single stream using a dual-buffer scheme is given by

$$\mathcal{B} = 2 * B. \tag{3.8}$$

Thus, supporting more streams from a disk requires increasing the value of $\mathcal{B}$ and results in a greater $\mathcal{L}$. Furthermore, this gain is bound to the maximum value of N as described in Eq. 3.5. The diminishing gains by increasing B for our reference

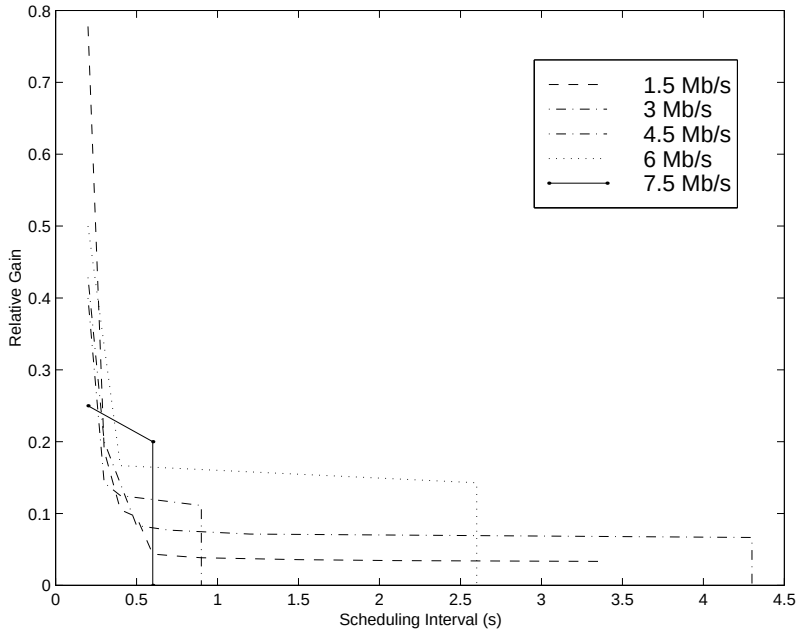


Figure 3.4: Throughput Gains vs. Scheduling Intervals

disk are illustrated in Fig. 3.4. Additionally, since the number of sessions that a disk can support is always an integer, not all the disk bandwidth can be effectively utilized. We now derive an upper limit on the maximum disk transfer size due to the on-disk buffering.

3.2.1 Disk Buffering Constraints

Most modern disk drives are equipped with rate-matching buffers so they can transfer data at maximum speeds over the host-to-I/O bus. This buffering also enables multiple drives to share the bus bandwidth. Arbitrarily increasing the disk transfer size can result in under utilization of the host-to-I/O bus as illustrated below. Table 3.3 describes the symbols introduced in this section.

Let R_{IO} be the I/O bus bandwidth. For simplicity, we assume that $R_{IO} \geq R_d$.

Table 3.3: Disk Buffer Parameters

Symbol	**Parameter**
R_{IO}	disk-host I/O bus transfer rate
B_d	on-disk buffer capacity
B_{min}	minimum disk transfer unit
B_{max}	maximum disk transfer unit
R_e	effective disk transfer rate

The minimum amount of data that must be buffered on the disk to achieve maximum I/O bus throughput is given by

$$B_{min} = B(1 - \frac{R_d}{R_{IO}}). \tag{3.9}$$

In other words, an amount of data proportional to the rate differential between the two devices must be buffered on the disk. However, modern disks employ zone bit recording which results in different transfer rates from different areas of the disk. By approximating R_d to be the maximum disk transfer rate we ensure that data are always transferred at the burst transfer rate.

If B_d is the size of the disk buffer, $B > B_d$, and we assume that the disk-buffer is dual ported (i.e., it can be written to and read from at the same time), the maximum amount of disk data that can be transferred at the full I/O bus rate is given by

$$B_{max} = B_d \frac{R_{IO}}{R_{IO} - R_d}. \tag{3.10}$$

Sustaining this transfer rate requires the disk buffer to be filled before transfer

on the bus begins and for the disk to continue feeding data into the buffer until it is completely drained. From this analysis, we compute the effective transfer rate of the I/O bus as

$$R_e = \begin{cases} R_{IO} & : \quad B \leq B_{max} \\ \frac{BR_{IO}R_d}{BR_{IO} - B_{max}(R_{IO} - R_d)} & : \quad B > B_{max} \end{cases} \tag{3.11}$$

The resulting I/O bandwidths for the reference disk are illustrated in Fig. 3.5 for an I/O bus whose transfer speed is 10 MB/s and maximum disk transfer rate of 6.1 MB/s. It can be seen that the effective I/O transfer rate approaches the maximum disk transfer rate beyond the bound expressed in Eq. 3.10. Modern disks will fragment large data transfers into multiple transfers and disconnect from the I/O bus between transfers. This behavior enables efficient use of the I/O bus even with large reads. However, this situation is analogous to a disk model with smaller transfer units and results in reduced disk utilization.

Consequently, we derive an upper bound on the number of concurrent sessions that a disk can support by replacing B with the value of B_{max} derived in Eq. 3.10 into Eq. 3.3:

$$N_{max} = \left\lfloor \frac{B_{max}}{R_c(T_{seek} + \frac{B_{max}}{R_d} + T_{rot})} \right\rfloor, \qquad N \leq \left\lfloor \frac{R_d}{R_c} \right\rfloor. \tag{3.12}$$

Fig. 3.6 illustrates the maximum number of sessions supported from the disk as described in Eq. 3.12. Only a fraction of the disk bandwidth is effectively utilized even when a maximum number of sessions are supported for a given bandwidth.

Fig. 3.7 illustrates the fraction of the disk bandwidth that is utilized when a maximum number of concurrent sessions are supported for a given bandwidth. The

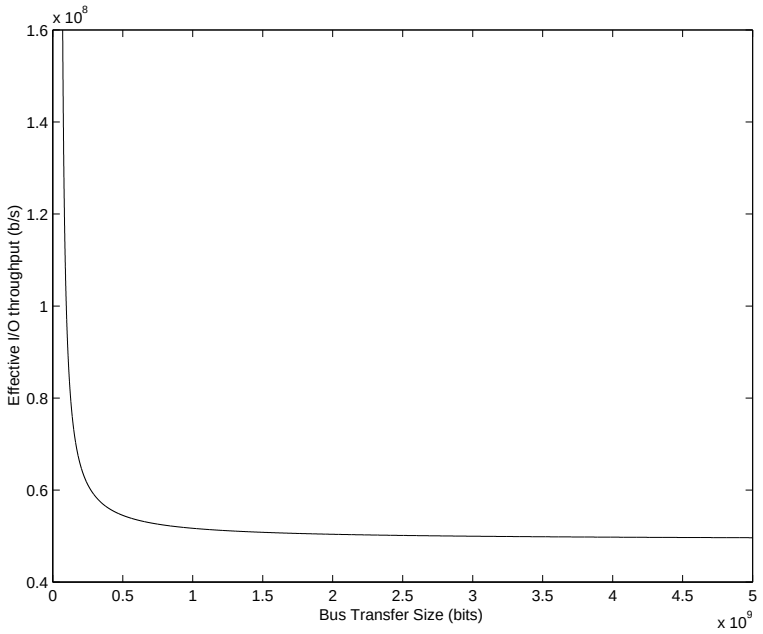


Figure 3.5: Effective I/O Bandwidth for a Single Disk (Analytical)

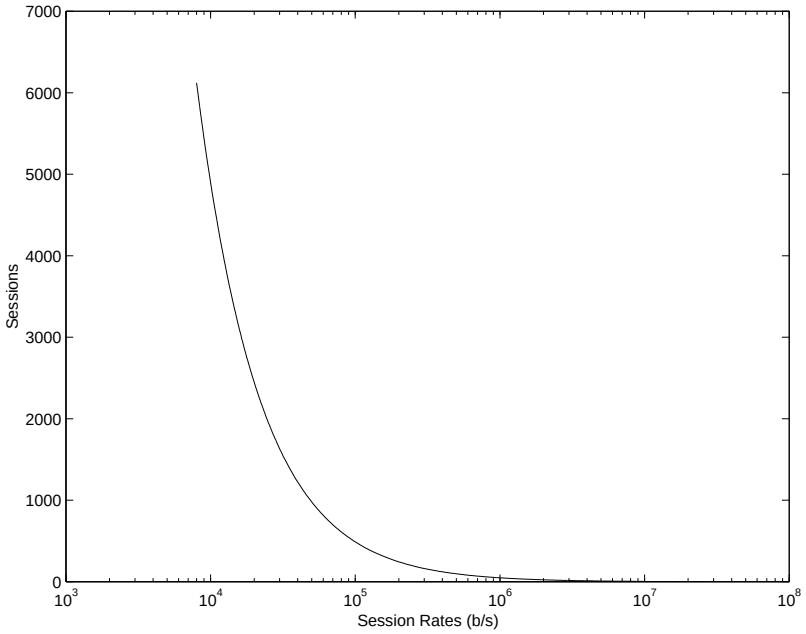


Figure 3.6: Maximum Number of Sessions vs. Session Bandwidth

achievable disk utilization is highest for the low bandwidth streams and less for the high bandwidth streams due to the higher packing density of low bandwidth streams. A fraction of the high bandwidth streams achieve high disk utilization when the achievable disk bandwidth is close to an integral multiple of the session bandwidth.

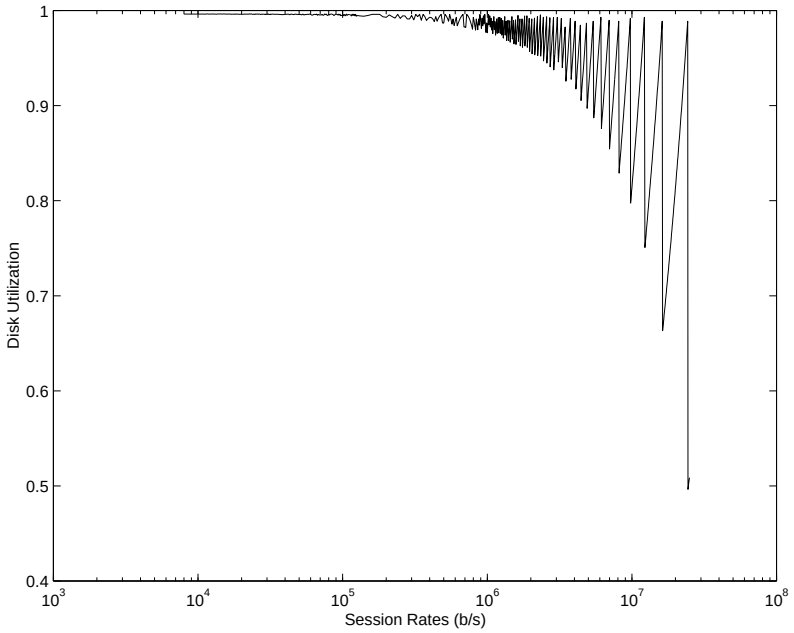


Figure 3.7: Maximum Disk Utilization

However, in either case, achieving optimal disk utilization requires a higher T_c, resulting in a large buffer and the subsequent startup latency. Fig. 3.8 illustrates the scheduling interval T_c necessary to support N_{max} sessions (i.e. maximum disk utilization).

Achieving a high disk utilization comes at an increased penalty of an extremely large scheduling interval for low bandwidth streams. Later, we describe the use of a cost model to evaluate the latency-utilization tradeoffs for a disk drive. We make the following observations:

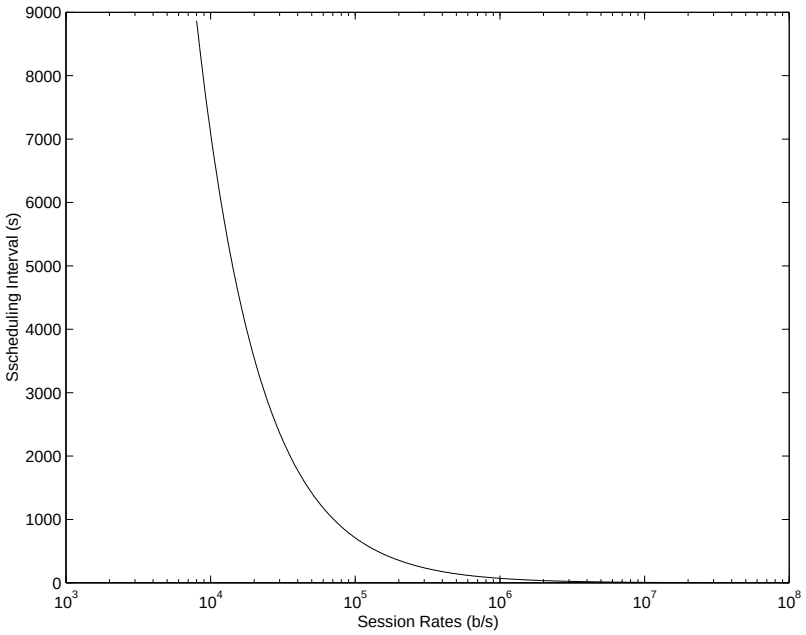


Figure 3.8: Scheduling Intervals to Achieve Maximum Disk Utilization

- Increasing the buffer size B increases the number of concurrent sessions; the penalty is an increased startup latency.
- When the session bandwidth is high, the startup latencies (response times) are lower.
- The relative gains due to increasing B decrease quickly and these gains are minimal for high bandwidth streams (diminishing returns).
- The gains achieved by increasing B are offset by the requirements of a larger buffer $\mathcal{B}$ per stream.

Until now, we have focused our attention on applying the disk model to accurately estimate the session streaming capability of the disk when the entire disk surface is used. However, as illustrated in Fig. 3.9, the bandwidth of the drive can vary significantly over different drive areas due to the disk geometry.

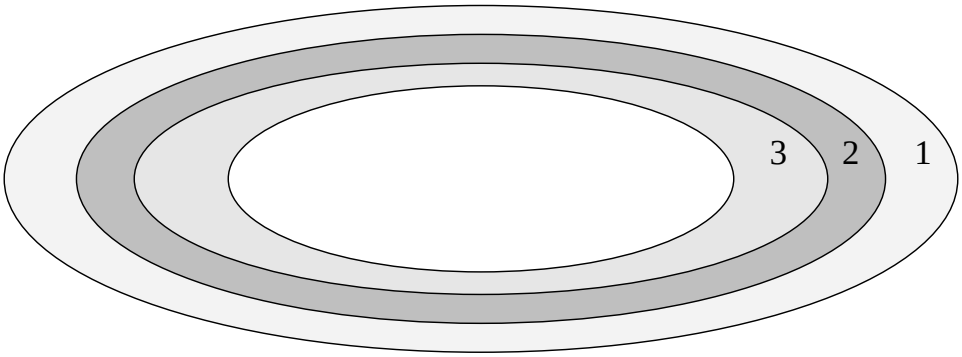


Figure 3.9: Disk Geometry and Zone Bit Recording

Modern drives make better use of available disk surface using a nearly constant recording density to store data on the entire disk surface. This results in increased disk capacity and transfer rates. However, the resulting disk transfer-rate is not uniform for all areas on the disk. To ease management of data, disk drives are partitioned into zones and the transfer rate within each zone is fixed. For example, in Fig. 3.9, we illustrate a disk surface with three zones. Zone 1 has the highest transfer rate and zone 3 has the lowest transfer rate.

This zoning design leads us to explore the possibility to use the disk selectively to support a larger number of sessions by using only a part of the disk (which reduces both seek overheads and utilizes the disk bandwidth more efficiently). In the next section, we describe a disk model that builds on the work in this section to explore this tradeoff.

3.3 Performance Tradeoffs in ZBR Disks

In this section, we explore data-layout policies for ZBR drives that tradeoff capacity for bandwidth to meet a given capacity-I/O requirement. In particular, we are in-

terested in the feasibility of "short-stroking" a disk by using using only some of the outer zones to deliver data at a higher rate and support more concurrent sessions at the expense of storage capacity. Table 3.4 describes the symbols introduced in this section.

Table 3.4: ZBR Disk Parameters

Symbol	**Parameter**
Z	number of zones
R_{Zi}	disk transfer rate for zone i
Θ_i	number of tracks in zone i
S_i	sectors per track in zone i
Δ	sector size in bytes
ω	disk rpm
p	number of read/write surfaces (disk platters)
C_i	storage capacity of zone i
Φ_i	ZoneGroup i
α_i	transfer rate for Φ_i
β_i	storage capacity for Φ_i
N_i	number of concurrent sessions supported by Φ_i

Consider a disk with Z zones. The data transfer rate for zone i can be estimated as

$$R_{Zi} = \frac{8 S_i \Delta \omega}{60} \qquad \text{b/s.} \tag{3.13}$$

Similarly, the capacity of zone i, C_i is given by

$$C_i = p \Delta S_i \Theta_i \qquad \text{Bytes.} \tag{3.14}$$

The transfer profile of the disk as derived in Eq. 3.13 is not a function of the number of platters in the system. The number of disk platters only affects the capacity of the disk drive.

Let zone 1 represent the outer zone and zone Z the inner zone. We define a ZoneGroup Φ_i as the collection of disk zones from zone 1 through zone i. Φ_Z represents a ZoneGroup that includes all zones (i.e., the entire disk).

Our decision to always include the outer zones in a ZoneGroup are motivated by the observation that this scheme takes advantage of the higher transfer rates of the outer zones. Furthermore, this definition of a ZoneGroup always selects the best disk structure for a given bandwidth-capacity combination as shown below.

Lemma 1 *Selecting a ZoneGroup that is contiguous from the outer zone always yields the best capacity-bandwidth profile on average.*

Proof: If we select a collection of zones that does not include the outer zones, adding a track from the next outer zone and deleting a track from the innermost zone increases the average drive bandwidth. Thus, we have gained in bandwidth and lost nothing in terms of capacity. Secondly, the outer zones require fewer tracks to store the same amount of data. Hence, the seek distances spanned by an applied load are always smaller on average. This represents a second order increase in bandwidth with no loss in capacity.

The average rate of data transfer for Φ_i is given by

$$\alpha_i = \frac{\sum_{j=1}^{i} R_{Zj}\Theta_j}{\sum_{k=1}^{i} \Theta_k}. \tag{3.15}$$

Similarly, the capacity of Φ_i is given by

$$\beta_i = \sum_{j=1}^{i} C_i. \tag{3.16}$$

The number of concurrent sessions that a disk can support is controlled by the disk seek and rotational overheads as well as the disk transfer rates. It is clear that the seek profile for the disk is a function of Φ. Applying the results from Eq. 3.5, we can estimate the number of sessions supported by a ZoneGroup i as:

$$N_i \left(\frac{B}{\alpha_i} + T_{seek} \left(\frac{\sum_{j=1}^{i} \Theta_j}{N_i} \right) + T_{rot} \right) \leq \frac{B}{R_c} \tag{3.17}$$

In Eq. 3.17 N_i is the number of concurrent sessions streamed from Φ_i, B is the data retrieved for each user session in one scheduling interval, and R_c is the rate of data consumption by the user.

A solution for N_i is obtained by substituting T_{seek} with the seek profile described in Eq. 3.5 by solving for the roots of

$$N_i^2(\phi^2 + a^2) - N_i(2\phi\mu + a^2\hat{\Theta}_i) + \mu^2 = 0 \tag{3.18}$$

where

$$\phi = \frac{B}{\alpha_i} - b + c + T_{rot} \qquad \text{and}$$

$$\mu = \frac{B}{R_c} - b\hat{\Theta}_i.$$

This value of N_i can be used to determine the number of sessions a disk can support in a given ZoneGroup. We now describe the use of this information to study capacity bandwidth tradeoffs in a ZBR drive.

3.3.1 Trading Capacity for Bandwidth in a ZBR Disk Drive

Consider a CM server system that must support K sessions and M media objects. We assume, for simplicity that all sessions are identical, each with a bandwidth requirement of R_c b/s. We identify two constraints based on our earlier discussion. The capacity constraint is given by the object size distribution and the bandwidth constraint is given by the total user bandwidth requirement.

In Section 3.3, we developed an improved performance model for ZBR disks. We now apply this model to the design of a large scale server model. For a given Φ_i, the number of disks necessary to support a given object size distribution can be estimated as

$$D_{ci} = \left\lceil \frac{\sum_{j=1}^{M} o_j}{\beta_i} \right\rceil \tag{3.19}$$

where o_j is the capacity requirement of object j.

Similarly, the number of storage devices necessary to support K concurrent user sessions is given by

$$D_{bi} = \left\lceil \frac{K}{N_i} \right\rceil \tag{3.20}$$

where N_i is obtained by solving Eq. 3.18.

For a given system both the capacity and storage requirements must be satisfied. Given a ZoneGroup Φ_i, the number of disks necessary to satisfy both the capacity and bandwidth constraints is given by

$$D_i = max(D_{ci}, D_{bi}).$$

The optimal configuration for the system can be chosen by selecting the ZoneGroup that minimizes the number of disks required. The optimal ZoneGroup $\hat{\Phi}$ is given by

$$\hat{\Phi} = \Phi_i, \qquad (D_i \leq D_j \quad \forall i, j). \tag{3.21}$$

It is clear that $\hat{\Phi}$ is affected by the choice of B. Therefore, in choosing $\hat{\Phi}$, the system must be evaluated for all operational values within the allowed range of B for one that minimizes $\hat{\Phi}$ while simultaneously minimizing B. This allows us to evaluate the feasibility of using ZBR tradeoffs to design video servers.

3.4 Performance Measurements and Validation

To test the correctness of the disk-streaming models described in this chapter, we conducted measurement studies on two real disk systems. The results described below are for the Seagate^{TM} Hawk (ST11200ND) 1GB drive, and the Seagate^{TM} Barracuda (ST15150N) [58, 59]. The drives represent two different generations of disk drives, the Barracuda being the more recent and modern of the two. Appendix A1 gives details of the disk drive parameters according to Seagate specifications.

3.4.1 Single Disk Measurement and Evaluation

In the first experiment we conducted, the disk data were read contiguously and linearly, from the outer to the inner cylinders. Continuous disk activity was ensured by keeping the disk request buffer always full. The test was designed to wrap around and continue reading data from the first disk block as soon as it completed reading

the last disk block.

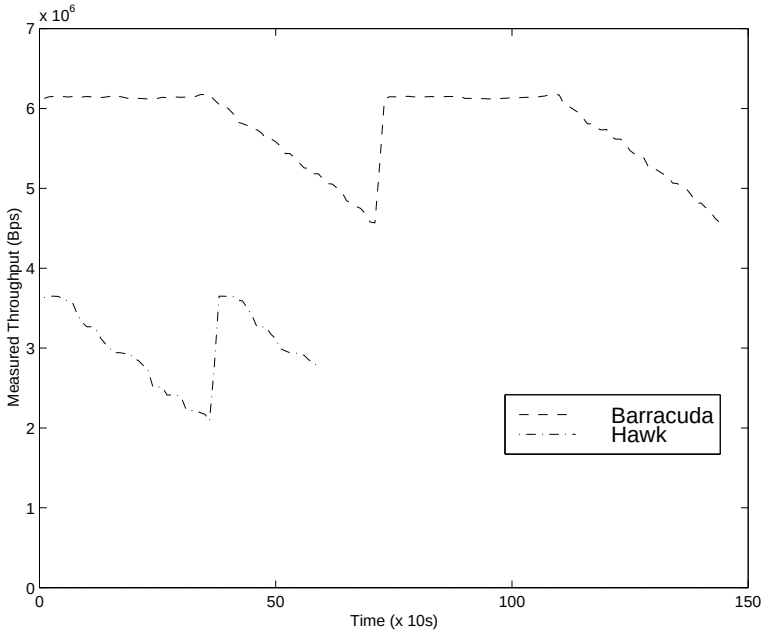


Figure 3.10: Disk Linear Read

The measured disk throughput over time is illustrated in Fig. 3.10. The periodic behavior of the graph is attributed to the wrap-around nature of the test. Within a single read cycle, it is clear that the disk transfer rate is non-uniform. Furthermore, the effects of ZBR are immediately apparent. The initial disk throughput is high, representing data transfer from the outer zones and decreases as the test proceeds.

The average disk throughput as a function of the disk transfer size, using the data from the linear reads is represented in Fig. 3.11 as a function of the request size. We observe that the disk throughput is constant only when the request sizes are large. For small transfers, the switching and I/O overheads preclude optimal utilization of the disk, even with zero seeking overhead.

The aggregate disk throughput for the C-SCAN and FCFS scheduling schemes

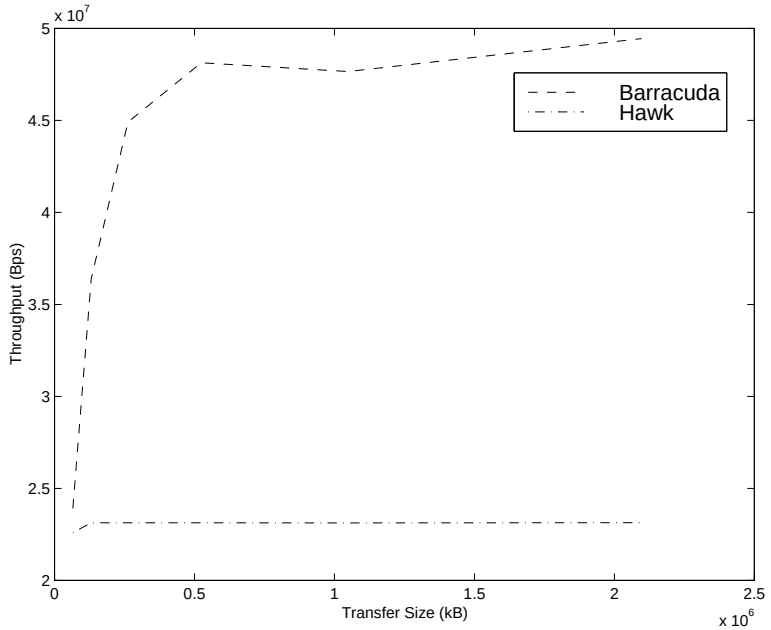


Figure 3.11: Average Disk Throughput vs. Transfer Size

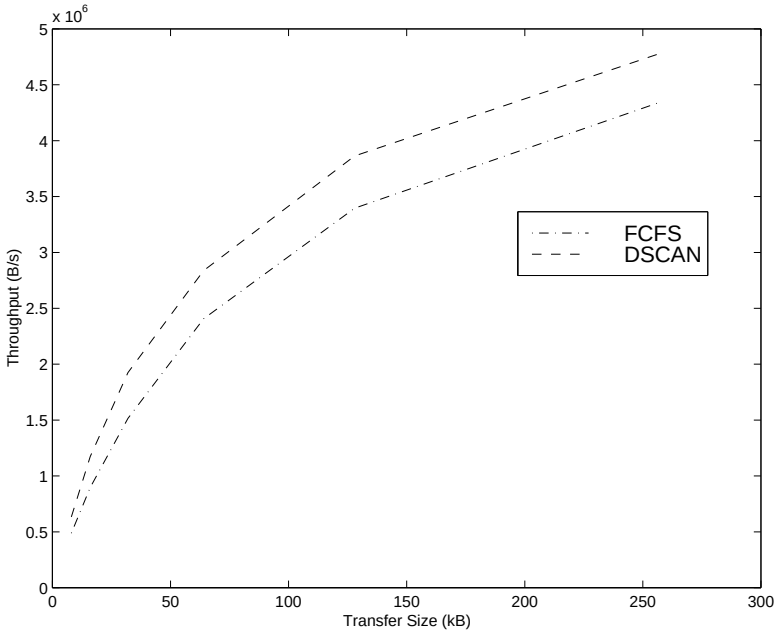


Figure 3.12: Scheduling Gains Comparison

as a function of the transfer size is illustrated in Fig. 3.12. Clearly, C-SCAN is a better disk scheduling mechanism. However, the relative scheduling gains decrease for large transfer sizes.

The streaming throughput is lesser than the maximum measured throughput due to the constraint on the number of sessions being an integer. The model can be further constrained by noting that for real disks, transfers always take place in discrete units called disk blocks. The size of a disk block varies depending on the system under use. The value of N as given by Eq. 3.5 is well suited to determine an admission control policy for the disk system. This model can also be used in capacity planning to determine the number of disks necessary to support a given throughput and session bandwidth requirement.

When streams with different bandwidth requirements must be supported, the model can be easily extended [55]. There have been several studies in the recent past that have accurately characterized disk behavior that can be used to evaluate exact disk performance [57, 72]. These models can be easily applied to compute the additional overheads that must be included in our analysis. Appendix A3 describes the computation of overheads for inclusion in the disk model.

3.4.2 ZBR Measurements and Evaluation

In order to determine the ZBR characteristics of the disk and demonstrate the validity of our tests, we conducted measurement studies on both the Barracuda and Hawk Drives. Our experiments consisted of selecting ZoneGroups for each of the drives and running throughput measurements for both the FCFS and C-SCAN scheduling policies.

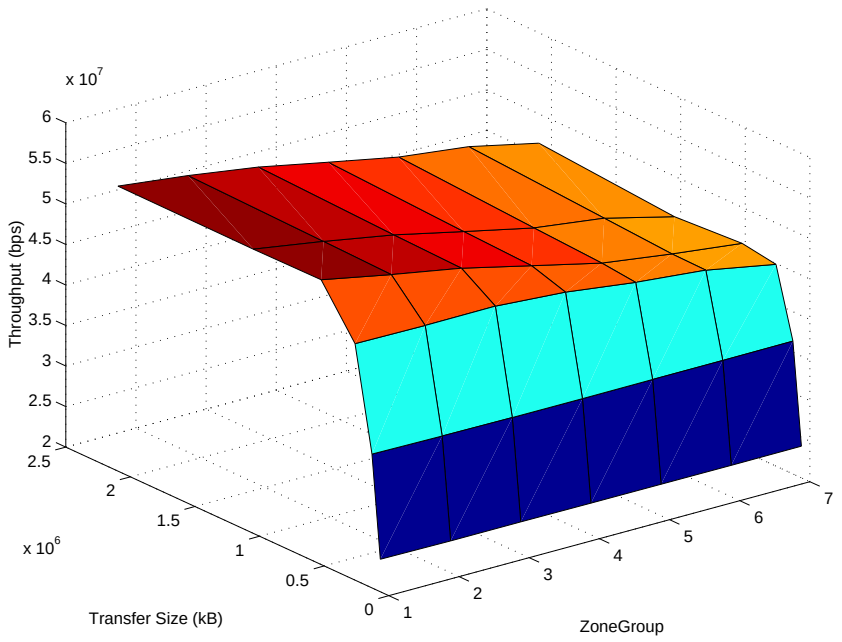


Figure 3.13: ZBR Profile of the Barracuda Drive

The I/O bandwidth and the corresponding disk capacity for the Barracuda drive as a function of the ZoneGroup are illustrated in Figs. 3.13 and 3.14, respectively. It can be seen that the capacity and I/O curves have opposite effects, as expected. As more zones are included in a ZoneGroup, available storage increases but the average disk bandwidth is reduced. It is clear that it is indeed feasible to tradeoff capacity for bandwidth using this profile.

The tradeoff in storage requirement as a function of the scheduling interval and the ZoneGroup for MPEG-II streams with a 1,000 user 100 movie requirement is illustrated in Fig. 3.15. The resulting optimal ZoneGroup ($\hat{\Phi}$) is a surface that corresponds to the maximum value of the capacity and bandwidth surfaces. The optimal storage configuration is selected by choosing the point on the surface that corresponds to the least number of disks and the smallest scheduling interval.

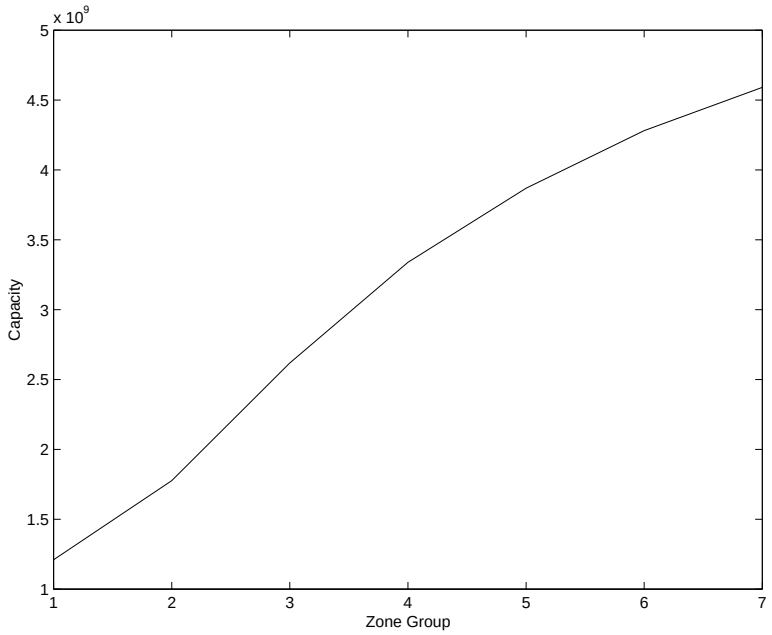


Figure 3.14: Disk Capacity Profile

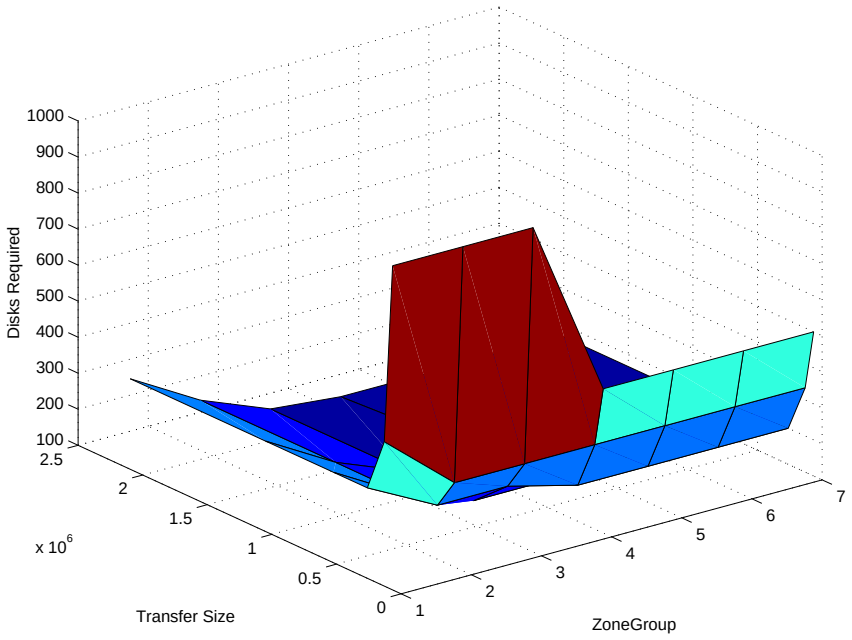


Figure 3.15: Capacity-Bandwidth Tradeoffs for the Barracuda

In order to test the capacity-bandwidth tradeoffs using the models proposed in Section 3.3.1 we applied the ZBR mapping structure derived in the earlier sections to evaluate the requirements for storage and capacity for 5 different workload considerations; (i) 100 users and 10 objects, (ii) 1,000 users and 100 objects, (iii) 10,000 users and 500 objects and (iv) 10,000 users and 1,000 objects, and (v) 100,000 users and 10,000 objects. We consider MPEG-1 (1.5 Mb/s) and MPEG-II (6 Mb/s) rates at an average object length of 90 minutes and 64 Kb/s objects at 10 minutes each. For each situation, we evaluated the best ZBR configuration.

The minimum storage requirements as obtained from the technique described above are illustrated Tables 3.5, 3.6 and 3.7 for the Barracuda drive. The number of disks necessary to support the given load with zone optimization is represented by D_o and the corresponding retrieval unit size by T_o. Similarly, the number of disks necessary to support the requisite load and capacity without any zone optimization is given by D_n and T_n respectively.

Table 3.5: Storage Requirements for MPEG-II streams

Users	**Objects**	D_o	T_o	ZO	D_n	T_n	**% less**
100	10	15	256K	4	17	128K	13
1000	100	143	256K	4	167	128K	14.3
10000	500	1429	128K	2	1667	128K	14.3
10000	1000	1429	256K	4	1667	128K	14.3
100000	10000	14286	128K	1	16667	128K	14.3

The tables demonstrate that the ZBR information can be selectively chosen to optimize system performance depending on the system requirements. It is clear that depending on the workload, significant reduction in the number of disks necessary to

Table 3.6: Storage Requirements for MPEG-I streams

Users	**Objects**	D_o	T_o	ZO	D_n	T_n	**% less**
100	10	4	128K	3	4	128K	0.0
1000	100	34	256K	4	36	128K	5.5
10000	500	313	256K	2	358	128K	12.56
10000	1000	334	256K	4	358	128K	12.56
100000	10000	3125	256K	1	3572	128K	12.51

Table 3.7: Storage Requirements for 64 Kb/s streams

Users	**Objects**	D_o	T_o	ZO	D_n	T_n	**% less**
100	10	1	8K	1	1	8K	0.0
1000	100	2	64K	1	2	64K	0.0
10000	500	14	256K	1	16	128K	12.5
10000	1000	14	256K	1	16	128K	12.5
100000	10000	132	256K	1	153	128K	13.72

support the desired workload is possible. This result validates our earlier assumption that it is possible to tradeoff bandwidth with storage to design disk-based video servers.

However, in evaluating the capacity-bandwidth tradeoff of the ZBR drive, we ignored the requirements of buffering and the corresponding startup latency. In the next section, we consider a cost model to evaluate the effects of these constraints on server performance.

3.5 Bandwidth-Latency Tradeoffs

In the earlier sections we established the performance requirements and I/O device limitations for CM data types. We demonstrated the capacity-bandwidth tradeoffs that are feasible using the ZBR characteristics of modern disk drives.

CM server performance is measured in terms of the conflicting requirements of access latency and utilization. We now describe a system cost model for evaluating this tradeoff. The proposed model uses *cost per unit bandwidth* as metric to evaluate the effectiveness of choosing a given disk transfer size on system cost. Existing disk and memory prices are used as parameters in the model to justify its utility.

Let C_B and C_d represent the unit costs (cost/bit) of buffer and disk storage respectively. Disk storage costs can be assumed to increase linearly with increasing capacity requirements. Using the dual-buffer model, the buffering requirements for each session supported from the drive can be assumed to be at least twice the size of the disk transfer. Consequently, the cost of buffering for each session served from the server is $2BC_B$. The cost of disk storage is independent of the number of sessions and is given by DdC_d where d is the disk capacity and D is the total number of disks

in the system. If the server system can support N contiguous sessions, the cost per stream C_N is given by

$$C_N = 2BC_B + \frac{DdC_d}{N}. \tag{3.22}$$

In this model, we have neglected the disk connectivity cost assuming that it can be included in C_d. C_N is affected by the choice of D, C_B, and C_d.

While cost-per-stream C_N is interesting to the end user, it is not a fair measure as different streams result in different values of C_N. A more reasonable cost measure is the price per unit bandwidth $C_u = C_N/R_c$. From Eq. 3.22, we can derive C_u as

$$C_u = \frac{2BC_B}{R_c} + \frac{DdC_d}{NR_c}. \tag{3.23}$$

Fig. 3.16 illustrates the behavior of C_u for a range of C_B values using the parameters of the Barracuda drive described earlier. It is clear that the cost function thus defined can identify the buffer size that yields the least C_u.

The fraction of buffer cost as a function of the overall cost is illustrated in Fig. 3.17. It appears that the minimum cost point corresponds to the point when the buffer cost is approximately equal to one-half the total system cost.

The primary constraints on C_u are the limits imposed by B. Clearly, the objective of server design is to minimize C_u subject to the condition that B (and hence T_c) is below a given value. Replacing N using the bound derived in Eq. 3.6 we obtain

$$C_u = \frac{2BC_B}{R_c} + \frac{DdC_d * 2(\phi^2 + a^2)}{R_c(2\phi\mu + a^2\Theta)}. \tag{3.24}$$

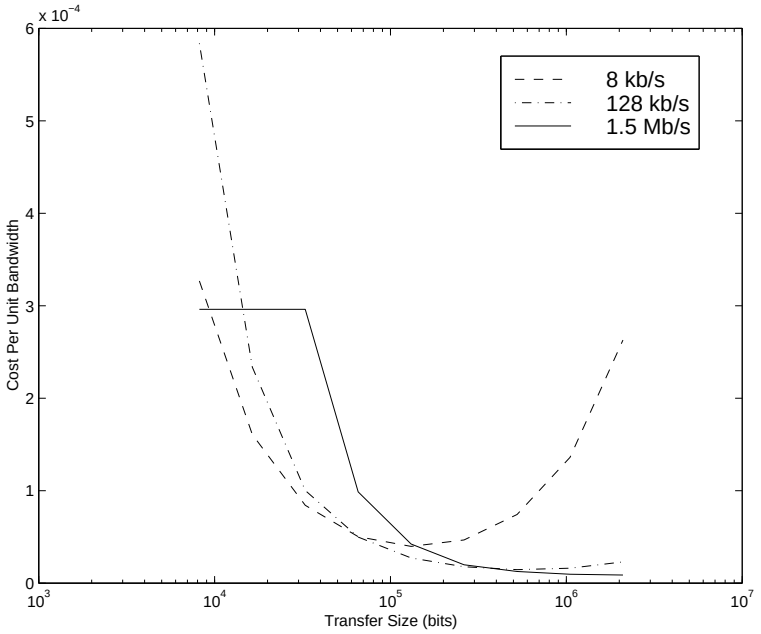


Figure 3.16: C_u vs. B

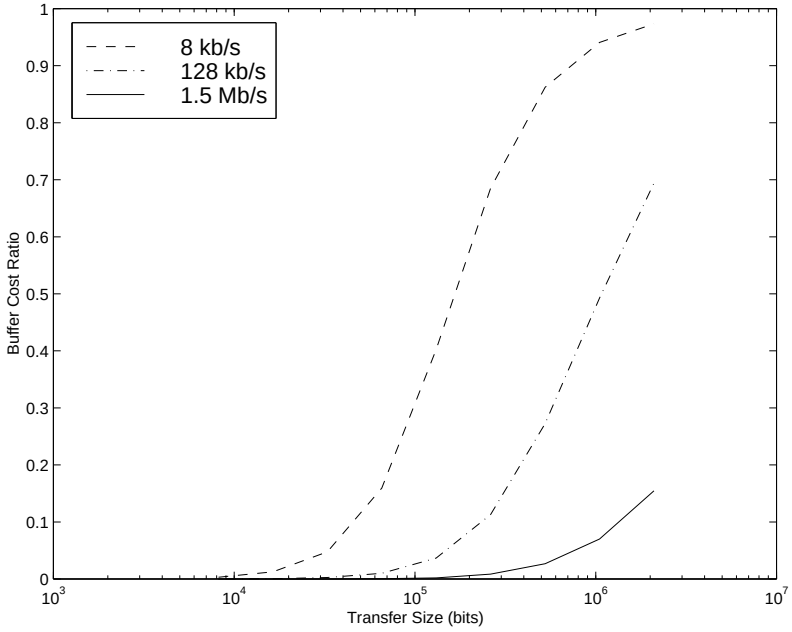


Figure 3.17: Buffer Cost Ratio

Eq. 3.24 is only dependent on B. The first part of Eq. 3.24 is linearly dependent on B whereas the second is inversely dependent. Thus, we derive the minimum value of C_u for a given C_B by setting:

$$\frac{dC_u}{dB} = 0. \tag{3.25}$$

The resulting B is obtained by solving

$$\begin{aligned} & B^4(-8C_BR_d) + \\ & B^3(16C_BR_d\mathcal{M}) + \\ & B^2(2\mathcal{MS} - 4\mathcal{S}kR_d - 8\mathcal{M}^2C_BR_d - 8C_BR_d\mathcal{T}) + \\ & B(8\mathcal{LS} - 2\mathcal{ST} + 8C_BR_d\mathcal{MT}) + \\ & 2kR_d\mathcal{ST} - 2C_BR_d\mathcal{T}^2 - 2\mathcal{MSY} = 0. \end{aligned} \tag{3.26}$$

Appendix A4 describes the derivation of Eq. 3.26 and the behavior of B. As described in Appendix A4, Eq. 3.26 has only one positive root which maps to the desired transfer size.

However, minimizing C_u does not address the latency issue. Because startup latency increases linearly with increasing B, any solution to Eq. 3.26 is bound by the latency requirement. If the latency constraint intersects the cost curve before the minimum is reached, the optimal operating point is the buffer size that corresponds to the desired latency. If it is to the right of the minimum, we can use the minimum latency requirement.

For disk arrays, the cost function can be used exactly as shown without any penalty when the data layout is block-interleaved and each disk is accessed inde-

pendently. However, for bit or byte-interleaved systems, the cost function must be rewritten as

$$C_u = \frac{2BDC_B}{R_c} + \frac{DdC_d}{NR_c} \tag{3.27}$$

to account for the fact that the buffer size must be sufficient for at least twice the amount of data retrieved in one scheduling interval. It is clear from this model that the buffering costs dominate the cost function at low latency requirements and disk costs dominate at high latency requirements. We now consider results from applying this model to disk systems and the resulting performance implications.

3.5.1 Application of the Cost Model

To understand the implications of the proposed cost model, we evaluated the performance of several disk organizations assuming a C_B of \$4/MB and a C_d of \$0.1/MB. The cost model was evaluated for a wide range of session bandwidth requirements and disk scheduling intervals using the parameters of the Barracuda drive described earlier.

Effects of Disk Capacity

Figs. 3.18 - 3.22 illustrate the minimum achievable C_u and the corresponding system characteristics using the parameters of a Barracuda drives and for capacities of 2.4GB, 4.5GB and, 9GB respectively. For each capacity configuration, the disk parameters are identical and only differ in the number of disk-platters.

The cost per unit bandwidth as a function of session rate and scheduling

interval is illustrated in Fig. 3.18. We observe that C_u is nearly an order of magnitude higher for the 8 Kb/s streams when compared with the 1.5 Mb/s streams. This discrepancy is due to the greater significance of the buffering costs. This also implies that the lower bandwidth streams will suffer from a higher startup latency in general. It is clear that for disks with identical physical properties, it is advantageous to use multiple disks with smaller capacity to take advantage of the gains in bandwidth.

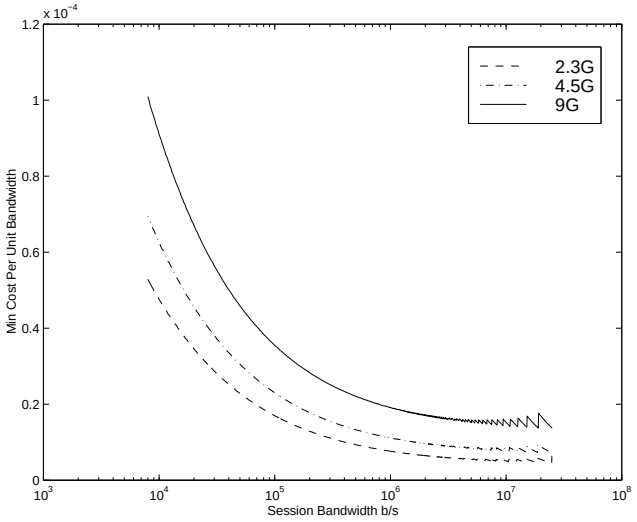


Figure 3.18: C_u vs. R_c

The minimum cost curve is non-monotonic towards the high-bandwidth end. This behavior is due to the lower count of the number of streams that can be supported from the disk drive. Consequently, the effective disk utilization fluctuates enormously, causing the observed behavior. The figure demonstrates that disk-based server architectures are better for serving high bandwidth sessions. This is because at high bandwidths fewer streams are served, which lowers the frequency of disk seeks. As a result, overheads are reduced and sessions can be supported for low values of T_c, reducing the buffering requirements.

The transfer size B to achieve the minimum C_u is illustrated in Fig. 3.19. We

observe that the transfer sizes are enormous for the high bandwidth streams and are impractical. This behavior is a fallout of our desire to minimize C_u. This yields a large N and a high value of B. Consequently, for large values of R_c, the value of B must be constrained.

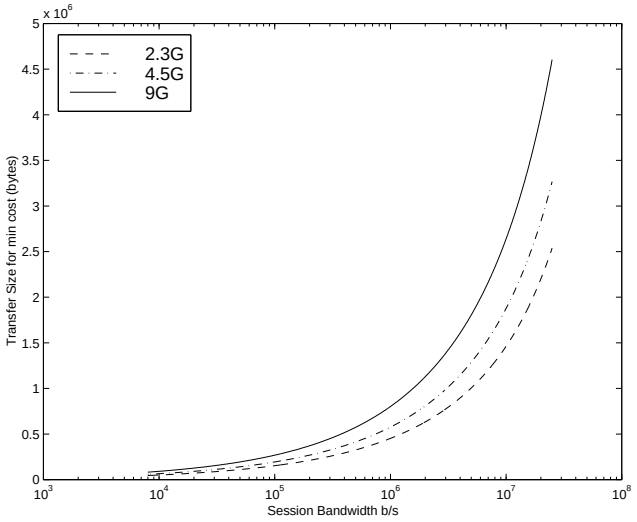


Figure 3.19: B vs. R_c for Minimum Cost

The scheduling interval ($T_c = B/R_c$) as a consequence of minimizing the system cost is illustrated in Fig. 3.20. We immediately note that the low bandwidth streams require large scheduling intervals to achieve a minimum cost. This requirement directly maps to a significant startup latency during periods of high system utilization.

Disk bandwidth utilization ($T_c = B/R_c$) due to minimizing C_u is illustrated in Fig. 3.21. It is clear that low bandwidth streams poorly utilize available disk bandwidth. The effective disk bandwidth utilization increases nearly monotonically and fluctuates rapidly for high bandwidth streams. This behavior is due to the sessions that the disk supports, causing rapid fluctuations in disk utilization. This leads us to an interesting observation that under high loads, the storage system is

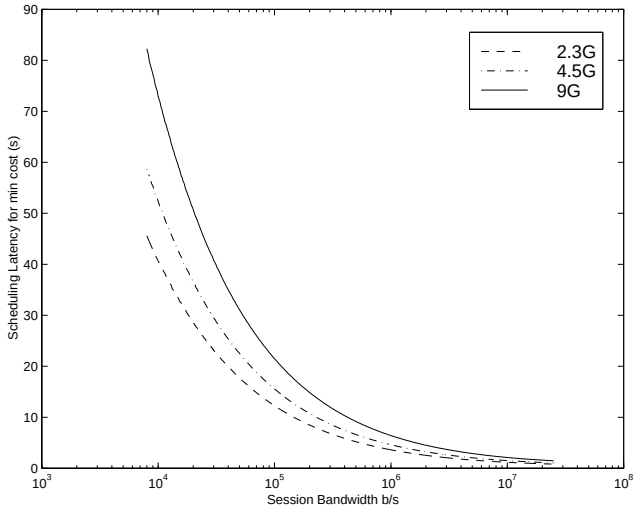


Figure 3.20: T_c vs. R_c for Minimum C_u

able to respond much more quickly to user interactions when serving high bandwidth sessions.

Finally, we illustrate the number of sessions N supported at the minimum value of C_u in Fig. 3.22.

In analyzing the cost model, we have considered ideal disk bandwidth parameters (i.e., we assumed that the disk always transfers data at its average bandwidth). However, as described in the earlier measurement studies, effective disk bandwidth is directly proportional to the disk transfer size. This constraint further increases the cost of streaming low bandwidth streams.

We also observe that the small capacity drive (2.3GB) provides the best C_u and the smallest T_c. Consequently, it is preferable to choose smaller drives as they provide the best C_u. Furthermore, for an identical capacity requirement, the smaller drive provides a higher throughput.

Figs. 3.23 and 3.24 illustrate the effects of constraining T_c to be less than 2s.

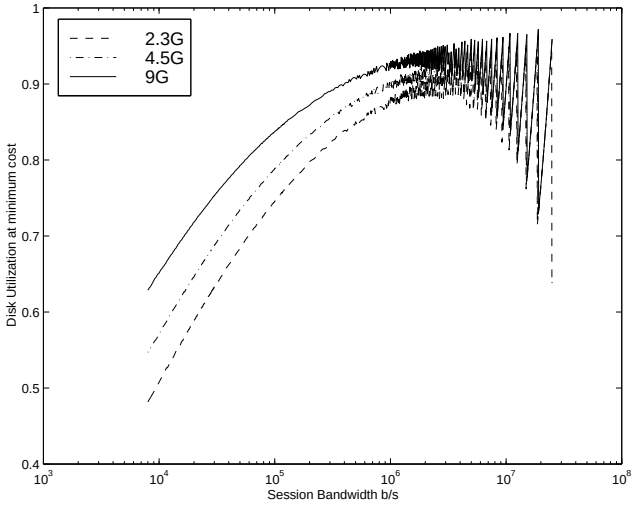


Figure 3.21: Disk Utilization vs. R_c for Minimum C_u

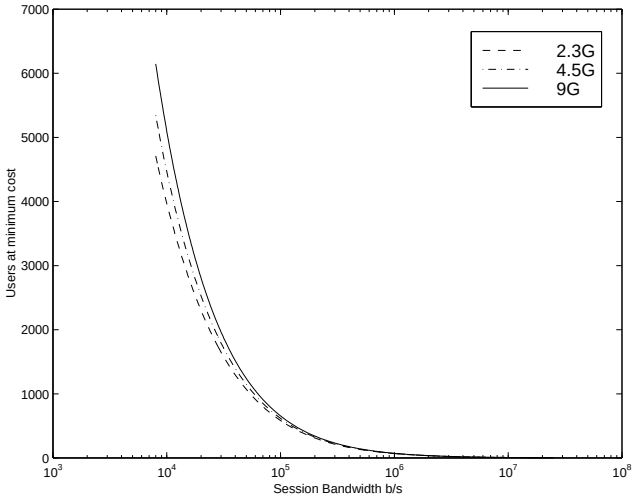


Figure 3.22: N vs. R_c for Minimum C_u

We observe that C_u is significantly higher for the low bandwidth streams under this constraint.

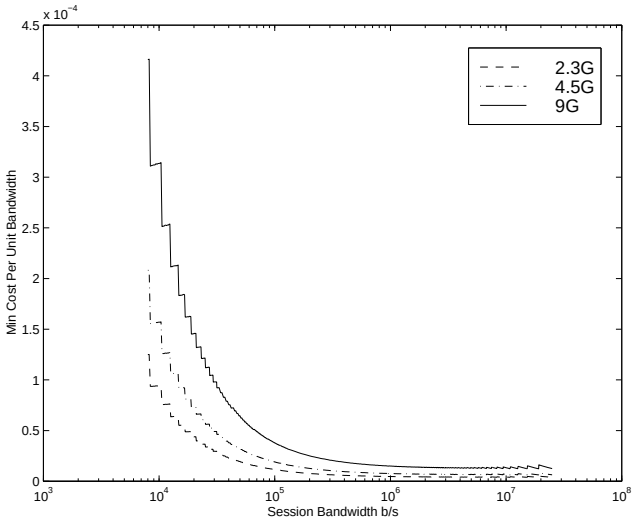


Figure 3.23: Minimum C_u vs. R_c when $T_c < 2s$

However, we observe that constraining T_c results in the disk capacity having no effect on the number of concurrent sessions that the disk can support. This is because constraining T_c to low bandwidths yields identical transfer sizes that are not dependent on disk capacity as cost is no longer the main factor.

These results illustrate the pitfalls of assuming that storage performance characteristics scale linearly with session bandwidth and latency requirements. Buffering costs are small for low bandwidth streams, however they increase with reduced latency requirements. High bandwidth streams can be served at low latencies without significantly changing the buffering and cost parameters.

It is apparent that based on an operating cost it is possible to specify scheduling intervals and disk layout policies that can best utilize available resources and satisfy the requirements of a maximum number of users. This knowledge can be used to

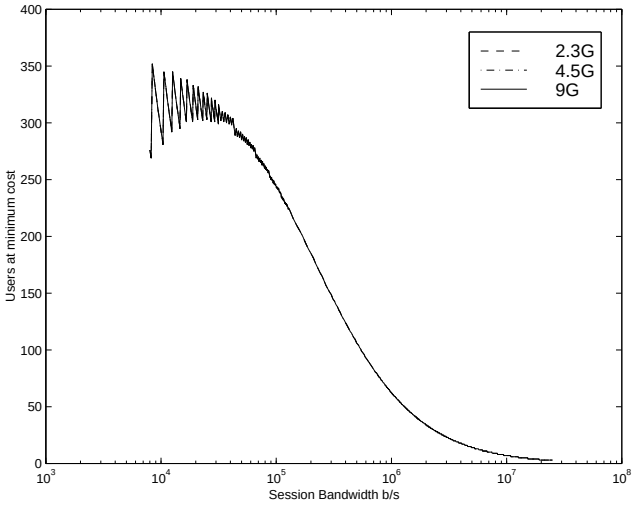


Figure 3.24: Minimum N vs. R_c when $T_c < 2s$

develop a scalable disk scheduling policy where the user may benefit from a better quality of service at lower cost.

3.5.2 Application in ZBR Drives

Application of the cost model to evaluate capacity-bandwidth tradeoffs for the ZBR model described in Section 3.3 requires the cost model to be modified. Since a ZoneGroup utilizes only part of the disk capacity, C_d must be scaled appropriately for inclusion in the cost equation. We define the modified cost equation for a ZoneGroup i as

$$C_{ui} = \frac{2BC_B}{R_c} + \frac{DdC_{di}}{NR_c} \tag{3.28}$$

where

$$C_{di} = \frac{C_d \beta_Z}{\beta_i}.$$

The operating point that yields the least unit cost can be obtained using the relationship $dC_{ui}/dB = 0$. Fig. 3.25 illustrates the minimum value of C_{ui} as a function of the ZoneGroup for the Barracuda drive using the bandwidth parameters for the drives derived in Section 3.4 for a stream bandwidth of 1.5 Mb/s. We observe that C_u increases as more zones are included in the ZoneGroup.

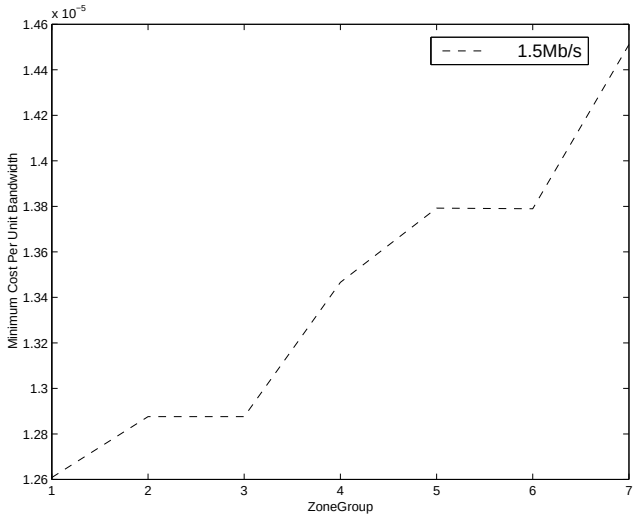


Figure 3.25: Minimum C_u vs. Φ

We see from Fig. 3.25 that the C_u profile for the Barracuda drive is clearly biased towards the choice of the first ZoneGroup as having the best C_u. This bias follows from the observation that the first ZoneGroup supports a higher number of concurrent sessions for a given disk capacity. However, the C_u must be clearly balanced against the capacity requirement. For a given capacity requirement, utilizing only the outer zones implies that a larger number of drives must be aggregated to meet the capacity requirements. This in turn increases the cost of interconnecting and maintaining the system. Defining this requirement requires a mapping of the disk interconnect cost to the ZoneGroup structure and is not considered in this dissertation.

3.6 Conclusions and Design Guidelines

In this chapter, we have examined several hardware and I/O constraints that affect the performance of a CM server. Several measurement studies were conducted on real disk systems to evaluate the feasibility of applying the models to real systems. In doing so, we described the shortcomings in existing disk streaming models and described modifications that yield more accurate predictions. In the second part of this chapter, we explored the possibility of trading capacity for bandwidth in ZBR disks by exploiting the non uniform transfer speeds of electro-mechanical disks. We subsequently proposed and demonstrated the use of a cost model to evaluate latency-bandwidth tradeoffs in a CM server. Some of the important conclusions from this chapter are enumerated below.

- Server configurations can trade disk capacity for bandwidth to design optimal configurations that minimize storage requirements. Such a policy translates to reduced system costs and a process to make video servers more economical to build.

- While it is beneficial to lay out data contiguously for a single transfer, for most high bandwidth applications, data can be dispersed in fixed size units without ill effect. This is an important observation as it frees us from the requirement of laying CM data contiguously.

- CM disk utilization is bounded by the size of the available disk buffer and the choice of the disk-to-host I/O protocol. The I/O subsystem limits the maximum achievable disk utilization and there are few gains from scheduling with very large transfer units. Furthermore, the constraints on session requirements preclude a 100% utilization of disk bandwidth.

- The choice of a fixed transfer unit helps us predict performance accurately. Since multirate sessions can be accommodated within this constraint, bandwidth can be utilized more efficiently.

The work in this chapter establishes a foundation on which one can build large storage hierarchies. This work is directly applicable to single or multiple disk systems (e.g., RAID). Scaling this work to include disk arrays is straightforward. For block interleaved (e.g., RAID-5) systems, the models described in this chapter can be applied exactly as described with little modification. For bit interleaved systems, the models are modified by replacing R_c with $\frac{R_c}{D}$ where D represents the number of disks in the array.

Chapter 4

Data Placement and Reorganization in Large Scale Video Servers

Synopsis

In this chapter we consider the characteristics of user access to stored content in a CM server and their effects on server operation. The notion of object-popularity is introduced as a factor in server design. We subsequently describe models for characterizing user access behavior derived from studies on WWW server and movie theater data. The effects of changing object-popularities on server operation are used to define criteria for minimizing session blocking probability for replicated servers. Finally, we consider the issue of data replacement in CM servers and propose resource allocation mechanisms for reducing reorganization times.

4.1 Introduction

The performance of a CM server is affected by the dynamics of user access behavior and the storage and bandwidth requirements of its media objects [24, 25, 53]. Within this context, a CM server can be visualized as a data repository containing a finite number of media objects. An object can be an audio clip, video, text or graphics component with a finite size. For an object without a notion of time (e.g., text and graphics), a reference bandwidth can be associated via the the maximum acceptable latency in displaying the object. For example, a static object of size B bytes that can be displayed with a latency $\mathcal{L}$ s would have a bandwidth $R = B/\mathcal{L}$ B/s. Table 4.1 introduces some of the symbols used in this chapter.

Let N represent the number of users accessing the system and K the number of stored objects at a given time. Parameter K can vary over time but is fixed during an operational interval Υ between server updates. We define a server update as an operation that results in data being altered in the server, either via deletion, addition, or reorganization. To determine the capacity and bandwidth requirements for a CM server, we must consider the relative popularities of the media objects in addition to their individual capacity and bandwidth requirements [53]. We can define a popularity vector P consisting of the normalized access demands for the K objects as

$$P = [p_1, p_2, ..., p_K] \quad \text{where} \quad \sum_{i=1}^{K} p_i = 1.$$

P is non-stationary and changes over time due to the changing popularities of the media objects. The current-popularity p_i of a given object i can be considerably different from its overall *historical* popularity. For example, in a movie server, a title such as *Casablanca* is a classic which is considered very popular in a historic sense;

Table 4.1: Server System Parameters

K	number of stored movies
N	user population
p_i	popularity of movie i (probability of movie access)
P	popularity vector $[p_1, p_2, ..., p_K]$
p_{max}	popularity of the most popular movie
Υ	interval between server updates
D	number of storage devices
O_i	number of copies of object i
l	maximum number of concurrent sessions per disk
$N_i(t)$	number of users accessing object i at time t
s_i	probability of access for disk i
S	probability vector for disk access $[s_1, s_2, ..., s_d]$
Q	probability of available sessions from a disk
Z	$K \times D$ population replication matrix
λ	request arrival rate at the CM system
P_B	probability that a user's call is blocked
P_{Bi}	probability that a call is blocked at disk i
μ	average call holding time
k	number of objects stored on a disk

however, on any given day, few customers might access it and its current-popularity will be small by our definition. In this dissertation, we use the term *popularity* to describe the current-popularity of an object.

The mapping of objects to popularities can change rapidly but typically, this mapping to popularities remains constant within a reasonable interval (e.g., 24 hours for a VOD database). This assumption is consistent with a recomputation of popularity during every server update. For example, consider a VOD database with $K = 10$. Let $[p_1, p_2, ..., p_{10}]$ be the popularities of the individual movies in descending order of popularity. A new movie that is most popular would map to p_1. As time passes, its popularity drops, until it is discarded and replaced by another movie.

The exact form of P is often unknown due to the unknown behavior of user accesses. Estimating the capacity and bandwidth requirements requires a priori knowledge about the nature of user accesses and the distribution of media units in addition to their bandwidth and storage requirements. When this information is not available, one must approximate the nature of user accesses to design the system.

The Zipf distribution [40]:

$$p_i = \frac{1}{i \sum_{j=1}^{K} \frac{1}{j}} \tag{4.1}$$

is most commonly used to characterize the skew in access demands among the set of available rental videos [1, 23]. The Zipf distribution represents a nonlinear access characteristic in which a small fraction of the objects are accessed the most often. The parameters of the Zipf distribution can be varied to modify the skew of accesses for modeling purposes [24].

However, the Zipf distribution is more useful for characterizing the long term

access behavior [40]. The instantaneous popularities of the different objects in the system are more general and cannot be approximated via a Zipf distribution. Furthermore, CM server models that use the Zipf distribution typically use access statistics originating from video store rentals or libraries where the user is limited to the physical copies of an available object. They are are not truly representative of the interactive nature of accesses one would expect in a CM server. We now describe models that can be used to more accurately characterize access patterns at a CM server based on data derived from the WWW and movie theater revenues [66].

4.1.1 The WWW Access Model

Several recent studies have attempted to characterize user access and data characteristics of content served on the WWW [7, 11]. These studies compute object size distributions and their relative popularities to facilitate strategies for efficient data retrieval using caching techniques. However, the studies are based on static data sets where content updates are infrequent. Another aspect ignored in these studies are the effects of WWW "surfing" where most visits to the WWW server are from users who do not return frequently. Unfortunately, little information is available regarding access characteristics at sites with dynamic information content (e.g., online newspapers).

Thus, accesses to a WWW server are not sufficient to capture the dynamics of access behavior in an interactive CM server. However, it is possible to consider accesses to a small set of relatively long documents with a finite user population and correlate its behavior to a CM server as is described in this section.

Characteristics that can be extrapolated from a WWW to a CM server are the system loads at different intervals in a day. Fig. 4.1 illustrates the access demand on

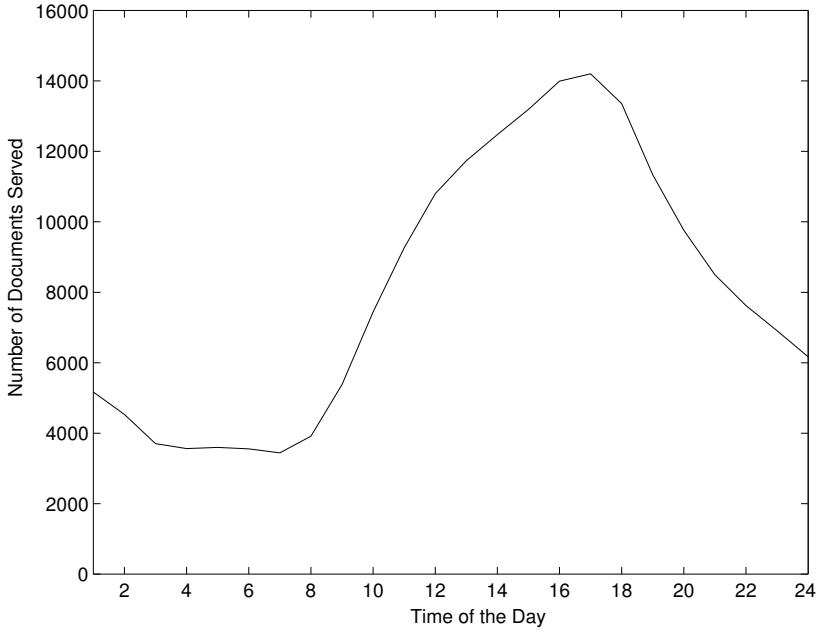


Figure 4.1: Server Loading vs. Time of Day

the WWW server at the Multimedia Communications Laboratory (MCL) at Boston University server for a 24 hour period, averaged over a year. It is apparent that the server experiences periods of moderate to low loading. Such a period, if available, can be used to schedule server updates and reorganization.

To evaluate the behavior of user accesses to dynamic content, we tracked accesses to the *Proceedings of the 5th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'95),* available online from the MCL server. The document hierarchy for the on-line proceedings consists of 95 documents of which approximately 40 are PostScript format. Access to this database is similar to that of a user accessing documents from an indexed library. However, the user is not restricted by the count of the physical number of copies of the desired document. This feature allows us to capture the characteristics of change in access popularity of a document with time.

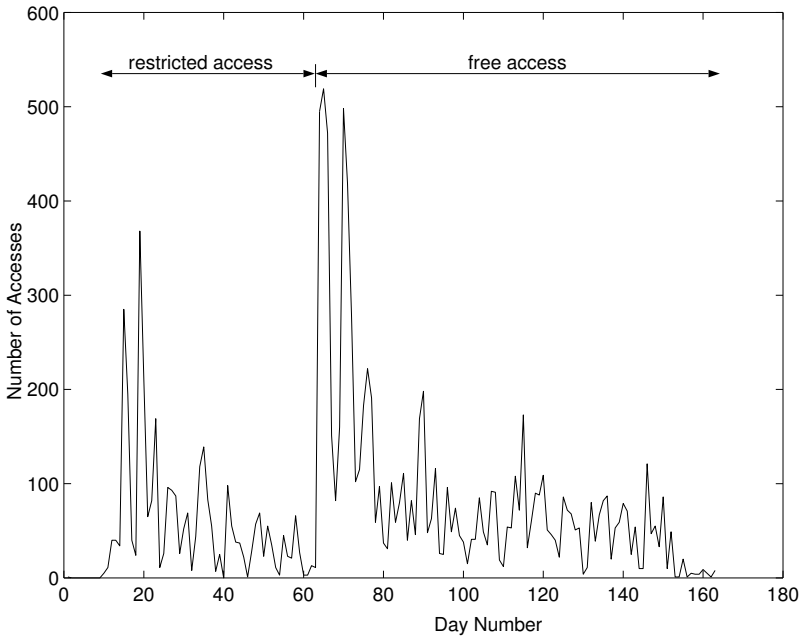


Figure 4.2: Daily Server Access Statistics

Fig. 4.2 illustrates the total accesses to the database for each day during our observational period. This curve illustrates two distinct regions. The first part represents an initial period of operation when access to the server was restricted to conference participants only. The second part represents the period when this restriction was removed and the proceedings were available to the general public. This availability was also publicized by announcing the proceedings to over 1,200 individuals. It is also clear from the figures that the there is a general decline in document popularity with time.

The relative popularities for all the documents in the server over the duration of the test period is illustrated in Fig. 4.3. Overall, a total of 90 documents were served. Fig. 4.3 illustrates a Zipf distribution fitted onto the popularity curve. We see that the Zipf distribution is a fairly accurate characterization of the cumulative accesses from the server. It was also observed that the most popular documents transferred

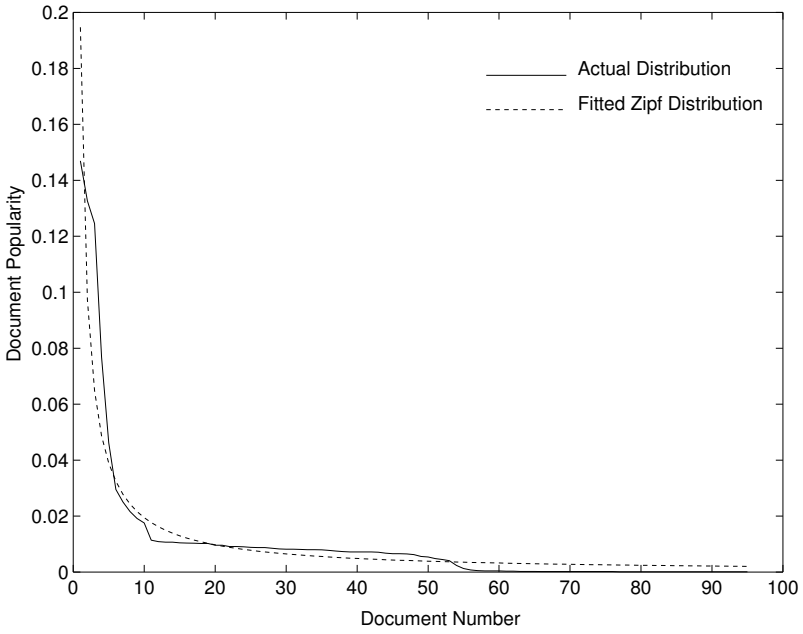


Figure 4.3: Popularity Distribution of All Documents

from the server were the index files that are smaller in size when compared to the actual papers.

The bulk of the data transfered from the server is due to the PostScript papers (complete documents). Fig. 4.4 illustrates the relative popularities of these documents over the entire test period. The access distribution for the papers is more uniformly distributed and does not match the Zipf distribution very closely. It is possible that eventually the access skew will map to the Zipf distribution. However, this characteristic is not immediately apparent by the short-term distributions.

Figs. 4.3 and 4.4 illustrate the cumulative popularities of the documents over the 180 day observation period. Fig. 4.5 illustrates the weekly sampling of the number of requests for the PostScript papers. It is clear from this figure that the access demands for the papers are dynamic in nature. While a general skew in the

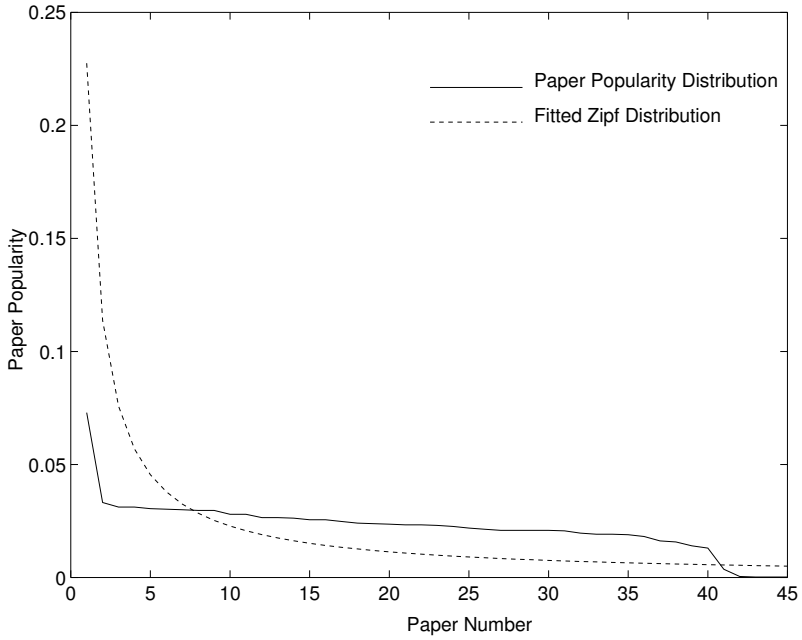


Figure 4.4: Popularity Distribution for Papers

relative popularity of the documents is observed, this skew varies with time.

It is apparent from these figures that an object's popularity declines with time. However, this behavior is not smooth and can change drastically, as was illustrated by the sudden increase in the number of requests after the availability of the documents was widely publicized. Such a scenario may occur in a CM database as a result of a favorable review or a publicity blitz. A CM server must be capable of absorbing such load fluctuations with little visible effect to the user.

Some features in this study are atypical for a CM server. For example, all documents became accessible to the users at the same time, which is similar to a CM system in which all objects are released at the at once. While the chances of 50 movies being released simultaneously are remote, it is often the case that 4 to 5 movies are released together. Even though all new movies are not equally popular, it

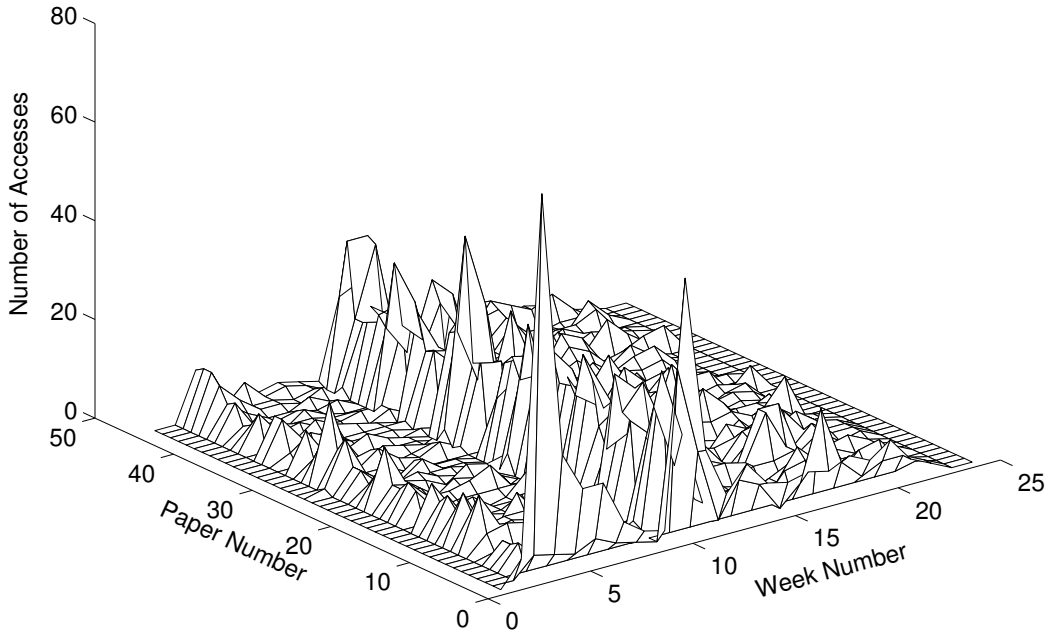


Figure 4.5: Weekly Access Demand for Papers

is possible that there is sufficient demand for all of them. We now address this issue by considering data from movie theater rentals.

4.1.2 Access Behavior in Movie Theaters

As a second source for modeling user access behavior we consider weekly revenues of movies released in North America. The data presented below is for the top 20 grossing movies and for a period of 23 weeks from May 5, 1995 to October 13, 1995, obtained from Entertainment Weekly Online [27]. Though the data are for the top 20 grossing movies, they are sufficient to capture the dynamics of user access behavior.

Fig. 4.6 illustrates the weekly grosses of the top 20 movies without considering individual movie behavior. It is clear that a small fraction of the movies are responsible for the maximum revenues. However, this distribution is non-stationary

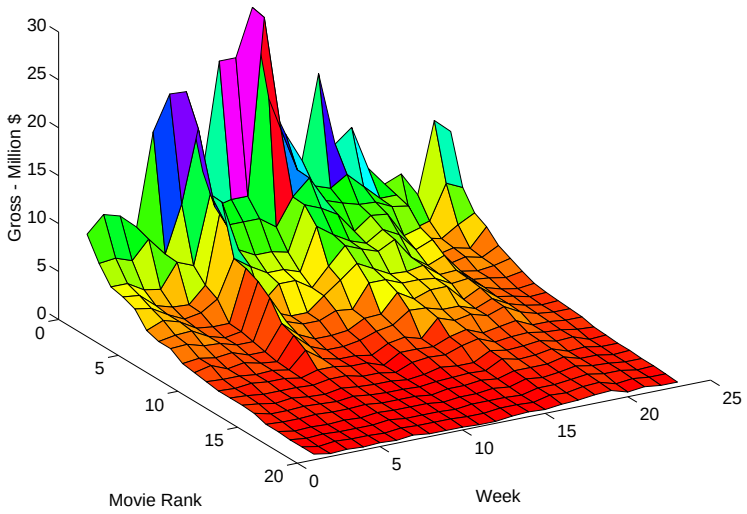


Figure 4.6: Weekly Grosses of top 20 movies

and changes from week to week.

Next, we consider the grosses of each movie as a function of its presence in the top 20 list. The resulting behavior is illustrated in Fig. 4.7. We observe that there is a decline in the revenue of a movie with time. However, the rate of decline is not constant and varies from movie to movie.

Finally, we consider the distribution of opening week revenues in Fig. 4.8. A vast majority of the movies (more than 60 %) have poor opening week revenues. This implies that the duration of their presence in the top 20 list is small as illustrated in Fig. 4.7. In other words, they are quickly replaced by new movies in the top 20 list.

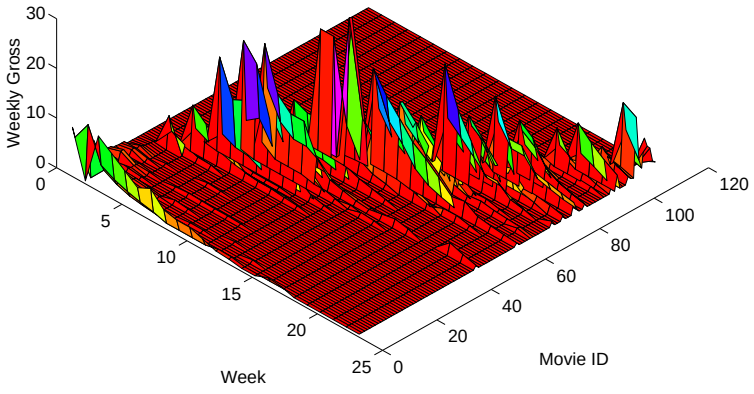


Figure 4.7: Weekly Gross Per Movie

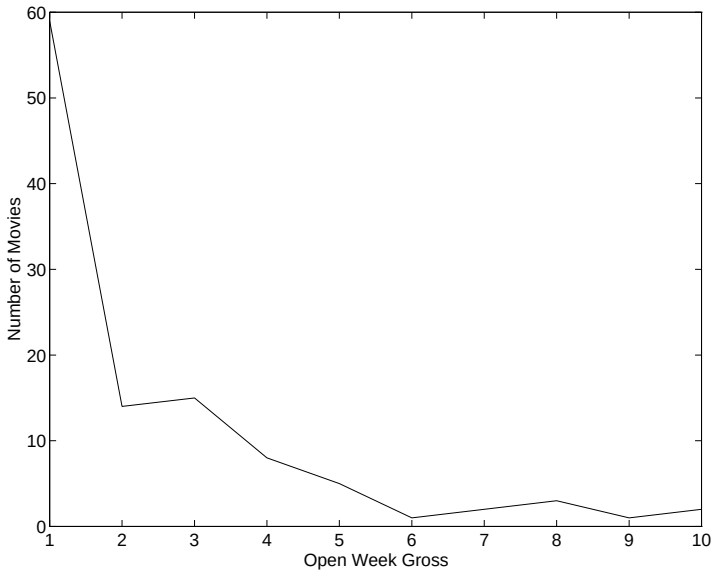


Figure 4.8: Distribution of Opening Week Revenues

4.1.3 Object Model for a CM Server

We now extrapolate observations from the access behavior in the WWW server and movie theaters to propose a model to capture the dynamic behavior of objects in a CM server. Our model uses three parameters to characterize the behavior of objects in the CM server:

1. The arrival rate of new movies into the system given by a rate parameter λ. λ is highly dependent on the context under which a CM server is evaluated. For example, in a movie server, arrivals are bursty as most movies are commonly released on weekends.

2. A weight parameter ω_i that determines the popularity of the object at the time of its manifestation in the server.

3. A rate parameter η that captures the change in an object's relative popularity a function of time.

Thus, each object i that manifests in the system is characterized by three parameters: an initial weight ω_i that represents its initial popularity, a rate parameter η_i that determines the rate of change of its popularity with time, and a parameter $\tau_i(t)$ that determines the instantaneous popularity (weight) of the object at a time t.

We have thus defined two parameters that can describe the behavior of an object i in a video database, (i) an initial parameter ω_i that can be a value generated from a distribution as illustrated in Fig. 4.8 and (ii) the relative weights or "liveness" that describes the relative weight of the object at a given point in time. Clearly, $\tau_i(t) = f(\omega_i, \eta_i)$. If we assume that the resulting change in popularity is exponential the two are related by

$$\tau_i(t) = \omega_i e^{-\eta t}. \tag{4.2}$$

Alternatively, η can be replaced by an observation-based scheme for computing the changes in object popularity. For example, we can use a linear predictor that considers the object popularity over two sampling intervals to compute the expected number of accesses to the server for the next interval. Let $N_i(t-2)$ and $N_i(t-1)$ be the number of customers accessing object i on intervals $(t-2)$ and $(t-1)$ respectively. Using a linear predictor, the number of customers expected to request i for interval t is computed as

$$N_i(t) = 2N_i(t-1) - N_i(t-2), \quad N_i(t) \geq 0. \tag{4.3}$$

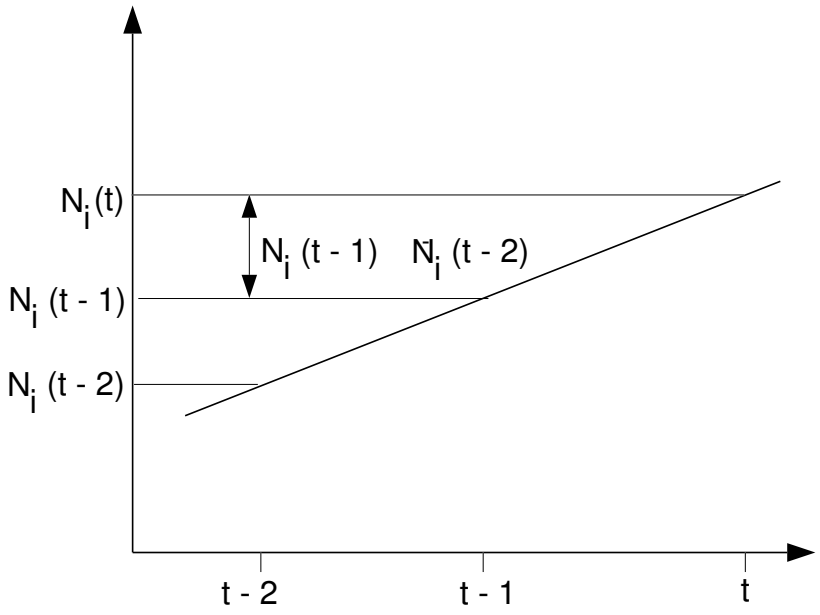


Figure 4.9: A Linear Predictor

The resulting $\tau_i(t)$ can be computed using the relationship $\tau_i(t) = N_i(t)$. For the linear predictor, the number of customers requesting the object is a linear function of the number of customers who accessed it in the previous two intervals. The slope is determined by the change in the number of users over the previous two days. The linear predictor is illustrated in the Fig 4.9. The popularity of a given object i for a given interval t is then computed as

$$p_i = \frac{N_i(t)}{\sum_{m=1}^{K} N_m(t)}. \tag{4.4}$$

The resulting popularity popularity vector for the movie database at a given instant t is now described by the function

$$P(t) = \left[\frac{\tau_1}{\delta}, \frac{\tau_2}{\delta}, ...\frac{\tau_K}{\delta}\right] \qquad \delta = \sum_{i=1}^{K} \tau_i. \tag{4.5}$$

$\tau_i(t)$ can be replaced by a more sophisticated function if necessary. The linear model is more flexible than the exponential one as it can handle both increases and decreases in object popularity. Since the predictor is only used to track object popularities, it can be easily replaced by more complex nonlinear predictors if necessary. In this dissertation, our focus is on the effects of changing user requirements on server operation and we limit ourselves to the simple models described in this section. In Chapter 5 we describe simulation studies that consider the effects of dynamic user accesses on the storage architectures of a CM server. We now consider the design of a large storage hierarchy.

4.2 Replication Policy for CM Storage

In this section, we consider system requirements for meeting a given access load using content replication across multiple servers. Replication is useful in scenarios where capacity and bandwidth constraints preclude the placing of all content on a single widely striped system.

If there are N customers accessing the system and N is large, the number of requests for a given object i can be approximated as $N * p_i$. Let

$$o_j = (x_j, y_j) \quad j \in K$$

represent the bandwidth and storage requirements of a media object j. The system bandwidth necessary to support this object is then given by

$$R_j = Np_jx_j.$$

The cumulative bandwidth requirements that the system must support can be then estimated as

$$R = \sum_{j=1}^{K} R_j. \tag{4.6}$$

For media systems with homogeneous bandwidth requirements, x_j can be replaced by a constant parameter R_c. Establishing a similar equation for the capacity requirements is much harder. Since each disk[1] has fixed I/O and storage capacity, an object can be replicated across several devices to meet its expected load. For replication, the capacity requirements for object i equals O_iy_i, where O_i is the number

[1]We use the term disk to represent any storage device including clusters of disks.

replicas of object i. Evaluating the capacity constraint is simpler for wide-striped as data is stored across all devices in the system. Consequently the storage requirement for object i is only y_i and the required capacity is given by

$$C = \sum_{j=1}^{K} y_j. \tag{4.7}$$

However, system I/O constraints and performance requirements (startup latencies) limit the number of devices that can be included in a single stripe and content may need to be replicated. In other words, the K objects are replicated to distribute object access demand to sufficient I/O bandwidth depending on their current probability of access and are stored on d disks. By restricting the number of objects K to be constant over a popularity recomputation interval (e.g., 24 hours for a movie database), objects are redistributed at the beginning of each interval. Thus, there is a tradeoff in the capacity requirement depending upon the choice of the server architecture.

4.2.1 Capacity and Bandwidth Estimation

We envision the CM server as an array of D disks (or disk clusters), each disk j with a finite capacity. Due to limited I/O capacity, each disk can only support a finite number of sessions, l, thereby limiting the peak number of active CM users to l. When device I/O bandwidth constraints are considered (i.e., limited sessions per device), user access demands are satisfied by object replication across the set of disks comprising the CM server.

If O_i represents the total number of copies of an object i, the probability of accessing a given object copy can be approximated as

$$p_{ic} = \frac{p_i}{O_i} \tag{4.8}$$

if each object copy is equiprobably accessible. However, the actual usage of a object copy is dependent on dynamic load patterns. We now establish a basic criterion.

Lemma 2 *There is no improvement in connection setup probability when more than one copy of the same object is placed on a video disk.*

Justification: Since we assume that user requests are independent, separation in object start times yields disk head movement as would occur for replicated object copies on the same device. Therefore, as demonstrated in Chapter 3, there is no throughput gain by placing more than one copy of the same object on the same disk. There may be marginal improvement due to reduced latency, but this is not predictable due to the variation in accesses from user to user. A priori knowledge of user interaction would be necessary during physical disk layout to achieve these gains.

As each disk supports a finite number of sessions, l, the lower limit on O_i is

$$O_i \geq \frac{p_i \times N}{l}. \tag{4.9}$$

In Eq. 4.9, we have implicitly assumed that the capacity of any object does not exceed that of the disk. This assumption is consistent with the characteristics of current storage architectures for a wide range of bandwidth requirements. Thus, with O_i clusters, only the bandwidth requirements of object i can be satisfied. The minimum number of disks required to support the ith object is given by a simple modification of Eq. 4.9, or

$$D_i \leq \left\lceil \frac{N \times p_i}{l} \right\rceil . \tag{4.10}$$

This inequality requires, in the limiting case, placing a single object copy on each single disk to meet the access demand. If additional content is placed on the clusters containing copies of O_i, the number of replicas of O_i may need to be increased to meet the bandwidth requirements. We can also define the leftover bandwidth on the disks containing object i as

$$L_i = D_i l - \lceil N \times p_i \rceil. \tag{4.11}$$

Clearly, any leftover bandwidth can be used to meet the streaming requirements of additional sessions.

Lemma 2 places a lower bound on the number of disks required in the CM system. If p_{max} is the probability of access for the most popular object, the lower bound on the number of disks required in the system is

$$D_{min} \geq \left\lceil \frac{N \times p_{max}}{l} \right\rceil . \tag{4.12}$$

Violation of this bound by object replication on the same disk leads to storage inefficiencies without any gains in the disk I/O bandwidth. We therefore define the bounds for the number of disks required for the system as

$$\left\lceil \frac{N \times p_{max}}{l} \right\rceil \leq d \leq \sum_{i=1}^{K} \left\lceil \frac{N \times p_i}{l} \right\rceil . \tag{4.13}$$

A single system can meet the requirements of all the sessions when $D_{min} = 1$ and L_{min} is sufficient to meet the requirements for all the sessions. In Eq. 4.13, the

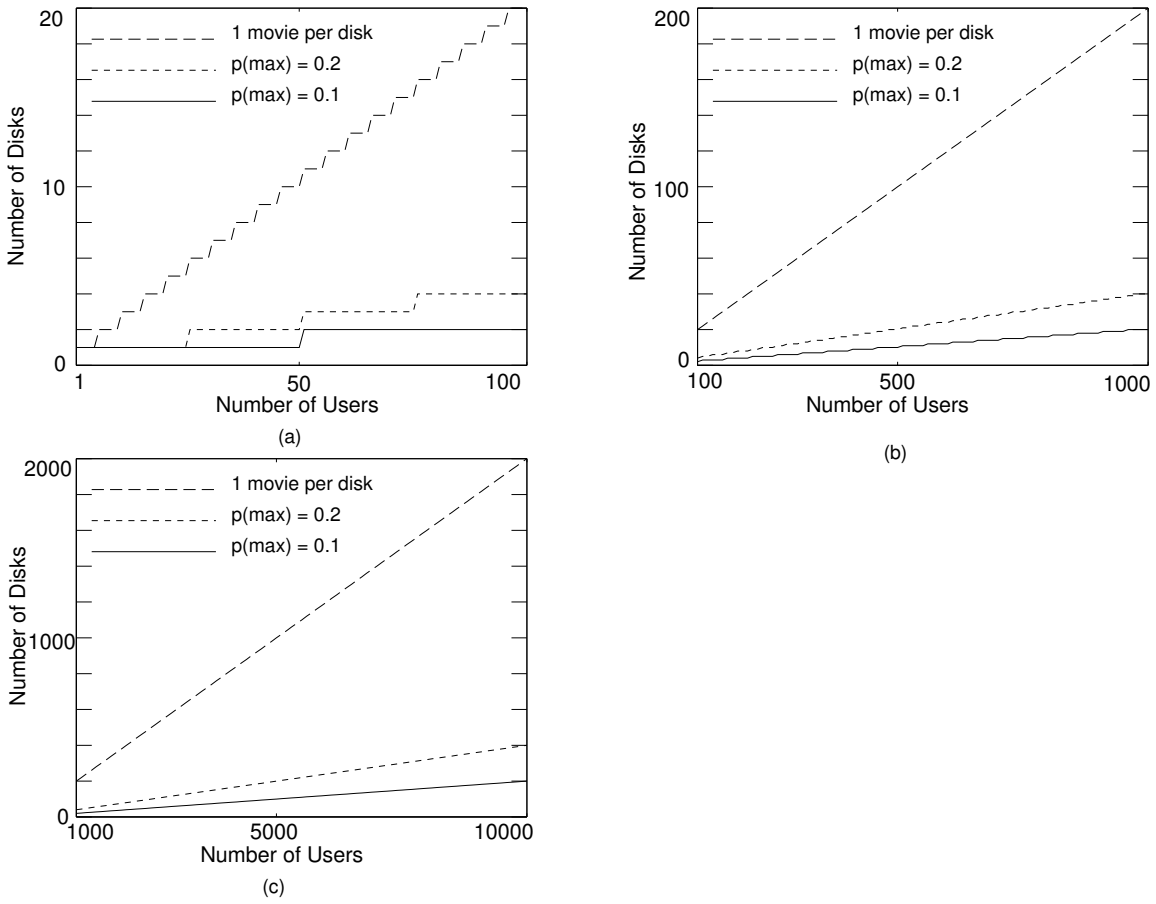


Figure 4.10: Object-Disk Assignment Bounds for a CM Server

lower bound is obtained from the constraints imposed by the most popular object, and the upper bound is derived from the extreme case when each disk holds only a single object. These bounds are illustrated in Fig. 4.10 for $l = 5$, and various values of N. Fig. 4.10(a) demonstrates the bounds for a small number of sessions, while Fig. 4.10(b) and (c) illustrate the bounds for a large number of sessions.

The bounds described in Eq. 4.13 assume that bandwidth is the primary bottleneck in establishing server requirements. The main constraints in determining the correct value of D are dependent on the characteristics of the media objects and the distribution of user requests. When both capacity and bandwidth constraints are

available, schemes based on bandwidth space ratios (BSR) can be applied to evaluate the number of storage devices to meet the object placement criteria [24]. However, given a set of devices, it becomes important to find the best replication technique to maximize the chances of a user successfully retrieve content from the server. We now establish the necessary criteria for minimizing session blocking probability in a CM server.

We define a popularity-replication matrix Z as a $K \times D$ element matrix such that

$$\sum_{i=1}^{K} z_{i,j} = s_j$$

$$\sum_{j=1}^{D} z_{i,j} = p_i$$

and

$$\sum_{i=1}^{K} \sum_{j=1}^{D} z_{i,j} = 1.$$

Here, s_j is the popularity of disk j being accessed by any user. Row i of Z corresponds to the product of R_i with p_{ic}. The probability of a successful connection Q from a disk is defined as

$$Q = P\,(\text{ less than } l \text{ connections on the disk})\,.$$

To ensure that a maximum number of sessions are supported, Q must be maximized. We have to allocate objects to disks such that the chance of a successful session is maximum.

Lemma 3 *The probability that a customer's request results in a successful connection is maximum when the objects are distributed such that each disk has a uniform probability of being accessed (i.e., $s_p \approx s_q$ for all p and q).*

Proof: Lemma 3 is established by the application of a queueing model and assuming exponential interarrival and holding times. Let us assume a Poisson arrival process with a rate λ. We want to minimize the probability of call blocking. Given the popularity vector for the disk array $[s_1, s_2, ..., s_d]$, and assuming random requests, the arrival rate at any disk i, λ_i is given by

$$\lambda_i = s_i \lambda.$$

If we assume that the viewing (call holding) times are exponentially distributed with a parameter μ, then the probability of blocking at server i, P_{Bi} is given by Erlang's B Formula [37],

$$P_{Bi} = \frac{(\frac{s_i\lambda}{\mu})^l / l!}{\sum_{j=0}^{l} (\frac{s_i\lambda}{\mu})^j / j!}. \tag{4.14}$$

The probability that a random request for a object is blocked is

$$P_B = \sum_{i=1}^{d} s_i P_{Bi}.$$

Substituting Equation 4.14 for P_{Bi}, we obtain,

$$P_B = \sum_{i=1}^{d} s_i \frac{(\frac{s_i\lambda}{\mu})^l / l!}{\sum_{j=0}^{l} (\frac{s_i\lambda}{\mu})^j / j!} \quad or$$

$$P_B = (\frac{\lambda}{\mu})^l / l! \sum_{i=1}^{d} \frac{s_i^{l+1}}{\sum_{j=0}^{l} (\frac{\lambda s_i}{\mu})^j / j!} \tag{4.15}$$

The first part of Equation 4.15 is constant for a given λ. Thus, P_B is a minimum when the summation term is minimized. In Appendix A5, it is shown by the method of Lagrangian multipliers [22] that

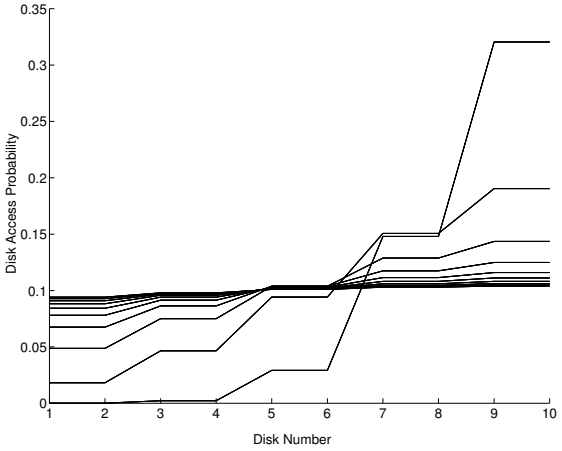
$$min(P_B) \Rightarrow s_i = 1/d, \quad \forall i. \tag{4.16}$$

Therefore, when disk accesses are equiprobable, the probability of request blocking is minimized. The following simulation study demonstrates the effects of miss ordering during object to disk assignment that can potentially degrade the optimal connection setup probability.

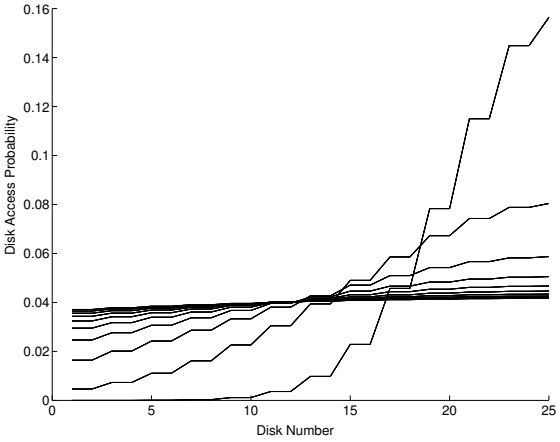
4.2.2 Simulation Study of Session Blocking Probability

To address the sensitivity of the replication scheme to sub-optimal object assignment we performed several simulations for different disk counts and popularity distributions. For each disk count, the initial popularity distributions differed significantly from the optimum, and gradually approached the uniform distribution. This allowed us to determine the tolerance of the system to sub-optimal orderings. Simulations were run for systems with 10, 25, 50, and 100 disks. Each disk was capable of supporting a maximum of 5 concurrent sessions with an average viewing time of 90 minutes. Figs. 4.11–4.13 illustrate the results from our simulations.

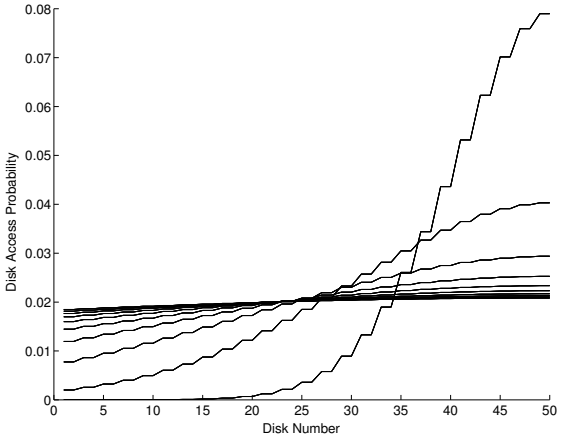
The distribution of arrival requests at the disks are illustrated in Fig. 4.11. The resulting probability of blocking for both the simulation study and the analytic computation are illustrated in Figs. 4.13 and 4.12 respectively. It is clear from the graphs that the theory is accurate in predicting the behaviour of the system, thus validating our conclusions.



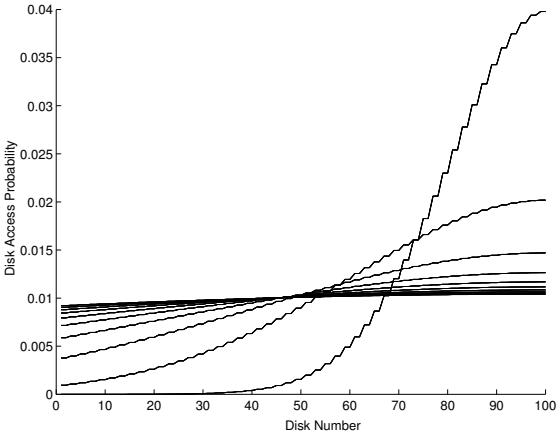
10 Disk System



25 Disk System

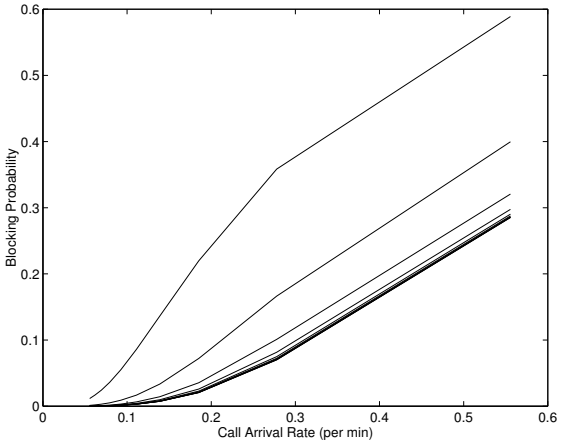


50 Disk System

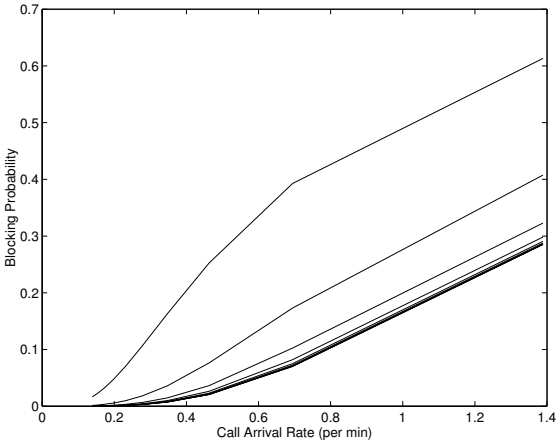


100 Disk System

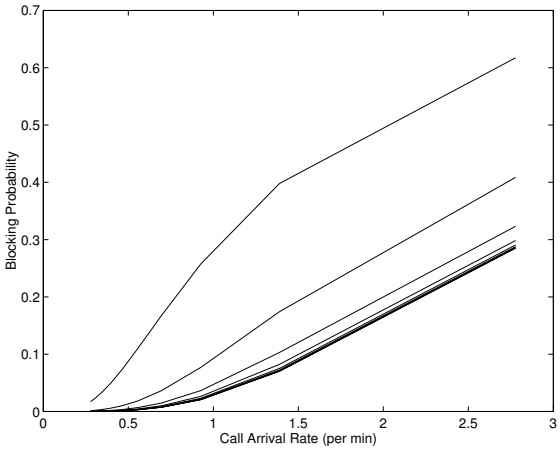
Figure 4.11: Request Arrival Distributions



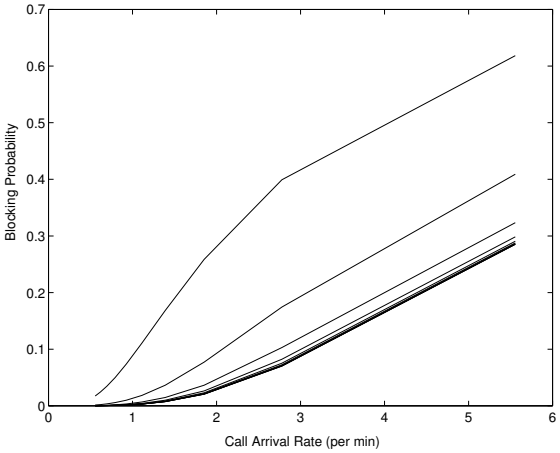
10 Disk System



25 Disk System

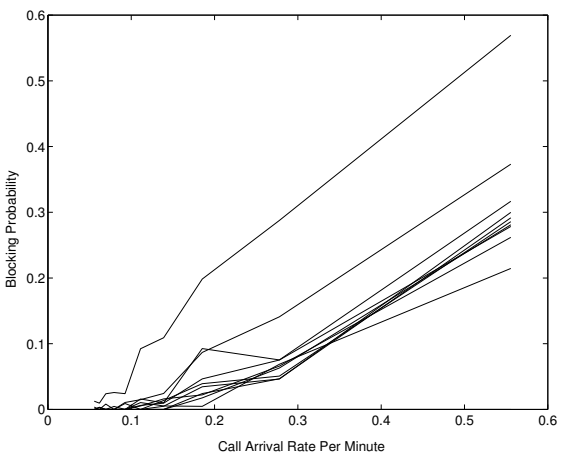


50 Disk System

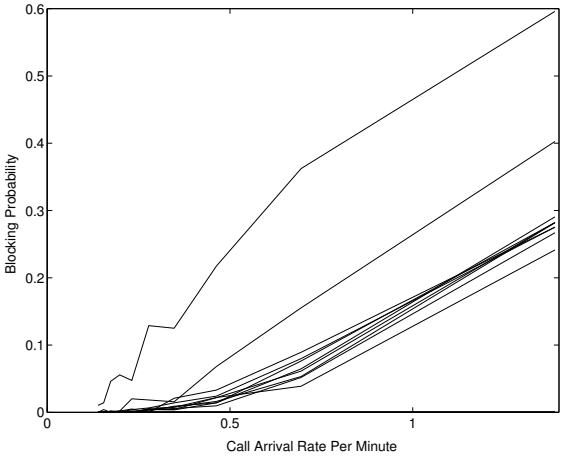


100 Disk System

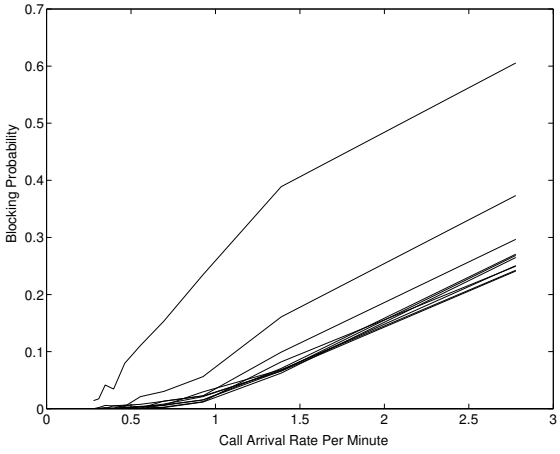
Figure 4.12: Call Blocking Probability (Analysis)



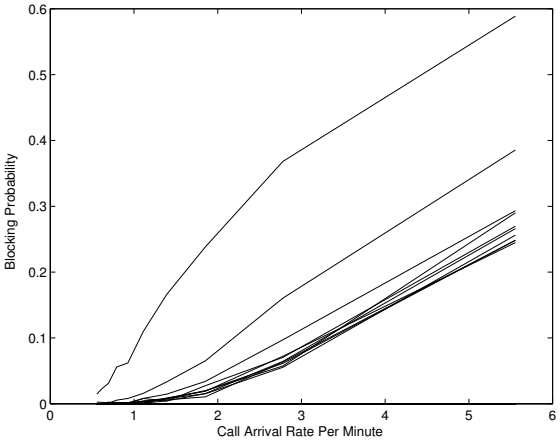
10 Disk System



25 Disk System



50 Disk System



100 Disk System

Figure 4.13: Call Blocking Probablilty (Simulation)

It is apparent from the figures that any system which deviates significantly from the balanced system performs poorly. The results indicate that the equiprobable assignment of movies to disks always yields the highest session availability. Furthermore, even moderately "flat" assignments within a reasonable tolerance (10–20 %) yield significant gains and are practically indistinguishable from the optimal.

In reality, the problem of allocating movies to disks to balance probabilities is analogous to the "bin packing" problem that is known to be NP hard. However, from our simulations, it is clear that moderately balanced system can still yield significant performance gains as measured in the number of sessions supported. This suggests that simple allocation schemes can yield satisfactory performance. For example, it has been demonstrated elsewhere that the BSR scheme achieves load skews of about 4–5% which are within the tolerance limits for the requirements described in this section [24].

Until now, we have primarily focused our discussion on the placement of data in a CM server for efficient retrieval. However our interest is in modeling dynamic server operation where the server content changes on the fly. We now consider the issues of data reorganization in CM servers and policies for governing their operation.

4.3 Server Reorganization

The reorganization of content in a CM server is affected by many factors including existing sessions, bandwidth availability and storage requirements for the new data. Ideally, the new data must become available with the least delay and a minimal disruption of any existing services. This problem has been addressed from the perspective of a text system by Alonso et al. [3]. However, for CM data, the problem

of timing and object size play an important role in the efficiency of reorganization. In this section, we examine schemes for CM server reorganization in great detail and study their performance.

We identify two approaches to video server reorganization. In the *blocking* approach, the disk is made unavailable during reorganization. In the *non-blocking* approach, the disk continues to be active and supports sessions while being reorganized. It is preferable to support non-blocking schemes as content continues to be available even during reorganization.

Non-blocking approaches to reorganization can be classified into *staging* and *dynamic write* schemes. The staging approach consists of redundant disks for reorganization purposes. The reorganization process consists of two steps, (i) duplicating the data to be replaced onto the redundant disks and (ii) rewriting the data onto the original disks. Thus, data are available for retrieval even during reorganization. However, the system must incur the added cost of an additional set of disk clusters. In the dynamic approach, data are written to the disk as sessions are in progress using any spare bandwidth.

A second classification of reorganization techniques is *preemptive* and *non-preemptive*. In any preemptive technique, existing sessions are interrupted for reorganization. Clearly this is undesirable and as a consequence, practical implementations for reorganization must be non-preemptive. Consequently, data on a disk cannot be replaced if a session is already retrieving the data. In other words, the session must be allowed to run to completion before data are replaced. Alternatively, the session can be transferred to an idle disk subject to bandwidth and content availability. As a result, the number of users accessing the system at the time of reorganization directly impacts its duration. When the number of active users is large, less bandwidth

is available for reorganization.

In this section, we focus on dynamic, *non-blocking non-preemptive* approaches to server reorganization and the resulting performance tradeoffs.

4.3.1 Mechanics of Server Reorganization

To understand the issues involved in reorganizing data in a video server, consider the server architecture illustrated in Fig. 4.14.

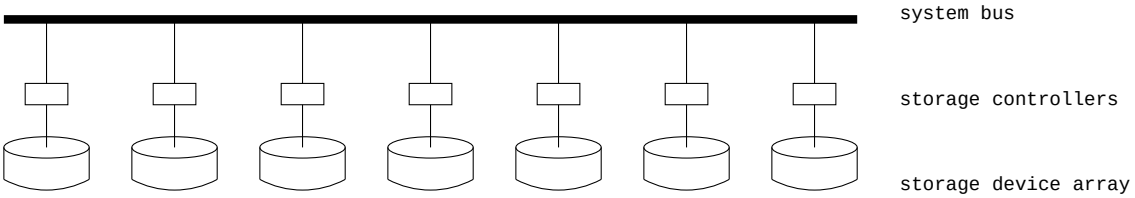


Figure 4.14: The Storage Server Architecture in a CM System

Several disks are connected to a high speed system bus used to transfer data to the delivery system. To make efficient use of bus bandwidth, data from the disks are first transferred to a buffer (controller) that stores and forwards the data to the communication device at a high speed. When data are to be written to the disks, the process is reversed. Any reorganization procedure is subject to the limitations due to the bus bandwidth, storage availability, and existing sessions. We list these criteria below:

- The size of the media object being replaced.
- The rate at which data can be written onto the disk.
- The availability of the system bus for transferring data to the disk.

- The number of users accessing the system during reorganization.
- The number and type of existing sessions on the disk.

The size of the media object being transferred o_i and the disk write speed R_{di} directly affect the speed at which the disk can be updated. In other words, the reorganization time is proportional to $\frac{o_i}{R_{di}}$. Table 4.2 lists the symbols used in this section.

Table 4.2: Reorganization Parameters

Symbol	**Parameter**
o_i	data to be reorganized on disk i
R_{di}	disk bandwidth available for reorganization
T_{wi}	time a disk must wait before overwriting o_{wi}
o_{wi}	data that can be written only after T_{wi}
R_{ni}	normalized write bandwidth
Υ_i	reorganization time for disk i
R_B	system bandwidth required for reorganization
T_n	normalization overhead

We now consider the mechanics of reorganization on the storage server. The first process in server reorganization is determining the system bandwidth necessary for reorganizing content. Let $\{o_i, o_{wi}, T_{wi}, R_{di}\}$ represent the reorganization requirements at disk i. o_i represents the total data data being written to disk i, T_{wi} the waiting interval before some data can be overwritten, o_{wi} the amount of data the system can write to after T_{wi} and R_{di} the disk bandwidth available for reorganization. A first step in reorganization is to estimate the amount of system bandwidth

R_B required for reorganization. Estimating this bandwidth requires computing the sum of the disk bandwidths over time.

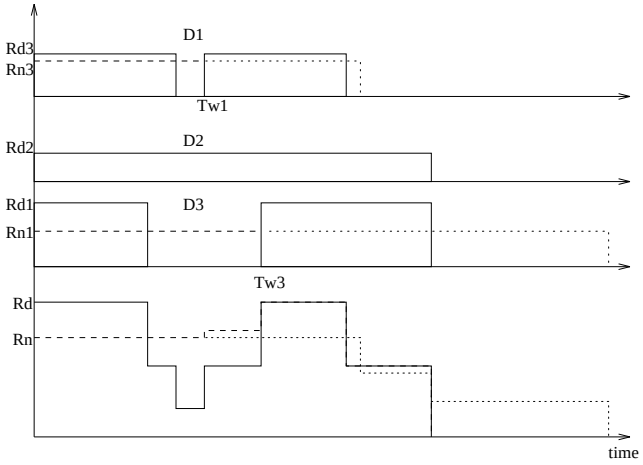


Figure 4.15: Reorganization Bandwidth Requirements

Fig 4.15 illustrates the reorganization requirements for a three disk system. $D1$ – $D3$ illustrate the reorganization processes at the three disks. In Fig. 4.15, $T_{w2} = 0$ and R_d (solid line) represents the cumulative bandwidth requirements of the three drives. The system cannot write $(o_i - o_{iw})$ bytes to the system before T_{wi}. Consequently, the reorganization time for disk i can be estimated as

$$\Upsilon_i = \begin{cases} \frac{o_{wi}}{R_{di}} + T_{wi} & : \quad o_i \leq o_{wi} + T_{wi} R_{di} \\ \frac{o_i}{R_{di}} & : \quad o_i > o_{wi} + T_{wi} R_{di} \end{cases} \tag{4.17}$$

The total time to reorganize the system is determined by $max(\Upsilon_i)$. The maximum system bandwidth necessary for reorganization can be computed as the sum of all the individual disk bandwidths or

$$R_B = \sum_{i=1}^{D} R_{di}. \tag{4.18}$$

Clearly, the proposed approach to reorganization requires an uneven allocation of system bandwidth. For some applications it may be desirable to allocate a fixed bandwidth to the disk for reorganization so the system has an accurate estimate of the bandwidth available for streaming sessions. In this scenario, we can apply the smoothing approach used to reduce the variance in the bandwidth requirements in VBR encoded video streams described in Chapter 2. Consequently, we define the *normalized* write bandwidth of a drive as

$$R_{ni} = \begin{cases} R_{di} & : \quad o_i = o_{wi}, T_{wi} > 0 \\ \frac{o_i - o_{wi}}{T_{wi}} & : \quad o_i \leq o_{wi} + T_{wi} R_{di} \\ R_{di} & : \quad o_i > o_{wi} + T_{wi} R_{di} \end{cases} \tag{4.19}$$

Allocation of the *normalized* bandwidth to the drive results in data being written continuously through the waiting interval, if possible. In Fig. 4.15, the effects of smoothing the write bandwidths are illustrated via the dotted lines. Normalization extends the reorganization times in the server and is undesirable when the speed of reorganization is more critical. We can compute this increase as:

$$T_n = \frac{o_{wi}}{R_{ni}} - \frac{o_{wi}}{R_{di}}$$

Depending on the system's requirements, R_{ni} can be increased to reduce the system reorganization times. However, this gain is constrained by the requirement that $R_{ni} \leq R_{di}$.

4.4 Summary

In this chapter, we considered some of the macro issues in the design of a CM server. Our study considered models for evaluating the characteristics of user accesses in large scale CM servers. We demonstrated through studies on WWW server and movie theater accesses data that accesses to a CM server are dynamic. This approach was used to develop a methodology for characterizing the relative popularities of documents in a CM server whose content changes over time.

Subsequently, we considered the I/O and storage constraints in server design due to the uneven distributions of access probabilities in the context of CM systems with identical session bandwidth requirements. Consequently, we considered load-balancing criteria in multi-disk CM servers and derived the requirements for minimizing the session blocking probability. Our model demonstrates that a balanced system yields the least chance of blocking. To evaluate the effect of sub-optimal orderings on server performance, we evaluated call blocking probabilities on unbalanced systems via simulation and analysis. The results demonstrate that even moderately balanced systems with load skews of 5–10% yield configurations with satisfactory performance.

Finally, we considered the bandwidth requirements for reorganizing content on a multi-disk CM server. We derived the system bandwidth requirement for server reorganization based on bandwidth and space criteria. Our analysis also considered scenarios in which the system must wait for a finite time before some of the content can be reorganized. We finally introduced the notion of smoothing the write bandwidth of the drive to minimize fluctuations in the system bandwidth. The effects of smoothing on reorganization times were derived.

Collectively, the work in this chapter can be applied as input to a given server configuration to evaluate the tradeoffs in server operation. In the next chapter, we describe the application of these techniques in the context of a multi-disk server to evaluate policies for governing server operation.

Chapter 5

Application in Server Design

Synopsis

In this chapter, we consider practical applications of the storage models and performance techniques described in earlier chapters. Consequently, we describe the derivation of an object size distribution for real movies using data derived from the Internet Movie Database. This distribution is used as input to a system model to evaluate storage requirements and evaluate performance bounds in a video-on-demand storage architecture.

5.1 Introduction

In the earlier chapters we considered two different aspects of server design. In Chapter 3 we considered micro, disk-level, issues in server design that can be used to determine the ideal server configuration for a given application bandwidth. In Chapter 4, we considered macro level issues in server design. Our discussion focused on models for characterizing user accesses of a CM server and the resulting criteria for load-balancing and content reorganization.

In this chapter, we consider the application of these techniques to evaluate storage and bandwidth parameters in a CM server. We demonstrate the application of the proposed user access model to determine storage requirements for supporting a given user configuration and an object size distribution.

5.2 Video Object Size Distribution

In order to characterize the distribution of object sizes the model of our CM server, we use data from the Internet Movie Database (IMDB). The IMDB contains information on most movies released in the past century and part of the description includes information on movie running times. For example, Fig. 5.1 illustrates a histogram of movie running times for 5,000 movies released in North America and pruned to include only movies that are longer than 20 minutes and shorter than 150 minutes.

We assume for convenience that the size of an encoded movie object is directly proportional to its running time. In other words, we assume that the digitization process encodes the movie at a constant rate. This is not true in general and especially so for VBR encoding. However, as described in Chapter 2, a smoothing buffer can be

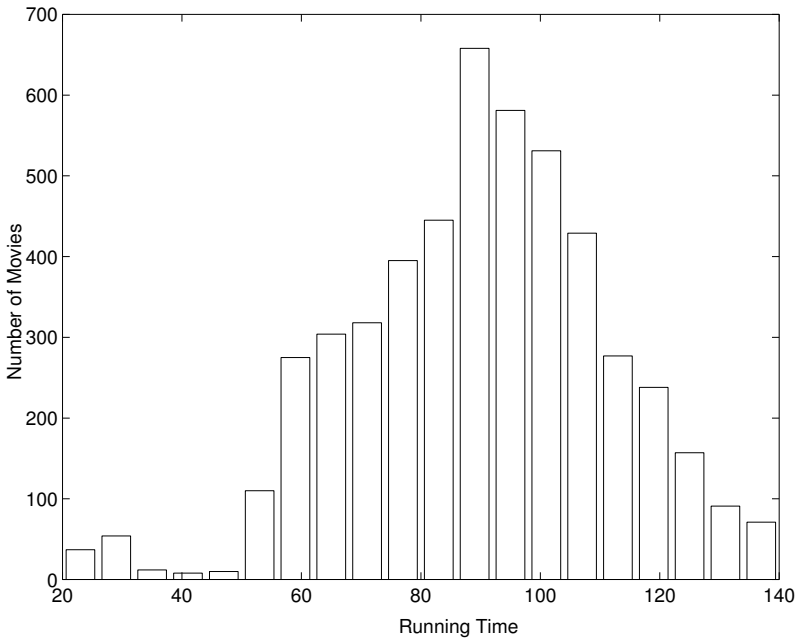


Figure 5.1: A Histogram of Movie Run Times

used to achieve an equivalent result with a uniform display rate. Furthermore, most practical MPEG encoders produce video at a constant rate. Thus, we can directly correlate the size of a movie object S to its rate of encoding R and its duration T by the relation

$$S = R * T. \tag{5.1}$$

Using this assumption, we proceed to describe the generation of the movie size distribution. The CDF (cumulative density function) of the movies in the IMDB list is illustrated in Fig. 5.2. This data includes a sample of 5,324 movies, with the longest one being 480 minutes and the smallest 11 minutes. The list is pruned to only consider movies released in North America and to eliminate duplicate entries.

The resulting cumulative distribution (called the Movie Object-Size Distribu-

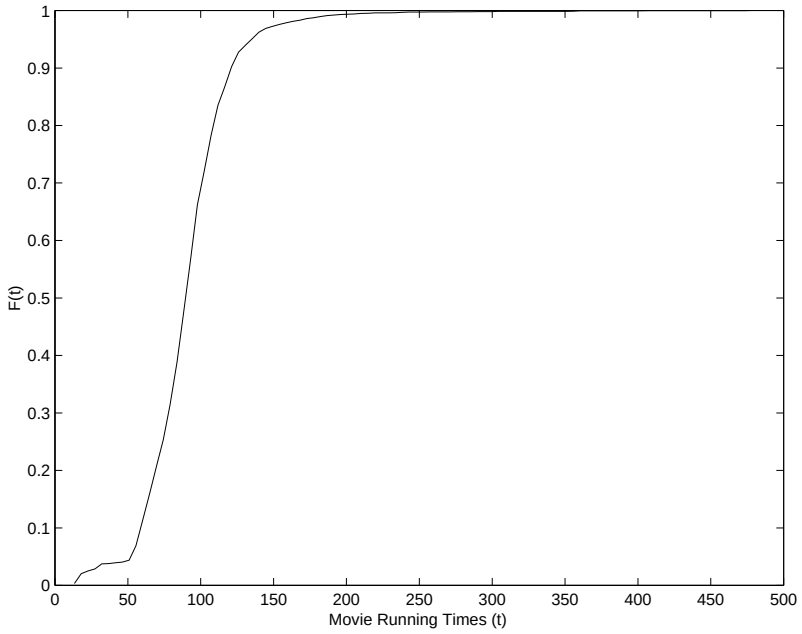


Figure 5.2: CDF of Movie Run Times

tion or MSD) $\mathcal{F}(x)$ is approximated by a combination of seven linear equations that are valid for a range of run times and is given by

$$x = \begin{cases} 0.0023a - 0.0276 & : \quad 10.0000 \leq a \leq 22.7250 \\ 0.0007a + 0.0102 & : \quad 22.7250 < a \leq 50.8650 \\ 0.0096a - 0.4451 & : \quad 50.8650 < a \leq 79.0050 \\ 0.0139a - 0.7865 & : \quad 79.0050 < a \leq 121.2150 \\ 0.0025a + 0.5997 & : \quad 121.2150 < a \leq 149.3550 \\ 0.0004a + 0.9068 & : \quad 149.3550 < a \leq 196.2550 \\ 0.00002a + 0.9885 & : \quad 196.2550 < a \leq 477.6550 \end{cases} \tag{5.2}$$

In Eq. 5.2, a represents the size of the movie in minutes of play time. Using MSD, it becomes feasible to generate a realistic distribution of movie objects sizes. The behavior of the MSD is illustrated by the dotted lines in Fig. 5.3.

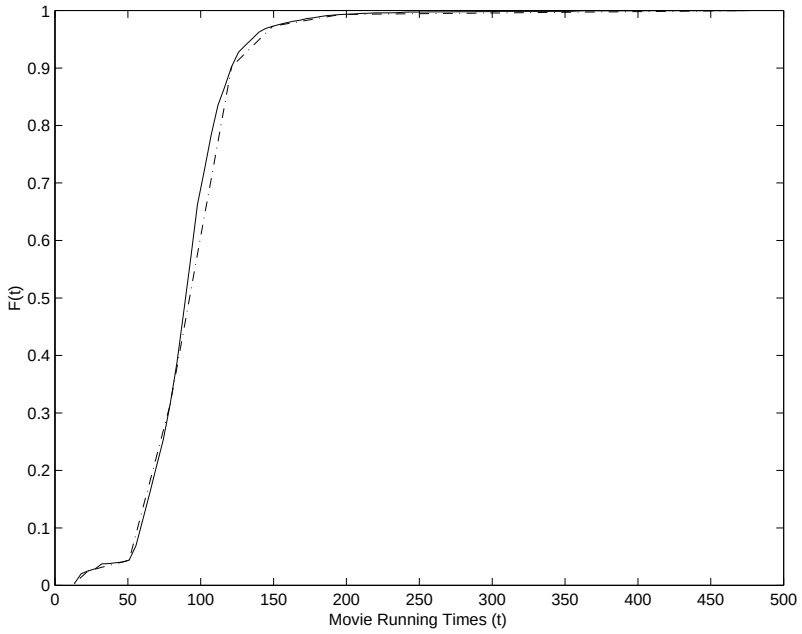


Figure 5.3: Analytical Model of Movie Run Times

In the next section we apply the MSD as an input to the system simulation model to study storage requirements in a CM server.

5.3 Simulation of Long-Term Access Behavior and Storage Requirements

To characterize long term access behavior and estimate the corresponding storage requirements in a storage server, we simulated access behavior in a server with the characteristics described in Table 5.1.

We assume that each week, several movies are released simultaneously and that this number is uniformly distributed between 1 and 20. Each new movie is assigned an initial weight based on a static Zipf distribution of length 60. The popularity

Table 5.1: Long Term Simulation Parameters

Stream rate	1.5 Mb/s
Average users in the system	1000
Maximum releases for a week	20
Simulation time in weeks	1000

of each movie is assumed to decrease exponentially with time and a rate parameter is assigned to a movie between 0 and 1. Finally, a movie is assigned a size that is randomly generated via the MDF. Thus, each new movie is assigned a size, popularity, and change parameters which are sufficient to capture its long-term behavior in the system. The system is seeded with an initial population of 60 movies and the movie popularities are assigned the standard Zipf distribution.

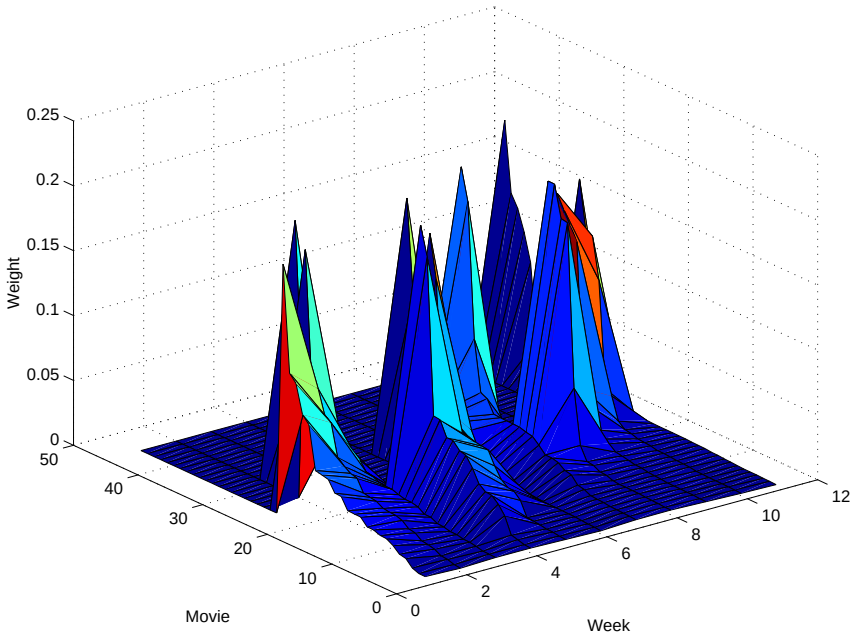


Figure 5.4: Weekly Popularity Distribution

Fig. 5.4 illustrates the distribution of popularities for a 11 week period for weeks 500-510 of the simulation. We see that with the applied load, not only does the popularity vector change, but the number of movies in the system also changes. This behavior partly due to the variable burst size and also due to the uneven rate of change of popularity among the different objects. Consequently, the user requests for movies varies with time and the system must support a variable number of movies.

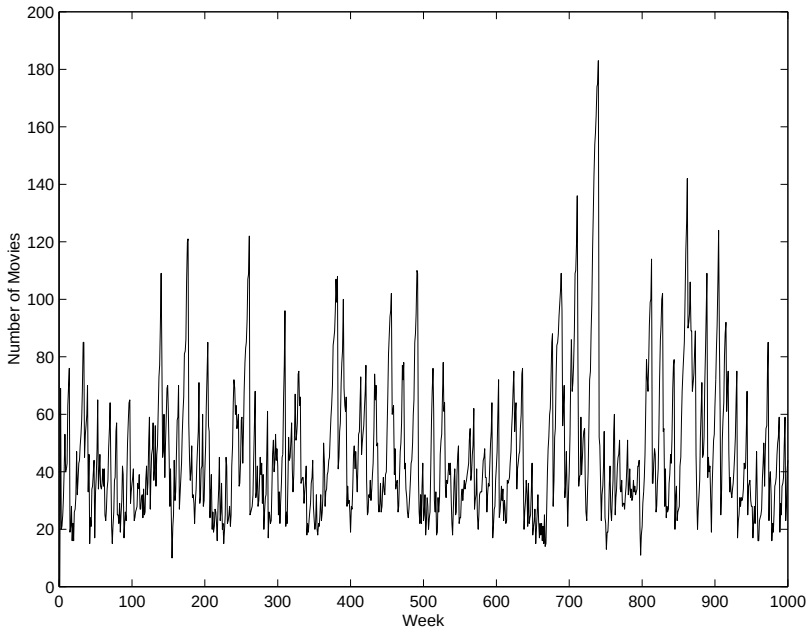


Figure 5.5: Number of Movies Requested Per Week

The number of movies in the system over the 1,000 week simulation period is illustrated in Fig. 5.5. The average number of movies requested by the user in a given week is 46, the maximum number of movies is 183, the minimum is 10 and the standard deviation is 24.

Fig. 5.6 illustrates the storage capacity necessary to support the requested movies as determined from the simulation. The average storage space necessary to support the system load is 48 GB. The minimum space is 11 GB, and the maximum

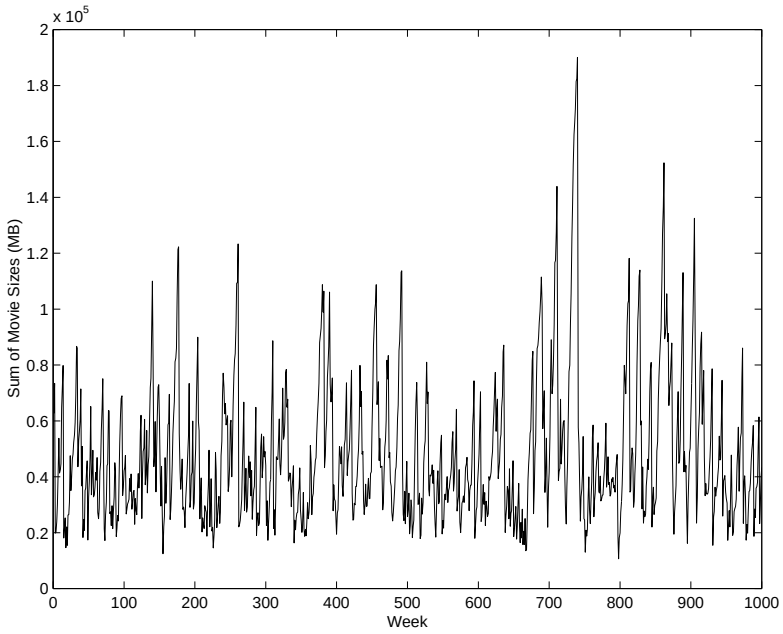


Figure 5.6: Weekly Storage Requirements

capacity is 190 GB. In this particular case, the week with a maximum number of movies also corresponds to the one with the greatest capacity requirement. However, this is not always true.

The simulation study demonstrates that characterizing the system requirements necessitates specifying worst case load scenarios. Once the worst load has been specified, we can estimate the number of disks necessary to support the given load using the parameters of the Barracuda drive as specified in Chapter 3. For the scenario described in this chapter, the configuration that uses the entire disk and minimizes C_u is derived as $B = 684KB$, and 22 drives to support 1,000 sessions. Using the entire disk surface, the system requires a total of 43 drives 4.5GB each or 21 9GB drives. Thus, we see that the disk configuration that best meets the capacity and bandwidth requirements and minimizes the disk count contains 22 9GB drives.

Chapter 6

Conclusions and Future Work

The design of a large scale storage server is a complex process. A practical design must balance many conflicting requirements. The most basic tradeoff in server design is between utilization and response time. In this dissertation, we consider these tradeoffs in the context of CM servers where data are streamed to the end-user and sessions are long-lived.

Disk based storage is currently the preferred technology for building large scale storage servers. In Chapter 3, we analyzed disk system performance for streaming CM data. We analyzed the C-SCAN disk scheduling algorithm and derived a formula to estimate the number of concurrent streams that a disk can support for a given stream bandwidth. We subsequently analyzed the behavior of this function for a range of session bandwidths. We next considered the effects of Zone Bit Recording to evaluate capacity bandwidth tradeoffs in modern drives. It was demonstrated that selectively grouping zones to take advantage of higher bandwidths on the outer disk zones can support more sessions from a disk. Finally, we applied a system cost model to evaluate the best disk configuration for a given session bandwidth.

Our analysis demonstrates that the mapping of session bandwidth to disk performance is non-linear and non-monotonic. We can conclude from this observation that disk based server architectures are inherently biased towards some session bandwidths. In general, we demonstrate that disk storage performs more efficiently for high-bandwidth applications. However, as was demonstrated, very high bandwidth applications can also suffer from poor performance due to the constraint that a disk can support only an integral number of sessions. Consequently, data striping must be employed to ensure high disk utilization.

For low bandwidth streams, replication or techniques that cache data in memory are preferable to disk level streaming of CM data. This recommendation arises out of the observation that applying the conventional streaming model yields extremely large response times during periods of high disk utilization. Finally, the behavior of the cost function can be influenced by the choice of the buffer size. Using a k buffer scheme (where the buffer per stream is $k * B$), can change the behavior of the cost function. Setting $k < 2$ yields configurations that can more efficiently support low bandwidth streams. However, this configuration has little effect on the high bandwidth streams where disk costs dominate buffer costs and require careful control of the buffer in the server operating system.

Optimizing disk configurations for server operation only addresses one aspect of server design. User access behavior and data characteristics also affect server performance. In Chapter 4, we considered macro level issues in server design. To characterize the changing popularities of media objects in a CM server, we studied the properties of user access behavior in WWW servers and movie theaters. These data were used as a basis to develop a model for qualifying changing user preferences and to describe the fluctuations in the load a CM server experiences. We also evaluated the criteria for load balancing in replicated storage servers to minimize session

blocking probability. We also developed a formulation for smoothing the bandwidth requirements during server reorganization based on a given server load.

Finally, we considered a model to describe the size of video objects in a video server. This model allows us to describe a video object size distribution that is based on movie running times as described in the Internet Movie Database. The model was used as input into a simulation model for characterizing the long term access behaviors of user accesses in a CM server with a fixed user population. We demonstrated the application of the user models to estimate worst case storage requirements in a CM server. We also illustrated the application of the worst case load to determine a server configuration with the least number of disks.

We can conclude from this study that a careful evaluation of the system load parameters must be considered to evaluate a configuration that best meets the performance requirements. Assuming a fixed skew in object popularity cannot characterize short term variations in server load and can yield unsatisfactory configurations.

6.1 Future Work

This work can be extended in several interesting and useful directions. We list some of these possibilities below:

- In our study, we ignored disk interconnect architectures and their effects on system performance. Developing a mapping between system cost and the Zone-Group architecture can yield a better estimate of the capacity bandwidth trade-offs for a drive.
- Our study assumes that data are streamed to each user from the CM server

with a fixed bandwidth. This assumption can be relaxed to include systems with different bandwidth requirements.

- In our study, we assume that each user is allocated a single stream. It is feasible to develop server architectures using dynamic service aggregation protocols to recover server bandwidth by aggregating users. We have conducted some initial work in this direction [38, 66].

- The study can also be extended to include disk and system failure models and their effect on server operation and cost.

In summary, the performance of a CM server is affected by many factors including the disk architecture, session bandwidth, user access behavior and the characteristics of the media objects. Disk characteristics are inherently biased against the application of the streaming model for low bandwidth sessions. Furthermore, disk storage constraints preclude efficient disk utilization for any session bandwidth. Efficiency of disk operation can be improved by balancing buffering with utilization to minimize the cost to the end-user. Determining the proper server configuration requires not only the session bandwidth but also the characteristics of the media objects and the nature of user accesses.

Bibliography

[1] K. Almeroth and M. Ammar, "On the Performance of a Multicast Delivery Video-on-Demand Service with Discontinuous VCR Actions," *Proc. Intl. Conf. on Communications (ICC '95)*, Seattle WA, June 1995, pp. 1631-1635.

[2] K. Almeroth and M. Ammar, "A Scalable, Interactive Video-On-Demand Service Using Multicast Communication," *Proc. Intl. Conf. on Computer Communication and Networks (IC3N '94)* , San Francisco CA, September 1995, pp. 292-296.

[3] R. Alonso, D. Barbara, and H. Garcia-Molina, "Data Caching Issues in an Information Retrieval System," *ACM Transactions on Database Systems*, Vol. 15, No. 3, September 1990, pp. 359-384.

[4] D. P. Anderson, Y. Osawa, and R. Govindan, "A File System for Continuous Media," *ACM Transactions on Computer Systems*, Vol. 10, No. 4, November 1992, pp. 311-337.

[5] P. Asthana, B.I. Finkelstein, and A.A. Fennema, "Rewritable Optical Disk Drive Technology," *IBM Journal of Research and Development*, Vol. 30, No. 5, 1996.

[6] S. Berson, S. Ghandeharizadeh, R. Muntz, and X. Ju, "Staggered Striping in Multimedia Information Systems," *Proc. ACM SIGMOD*, 1994, pp. 79-89.

[7] A. Bestavros, R. Carter, M. Crovella, C. Cunha, A. Heddaya, and S. Mirdad, "Application-Level Document Caching in the Internet," *Proc. 2nd Intl. Workshop on Services in Distributed and Networked Environments (SDNE'95)*, Whistler, Canada, 1995.

[8] Y. Birk, "Track-Pairing: A Novel Data Layout for VOD Servers with Multi-Zone-Recording Disks," *Proc. 2nd. IEEE Intl. Conf. on Multimedia Communication Systems (ICMCS'95)*, Washington D.C., May 1995, pp. 248-255.

[9] W. Bolosky, J. Baerra, R. Draves, R. Fitzgerald, G. Gibson, M. Jones, S. Levi, N. Myhrvhold, and R. Rashid, "The Tiger Video Fileserver," *Proc. 6th Intl. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'96)*, Zushi, Japan, April 1996, pp. 97-104.

[10] M.M. Buddhikot and G.M. Parulkar, Efficient Data Layout, Scheduling, and Playout Control in MARS, *Proc. 5th Intl. Workshop on Network and Operating System Support for Digital Audio (NOSSDAV'95)*, Durham NH, April 1995, pp. 339-350.

[11] L.D. Catledge and J.E. Pitkow, "Characterizing Browsing Strategies in the World-Wide Web," *Electronic Proc. 3rd Intl. World-Wide Web Conference*, http://www.igd.fhg.de/www/www95/proceedings/proceedings.html, 1995.

[12] E. Chang and A. Zakhor, "Admissions Control and Data Placement for VBR Video Servers," *Proc. 1st IEEE Intl. Conf. on Image Processing*, Austin, Texas, November 1994, pp. 278-282.

[13] E. Chang and A. Zakhor, "Variable Bit Rate MPEG Video Storage on Parallel Disk Arrays," *Proc. 1st Intl. Workshop on Community Networking*, San Francisco, CA, July 1994 pp. 127-137.

[14] M.-S. Chen, D.D. Kandlur, and P.S. Yu, "Optimization of the Grouped Sweeping Scheduling (GSS0 With Heterogeneous Multimedia Streams," *Proc. ACM Multimedia' 93*, Anaheim, CA, August 1993, pp. 235-242.

[15] M.-S. Chen, D.D. Kandlur, and P.S. Yu, "Support for Fully Interactive Playout in a Disk Array Based Video Server, *Proc. ACM Multimedia'94*, San Francisco CA, Oct 1994, pp. 391-398.

[16] H.-J. Chen and T.D.C. Little, "Physical Storage Organization for Time-Dependent Multimedia Data," *Proc. 4th ACM Intl. Conf. on Foundations of Data organizations and Algorithms (FODO'93)*, Evanston IL, Oct 1993, pp. 19-34.

[17] H.-J. Chen, T.D.C. Little, and D. Venkatesh, "A Storage and Retrieval Technique for Scalable Delivery of MPEG-Encoded Video," *Journal of Parallel and Distributed Computing*, Vol. 30., No. 2, November 1995, pp. 180-189.

[18] M.-S. Chen and D.D. Kandlur, "Stream Conversion to Support Interactive Video Playout," *IEEE Multimedia*, Vol. 3, No. 2, Summer 1996, pp. 51-58.

[19] H.-J. Chen, A. Krishnamurthy, T.D.C. Little, and D. Venkatesh, "A Scalable Video-on-Demand Service for the Provision of VCR-Like Functions," *Proc. 2nd IEEE Intl. Conf. on Multimedia Computing Systems* (ICMCS'95), Washington D.C., May 1995, pp. 65-72

[20] H.-J. Chen, "A Disk Scheduling Scheme and MPEG Data Layout Policy for Interactive Video Access from a Single Disk Storage Device," *Ph.D. Dissertation*,

Dept. of Electrical, Computer and Systems Engineering, Boston University, August 24, 1995.

[21] A. Cohen, W.A. Burkhard, and P.V. Rangan, "Pipelined Disk Arrays for Digital Movie Retrieval," *Proc. 2nd IEEE Intl. Conf. on Multimedia Computing and Systems (ICMCS'95)*, Washington DC, May 1995.

[22] S. Danø, *Nonlinear and Dynamic Programming,* Springer Verlag Ltd., New York, 1975.

[23] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling Policies for an On-Demand Video Server with Batching," *Proc. ACM Multimedia'94*, San Francisco, October 1994, pp. 15-23.

[24] A. Dan and D. Sitaram, "An Online Video Placement Policy on a Bandwidth to Space Ratio (BSR)," *Proc. ACM SIGMOD'95*, May 1995, pp. 376-385.

[25] Y.N. Doganata and A.N. Tantawi, "A Video Server Cost/Performance Estimator Tool," *Journal of Multimedia Tools and Applications,* Vol. 1, No. 2, June 1995, pp. 185-202.

[26] R. Elmasri and S.B. Navathe, *Fundamentals of Database Systems,* Benjamin/Cummings Publishing Co., Menlo Park CA, 1989.

[27] Entertainment Weekly Online, "Box Office Chart," *Entertainment Weekly Inc.,* http://www.pathfinder.com/ew, 1995.

[28] D. Ferrari and D. Verma, "A Scheme for Real-Time Channel Establishment in Wide Area Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, April 1990, pp. 368-379.

[29] A.D. Gelman, H. Kobrinski, L.S. Smoot, S.B.Weinstein, M. Fortier, and D. Lemay, " A Store-and-Forward Architecture for Video-on-Demand service," *Proc. IEEE ICC*, 1991, pp. 27.3.1-27.3.5.

[30] D.J. Gemmell, J. Han, R.J. Beaton, and S. Christodoulakis, "Delay-Sensitive Multimedia on Disks," *IEEE Multimedia*, Vol. 1, No. 3, Fall 1994, pp. 56-67.

[31] J. Gemmell and S. Christodoulakis, "Principles of Delay Sensitive Multimedia Data Storage and Retrieval," *ACM Transactions on Information Systems*, Vol. 10, No. 1, January 1992, pp. 51-90.

[32] S. Ghandeharizadeh, S.H. Kim, and C. Shahabi, "Continuous Display of Video Objects Using Multi-Zone Disks," *Technical report,* Univ. Southern California Computer Science Dept., 1995.

[33] L. Golubchik, J.C.S. Lui, and R. Muntz, "Reducing I/O Demand in Video-on-Demand Storage Servers," *Technical Report 940037*, Univ. California at Los-Angles Computer Science Dept., 1994.

[34] J.P. Hayes, *Computer Architecture and Organization,* McGraw Hill Book Co., New York, 1988.

[35] J.L. Hennessy and D.A. Patterson, *Computer Architecture: A Quantitative Approach,* Morgan Kaufmann Publishers Inc., Palo Alto CA, 1990.

[36] K. Keeton and R.H. Katz, "The Evaluation of Video Layout Strategies on a High-Bandwidth File Server," *Proc. 4th Intl. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'93)*, Lancaster, U.K., November 1993, pp. 237-248.

[37] L. Kleinrock, *Queueing Systems Vol. 1.,* John Wiley and Sons, New York, 1975.

[38] R. Krishnan, D. Venkatesh and T.D.C. Little, "A Failure and Overload Tolerance Mechanism for Continuous Media Servers," *Proc. ACM Multimedia'97*, Seattle WA, Nov 1997, pp. 131-142.

[39] M. Kumar, J.L. Kouloheris and M.J. McHugh, "A High Performance Multimedia Server For Broadband Network Environment," *IBM-TR RC-20059*, IBM Corp., 1995.

[40] F.W. Lancaster, *The Measurement and Evaluation of Library Services,* Information Resources Press, Urbana IL, 1977.

[41] A. Laursen, J. Olkin and M. Porter, "Oracle Media Server: Providing Consumer Based Interactive Access to Multimedia Data," *Proc. ACM SIGMOD'94*, pp. 470-477.

[42] E.K. Lee, "Performance Modeling and Analysis of Disk Arrays," *PhD Thesis*, Dept. of Computer Science, University of California at Berkeley, 1993.

[43] T.D.C. Little and D. Venkatesh, "Client-Server Metadata Management for the Delivery of Movies in a Video-on-Demand System," *Proc. 1st Intl. Workshop on Services in Distributed and Networked Environments (SDNE)*, June 27-28, Prague, Czech Republic, 1994, pp. 11-18.

[44] T.D.C. Little and D. Venkatesh, "Prospects for Interactive Video-on-Demand," *IEEE Multimedia*, Vol. 1, No. 3, Fall 1994, pp. 14-24.

[45] T.D.C. Little and D. Venkatesh, "Popularity-Based Assignment of Movies to Storage Devices in a Video-on-Demand System," *Multimedia Systems*, Vol. 2, No. 6, January 1995, pp. 280-287.

[46] T.D.C. Little, G. Ahanger, H.-J. Chen, R.J. Folz, J.F. Gibbon, A. Krishnamurthy, P. Lumba, M. Ramanathan, and D. Venkatesh "Selection and Dissem-

ination of Digital Video via the Virtual Video Browser," *Journal of Multimedia Tools and Applications*, Kluwer Publications, Vol. 1, No. 2, June 1995, pp. 149-172.

[47] J.C.L. Liu, D.H.C. Du and J.A. Schnepf, "Supporting Random Access on Real-Time Retrieval of Digital Continuous Media," *Computer Communications*, Vol. 18, No. 3, March 1995, pp. 145-159.

[48] P. Lougher and D. Shepherd, "The Design of a Storage Server for Continuous Media," *The Computer Journal*, Vol. 36, No. 1, February 1993, pp. 32-42.

[49] ISO/IEC, "Information technology - Generic coding of moving pictures and associated information," *ISO/IEC 13818(1-6) (International Standard)*, 1995.

[50] M. Nelson, M. Linton, and S. Owicki, "A Highly Available Scalable ITV System," *Proc. Symposium on Operating Systems (SOSP15)*, 1995, pp. 54-67.

[51] G. Neufeld, D. Makaroff and N. Hutchinson "Design and Implementation of a Variable Bit File Server," *Proc. 5th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'95)*, April 1995, pp. 375-378.

[52] J.-P. Nussbaumer, B.V. Patel, F. Schaffa, and J.P.G. Sterbenz, "Networking Requirements for Interactive Video on Demand," *IEEE Journal on Selected Areas in Communications,* Vol. 13, No. 5, June 1995, pp. 779-787.

[53] R. Ramarao, and V. Ramamoorthy, "Architectural Design of On-Demand Video Delivery Systems: The Spatio-Temporal Storage Allocation Problem," *Proc. ICC'91*, 1991, Denver CO, pp. 17.6.1-17.6.5.

[54] K.K. Ramakrishnan, L. Vaitzblit, C. Gray, U. Vahalia, D. Ting, P. Tzelnic, S. Glaser, and W. Duso, "Operating System Support for a Video-on-Demand

Service," *Proc. 4th International Workshop on Network and Operating System Support For Digital Audio and Video (NOSSDAV'93)*, Lancaster, England, November 1993, pp. 225-236.

[55] P. V. Rangan, H. M. Vin, and S. Ramanathan, "Designing an On-Demand Multimedia Service," *IEEE Communications Magazine*, Vol. 30, No. 7, July 1992, pp. 56-64.

[56] A.L.N. Reddy and J. Wylie, "Disk Scheduling in a Multimedia I/O System," *Proc. ACM Multimedia'93*, Anaheim, CA, August 1993, pp. 225-233.

[57] C. Ruemmler and J. Wilkes "An Introduction to Disk Drive Modeling," *Computer*, March 1994, pp. 17-28.

[58] Seagate Corp., *Seagate Product Specification: ST11200N,* Seagate Corp., 1995.

[59] Seagate Corp, *Seagate Product Specification: ST15150 Barracuda Disk Drive,* Seagate Corp., 1995.

[60] P.J. Shenoy and H.M. Vin, "Efficient Support for Scan Operations in Video Servers," *Proc. ACM Multimedia'95*, San Francisco CA, November 1995, pp. 131-140.

[61] A. Silberschatz, J. Peterson, and P. Galvin, *Operating System Concepts,* Addison Wesley, Reading MA, 1990.

[62] R. Steinmetz, "Multimedia File Systems Survey: Approaches for Continuous Media Disk Scheduling," *Computer Communications*, Vol. 18, No. 3, November 1995, pp. 133-144.

[63] A.S. Tanenbaum, *Operating Systems: Design and Implementation,* Prentice-Hall, New York, 1988.

[64] L. Vaitzblit, "A High-Resolution Video Server for Cinema of the Future," *IEEE Multimedia*, Vol. 2, No. 2, Fall 1995, pp. 65-69.

[65] D. Venkatesh and T.D.C. Little, "Dynamic Service Aggregation for Efficient Use of Resources in Interactive Video Delivery," *Proc. 5th Intl. Workshop on Network and Operating System Support for Digital Audio and Video* (NOSSDAV'95), Durham, New Hampshire, April 19-22, 1995, pp. 119-122.

[66] D. Venkatesh and T.D.C. Little, "Investigation of Web Server Access as a Basis for Designing Video-on-Demand Systems," *Proc. 1st Intl. Symposium on Photonics Technologies and Systems for Voice, Video, and Data Communications*, SPIE Vol. 2617-06, Philadelphia PA, October 1995.

[67] D. Venkatesh and T.D.C. Little, "Evaluating the Cost-Performance Characteristics of Disk Storage Systems Supporting Digital Video Content," *Proc. 6th Intl. Workshop on Network and Operating System Support for Digital Audio and Video*, Zushi, Japan, April 1996, pp 139-148.

[68] D. Venkatesh and T.D.C. Little, "The Use of Media Characteristics and User Behavior for the Design of Multimedia Servers," *Multimedia Information Storage and Management,* S.M. Chung (ed.), Kluwer Academic Publishers, Norwell MA, 1996, pp. 95-116.

[69] H. M. Vin and P. V. Rangan, "Designing a Multi-User HDTV Storage Server," *IEEE Journal on Selected Areas in Communications* Vol. 11, No. 1, January 1993, pp. 153-164.

[70] H.M. Vin, A. Goyal, A. Goyal and P. Goyal, "An Observation Based Admission Control Algorithm for Multimedia Servers," *Proc. 1st IEEE Intl. Conf. on*

Multimedia Computing and Systems (ICMCS'94), Boston MA, May 1994, pp. 234-243.

[71] G. Widerhold, *File Organization for Database Design,* McGraw Hill Book Co., New York, 1987.

[72] B.L. Worthington, G.R. Ganger, Y.N. Patt, and J. Wilkes, "On-Line Extraction of SCSI Disk Drive Parameters," *Proc. ACM Sigmetrics*, May 1995, pp. 146-156.

[73] P.S. Yu, J.L. Wolf, and H. Shachnai, "Design and Analysis of a Look-Ahead Scheduling Scheme to Support Pause-Resume for Video-on-Demand Applications," *Multimedia Systems*, Vol. 3, No. 4, 1995, pp. 137-149.

Appendix A1

Disk Drive Specifications

Table A1.1: Drive Specifications

Drive	p	T_{avg}	T_{max}	T_{min}	T_{rot}	ω
Seagate Barracuda	10	8.0 ms	17.0 ms	0.8 ms	4.17 ms	7200
Seagate Hawk	6	10.5 ms	20.0 ms	1.5 ms	5.40 ms	5400

Table A1.2: Seagate Barracuda Zone Specification

Zone	7	6	5	4	3	2	1
Sectors/Track	122	135	148	162	175	189	196
Tracks/Zone	483	583	684	849	918	572	1178

Table A1.3: Seagate Hawk Zone Specification

Zone	23	22	21	20	19	18	17	16	15	14	13	12
Sectors/Track	54	56	57	61	62	62	62	65	71	73	74	75
Tracks/Zone	83	82	168	35	71	36	71	157	79	80	41	81
Zone	11	10	9	8	7	6	5	4	3	2	1	
Sectors/Track	76	76	77	80	83	84	88	91	92	93	94	
Tracks/Zone	80	40	79	78	38	144	68	33	93	30	205	

Appendix A2

Behavior of N from the C-SCAN Approximation

In this section, we consider the behavior of N as described in Eq. 3.5. In Eq. 3.5, the value of N is obtained by solving for the roots of the quadratic equation:

$$N^2(\phi^2 + a^2) - N(2\phi\mu + a^2\Theta) + \mu^2 = 0 \tag{A1}$$

where

$$\phi = \frac{B}{R_d} - b + c + T_{rot} \quad \text{and}$$

$$\mu = \frac{B}{R_c} - b\Theta.$$

The standard solution to any quadratic equation $Ax^2 + Bx + C = 0$ is $x = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$. The denominator in the solution consists of the linear term and a square-root value.

Fig. A2.1 illustrates the behavior of the linear term, given by $2\phi\mu + a^2\Theta$ for bandwidths of 8 Kb/s, 128 Kb/s and 1.5 Mb/s for the Barracuda drive. Since μ is inversely proportional to R_c, the linear term is highest for 8 Kb/s.

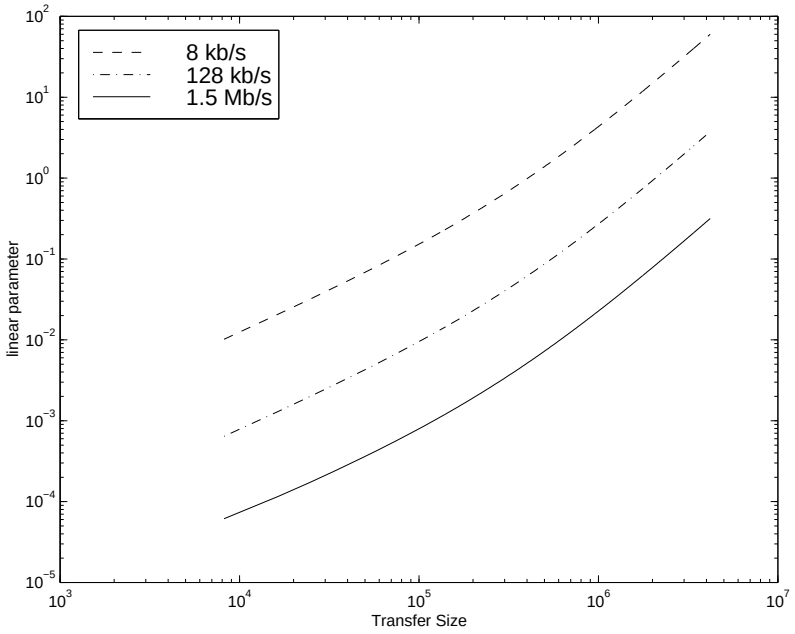


Figure A2.1: The Linear Term vs. B

Figs. A2.2 illustrates the values of the quadratic term and Fig. A2.3 illustrates the ratios of the linear and square-root term.

Fig. A2.3 illustrates that the linear term dominates the square-root term by three orders of magnitude and can hence be ignored in computing N. Fig. A2.3 also demonstrates a singularity for the 8 Kb/s stream. The singularity represents the point when the square-root term becomes negative. The range of session bandwidths for which the square-root term can be negative, yielding imaginary values for N is given by $R_c \leq R_d/\Phi$. However, N can be approximated as

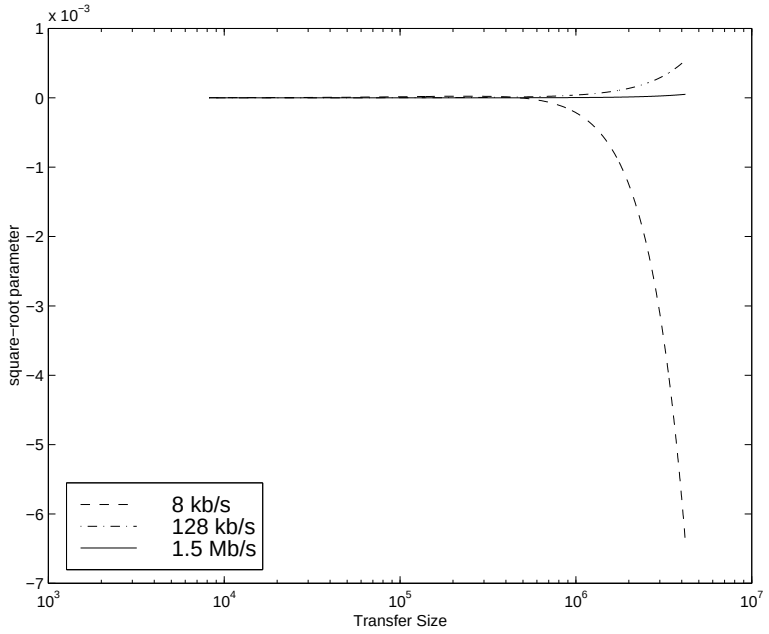


Figure A2.2: The Square-Root Parameter vs. B

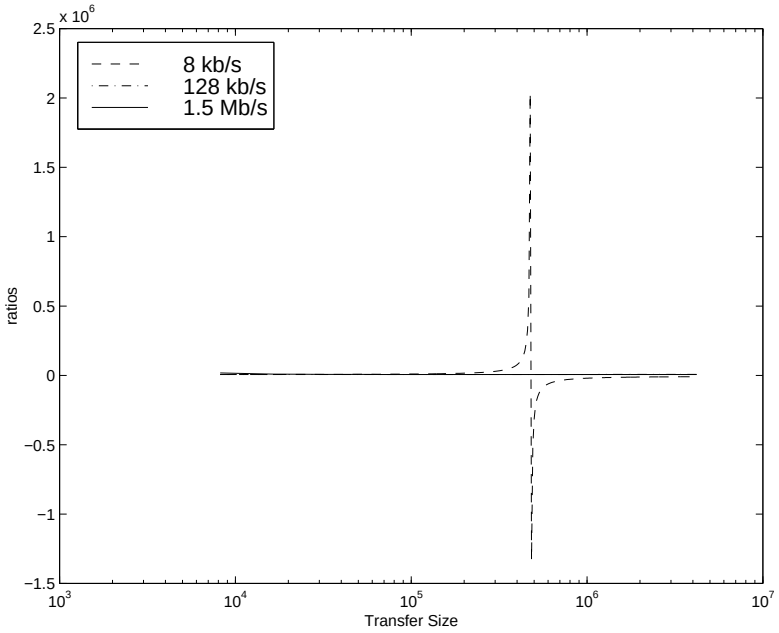


Figure A2.3: Ratios of the Linear and Square-Root Parameters vs. B

$$N \approx \left\lfloor \frac{2\phi\mu + a^2\Theta}{2 * (\phi^2 + a^2)} \right\rfloor \tag{A2}$$

when $R_c \leq R_d/\Phi$ which yields an accurate estimate of N

For high bandwidth sessions, very small transfer sizes yield negative linear values when $B/R_c << b\Theta$. For this scenario, evaluating the constraint $R_d > R_c$ allows us to evaluate whether a session can be supported from the disk. We now illustrate an algorithmic procedure for evaluating N given the disk parameters along with B and R_c.

1. $\phi = B/R_d - b + c + T_{rot}$
2. $\mu = B/R_c - b\Theta$
3. $sq = (2\phi\mu + a^2\Theta)^2 - 4\mu^2(\phi^2 + a^2)$
4. if $(sq > 0)$
5. $\quad N = \left\lfloor \frac{(2\phi\mu + a^2\Theta) - \sqrt{sq}}{2(\phi^2 + a^2)} \right\rfloor$
6. else
7. $\quad N = \left\lfloor \frac{(2\phi\mu + a^2\Theta)}{2(\phi^2 + a^2)} \right\rfloor$
8. if $(N == 0)$
9. $\quad$ if $(R_d > R_c)$
10. $\quad\quad N = 1$

The resulting N must be evaluated against the scheduling constraint in Eq. 3.4 to ensure that the estimated value of N is correct. N can be incremented to evaluate

whether the drive can support additional sessions. However, empirical data using parameters from the Barracuda drive demonstrates that the value of N as derived always satisfies the constraints of Eq. 3.5 and incrementing N cannot satisfy the constraints.

Appendix A3

Effects of Disk Transfer Overheads on Disk Efficiency

In this section, we consider the effects of disk transfer overheads on disk performance. Ideally, ample bandwidth is available on the disk-host I/O bus to ensure immediate transfer of data from the disk buffer to the host as soon as data is available on the disk. However, in reality, overheads in the I/O protocol and host operating system preclude efficient utilization of the disk. We now describe measurement studies on the Barracuda drive to evaluate its performance. It is demonstrated that including the transfer overheads yields a more reliable estimate of the streaming ability of the drive.

Fig. A3.1 illustrates usable drive throughput estimated by application of the disk model in Eq. 3.5 for the Barracuda drive. *oh* represents the behavior of N for values of R_d based on the drive characteristics described in Fig. 3.11. *optim* represents the behavior of N for R_d based on the manufacturers specifications. *meas* represents the measured throughput for a system with 32 seeks (sessions).

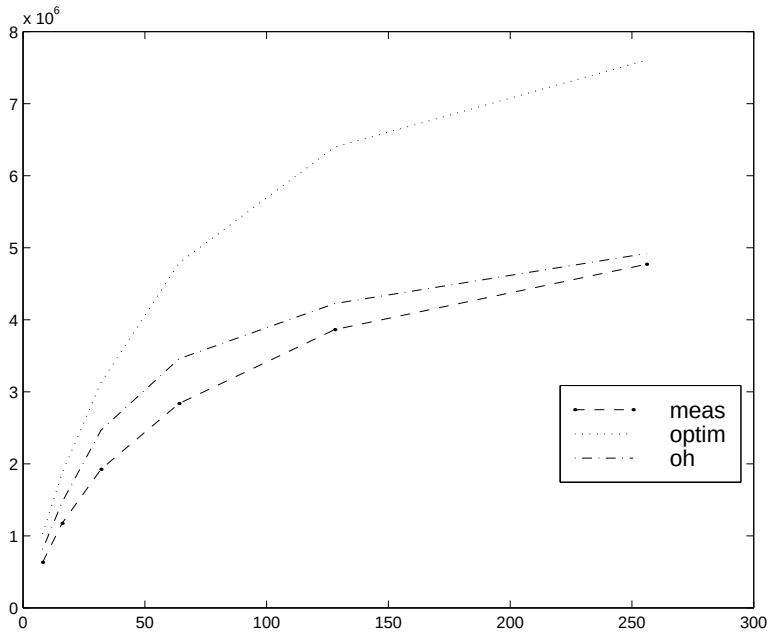


Figure A3.1: Disk Throughput vs. Transfer Size

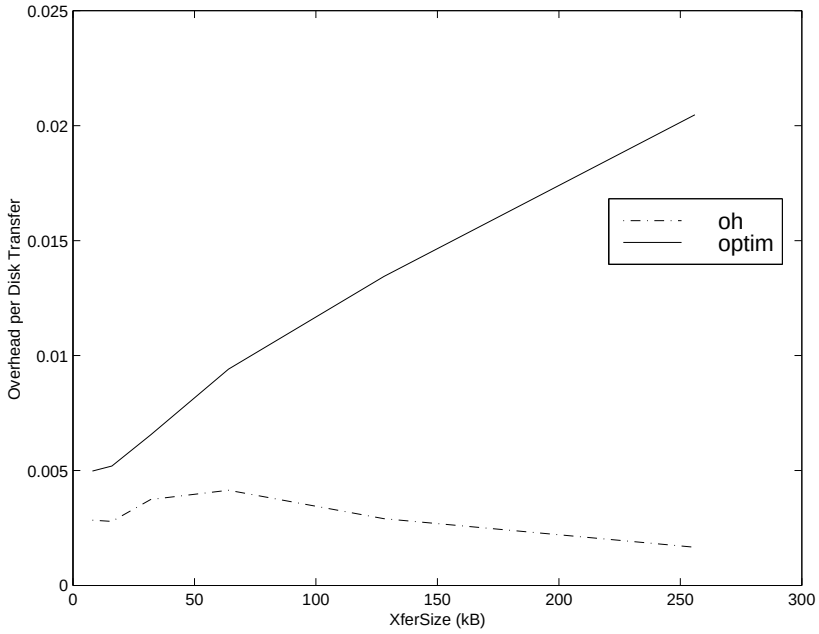


Figure A3.2: Overhead Per Transfer vs. Transfer Size

It is apparent from Fig. A3.1 that the disk models are optimistic and yield throughput estimates that exceed the observed value significantly. In order to more accurately estimate disk performance, we must include these additional overheads in the disk model. The overhead parameter is computed on the basis of disk transfers. Let O represent the I/O rate (transfers per second) as determined by the model and O' represent the measured I/O rate observation. We define the overhead per transfer T_{oh} as

$$T_{oh} = \frac{O - O'}{OO'}. \tag{A1}$$

Fig. A3.2 illustrates the value of T_{oh} for the two models from Fig. A3.1.

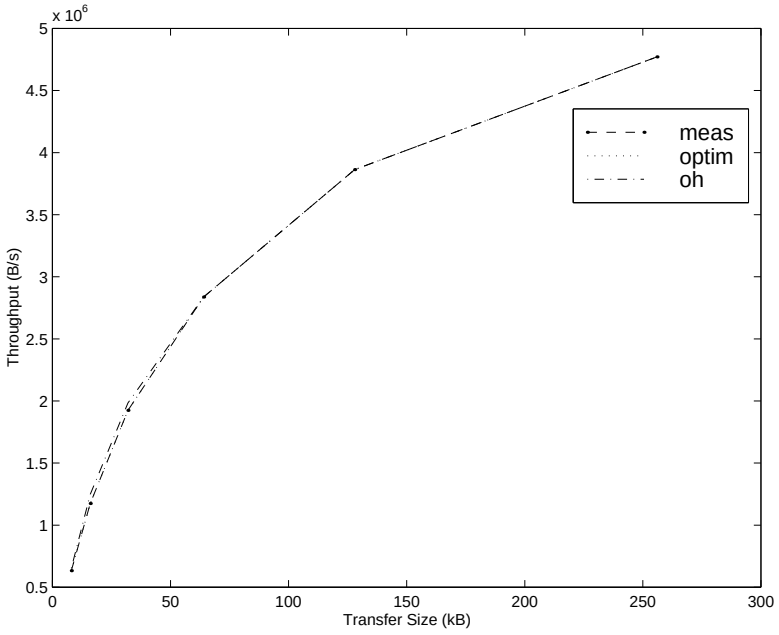


Figure A3.3: Effects of Including T_{oh}

Since T_{oh} is a constant value, its inclusion in the disk-model is straightforward. We note that T_{rot} is a constant overhead parameter and is independent of the number

of sessions on the disk. Consequently, we replace T_{rot} with $T_{rot} + T_{oh}$ in the model. The resulting behavior for the disk models is illustrated in Fig. A3.3. It is clear that including T_{oh} in the disk model yields a fairly accurate estimate of the streaming capabilities of the drive.

Appendix A4

Minimum Cost Computation

In this section, we describe the computation of the minimum cost buffer size and the behavior of the cost function. The cost per unit bandwidth is defined by Eq. 3.24

$$C_u = \frac{2BC_B}{R_c} + \frac{DdC_d * 2(\phi^2 + a^2)}{R_c(2\phi\mu + a^2\Theta)} \tag{A1}$$

which is a function of B. The minimum value of C_u is obtained by solving

$$\frac{dC_u}{dB} = 0. \tag{A2}$$

We Expand C_u as

$$C_u = \frac{1}{R_c}\left\{2C_B B + \frac{2DdC_d R_c(B^2 - 2BkR_d + (a^2 + k^2)R_d^2)}{R_d\{2B^2 - 2B(kR_d + \Theta bR_c) + R_d R_c(2bk\Theta + a^2\Theta)\}}\right\}$$

Reorganizing, we obtain

$$C_u = \frac{1}{R_c R_d}\left\{\frac{B^3(4C_B R_d) + B^2(\mathcal{S} - 4\mathcal{M}C_B R_d) + B(2C_B R_d \mathcal{T} - 2\mathcal{S}kR_d) + \mathcal{S}\mathcal{Y}}{2B^2 - 2B\mathcal{M} + \mathcal{T}}\right\} \tag{A3}$$

where

$$\mathcal{S} = 2DdC_d R_c$$

$$\mathcal{M} = \Theta b R_c + kR_d$$

$$\mathcal{T} = R_c R_d \Theta(2kb + a^2)$$

$$\mathcal{Y} = (a^2 + k^2)R_d^2 \qquad \text{and}$$

$$k = b - c + T_{rot}.$$

Differentiating, we obtain the condition for minimum cost as

$$\begin{aligned} &B^4(-8C_B R_d) + \\ &B^3(16C_B R_d \mathcal{M}) + \\ &B^2(2\mathcal{M}\mathcal{S} - 4\mathcal{S}kR_d - 8\mathcal{M}^2 C_B R_d - 8C_B R_d \mathcal{T}) + \\ &B(8\mathcal{L}\mathcal{S} - 2\mathcal{S}\mathcal{T} + 8C_B R_d \mathcal{M}\mathcal{T}) + \\ &2kR_d \mathcal{S}\mathcal{T} - 2C_B R_d \mathcal{T}^2 - 2\mathcal{M}\mathcal{S}\mathcal{Y} = 0. \end{aligned} \tag{A4}$$

Figs. A4.1 and A4.2 illustrate the behavior of Eq. A4 for a range of practical values of B. It is apparent that for the range of bandwidths considered, a solution to Eq. A4 has only one valid root in the desired range. Furthermore, for large values of B, The polynomial in Eq. A4 is negative. Any reduction technique can be easily applied to determine the correct value of B that minimizes C_u.

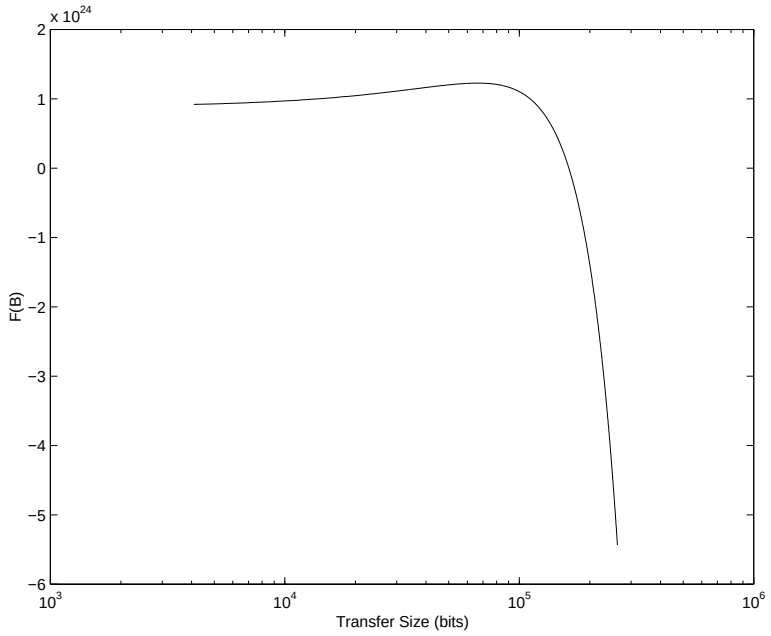


Figure A4.1: Behavior of Eq. A4 for 8 Kb/s Streams

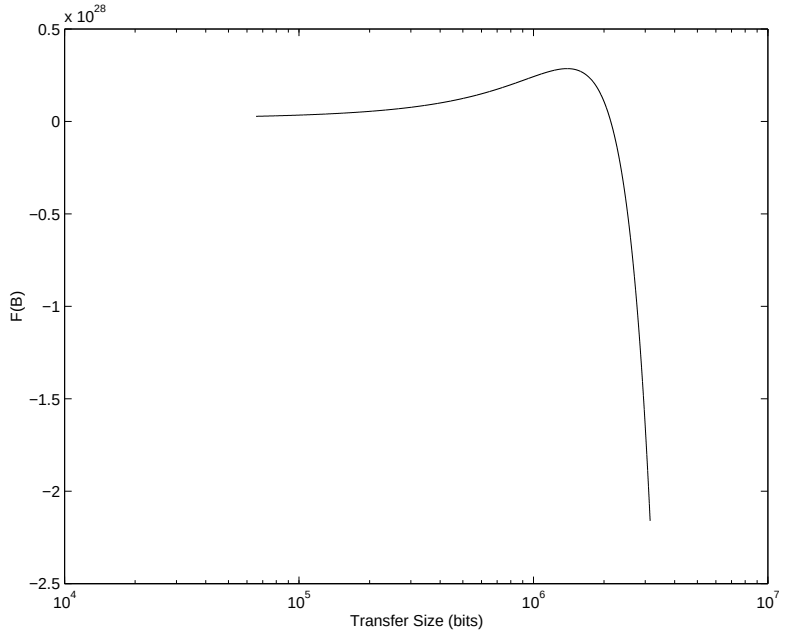


Figure A4.2: Behavior of Eq. A4 for 1.5 Mb/s Streams

Appendix A5

Justification of Lemma 3

Lemma 3 claims that an equiprobable allocation of objects to disks yields the least probability of blocking. We note that this problem is similar to other load balancing problems. By making all disks are equiprobably accessible, we attempt to avoid the formation of bottlenecks of system performance.

From Chapter 4, the allocation problem is to minimize

$$P_B = \sum_{i=1}^{d} \frac{s_i^{l+1}}{\sum_{j=0}^{l} (\frac{\lambda s_i}{\mu})^l / j!}$$

given

$$\sum_{i=1}^{d} s_i = 1, \quad s_i \geq 0.$$

Using the method of Lagrangian multipliers, the optimization function is written as

$$L = \sum_{i=1}^{d} \frac{s_i^{l+1}}{\sum_{j=0}^{l} (\frac{s_i \lambda}{\mu})^j} + k(\sum_{i=1}^{d} s_i - 1).$$

The condition for minimality is

$$\frac{\partial L}{\partial s_i} = 0 \quad \forall i.$$

After differentiation, all the polynomial equations in s_i are identical to one another. As a result, their roots are identical, and a solution is

$$s_i = Af(k) \qquad \text{where } A \text{ and } k \text{ are constants for all } i.$$

This implies that the uniform distribution results in the minimum P_B, justifying our assumption. However, we demonstrate both analytically and via simulations that the balanced system does indeed yield the lowest P_B for both exponential and bimodal holding times.

Biography

Dinesh Venkatesh is a Principal Design Engineer in the Network Storage Group at EMC Corp. of Hopkinton MA, where he designs and implements video storage systems. He has been associated with the Multimedia Communications Laboratory at Boston University throughout the completion of his degree.

Mr. Venkatesh holds a B.E. in Electronics Engineering from Bangalore University, an M.S. in Computer Engineering from Boston University and will recieve the PhD in Computer Engineering from Boston University in December 1997, with the completion of this dissertation. Mr. Venkatesh is a member of the ACM.