A STUDY OF THE HYBRID ADMISSION CONTROL ALGORITHM FOR MULTIMEDIA SERVER

N. M. Mosharaf Kabir Chowdhury

A Thesis Submitted to the Department of Computer Science and Engineering in the Partial Fulfillment of the Requirements for the
Degree of
Bachelor of Science in Engineering
(Computer Science and Engineering)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY
DHAKA, BANGLADESH

NOVEMBER 2006

The thesis **"A Study of the Hybrid Admission Control Algorithm for Multimedia Server",** submitted by N. M. Mosharaf Kabir Chowdhury, Roll No. 0005013, to the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Engineering (Computer Science and Engineering) and approved as to its style and contents. Examination held on November 4, 2006.

Supervisor

____

Dr. Md Mostofa Akbar
Associate Professor
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
Dhaka-1000
Bangladesh

Declaration

I, hereby, declare that the work presented in this thesis is the outcome of the investigation performed by me under the supervision of Dr. Md. Mostofa Akbar, Associate Professor, Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka. I also declare that no part of this thesis and thereof has been or is being submitted elsewhere for the award of any degree or Diploma.

Countersigned	Signature
....	....
(Dr. Md. Mostofa Akbar) Supervisor	(N. M. Mosharaf Kabir Chowdhury)

Acknowledgement

Here I would like to take the opportunity to express my gratefulness to the patrons of this thesis work, without whom I could never have completed this arduous task and I express my heartfelt gratitude to Almighty Allah for His divine blessings in this long journey.

Needless to say, that the only thing that kept me going was the support of a number of people. First and foremost, **Dr. Md. Mostofa Akbar**, Associate Professor, Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, "Without your unstinting encouragement and guidance, constant and energetic supervision, constructive criticism and invaluable advice there is just no way that I could have completed this thesis. You have always been there whenever I needed your help – from listening to my ideas, reading many inferior drafts to emending them at all stages. I thank you for everything".

I would also like to thank **Dewan Tanvir Ahmed**, Assistant Professor, Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, for his constant support to pick up and carry on with the work from the point he left.

I am also very much grateful to **Dr. M. Kaykobad**, Professor, Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, who always inspired me in the vast field of research.

I must acknowledge with due respect the constant support and patience of my family for completing the thesis.

Abstract

A multimedia server has to serve a large number of clients simultaneously. Considering the real-time requirements of each client and constant data transfer rate of storage devices, it must employ admission control algorithms to control client traffic in order to increase utilization of server resources. Hence, the main goal of an admission control algorithm is to accept enough traffic to efficiently utilize server resources, while not accepting clients whose admission may lead to violations of the service requirements of pre-existing clients. In this thesis we are examining a hybrid admission control algorithm that can handle a larger number of clients simultaneously. We also consider the use of multiple storage devices with separate file systems and thus increase the parallelism during disk access which considerably improves the server performance.

The performance of hybrid admission control algorithm is dependent on the disk-scheduling algorithm employed and hence on the time required to retrieve necessary disk blocks to satiate client requests. In this thesis we demonstrate the effect of using different disk scheduling algorithms on the performance of the hybrid admission control algorithm. For each disk scheduling algorithm we consider minimizing both the rotational latency and the seek time. In order to further decrease the service time we ensure parallelism by introducing concurrent disk access through separate file systems for separate disks. To provide continuous retrieval of each media stream, we have to ensure that service time is less than the minimum duration of a round. Since the service time is a function of the number of blocks and their relative positions on the disks, it may exceed the minimum duration of a round. We refer to such rounds as overflow rounds. We also introduce some parameters to restrict overflow rounds within limits.

We have devised an extension to the Hybrid Admission Control Algorithm to provide for multiple storage devices and analyzed its complexity. Finally, we have demonstrated the increment in performance and utilization of multimedia server using both normal and extended algorithm with multi-storage and effects of using different disk scheduling algorithms alongside them through extensive simulation.

Table of Contents

List of Figures

List of Tables

Glossary of Terms

ACA	Admission Control Algorithm
CM	Continuous Media
CDN	Content Delivery Network
DACA	Deterministic Admission Control Algorithm
DM	Discrete Media
ϵ	Parameter to Restrict Admission
η	Average Time to Retrieve a Media Block
FCFS	First Come First Serve
GPFS	General Parallel File System
HACA	Hybrid Admission Control Algorithm
K	Number of Media Blocks Accessed in a Single Round
m	Total Number of Disks in Multi-Disk System
MST	Minimum Spanning Tree
n	Total Number of Client
NODSA	Near Optimal Disk Scheduling Algorithm
NRT	Non Real-time
NUG	Normal User Group
QoS	Quality of Service
R	Minimum Duration of a Round
RT	Real-time
SACA	Statistical Admission Control Algorithm
SCAN	Scan Disk Scheduling Algorithm
SLA	Service Level Agreement
SSTF	Shortest Seek Time First
SUG	Super User Group
τ	Service Time
UG	User Group

Chapter One

Introduction

1.1 Motivation

With the recent advances in computer capabilities, compression technologies and broadband networking audio and video applications have become an integral part of our everyday computational life. It has become necessary for us to provide an integrated environment for the execution of these multimedia applications. It becomes especially relevant in content servers that can serve different kinds of data to various clients. Unlike traditional applications, multimedia applications have quite different resource and performance constraints [1]. They require time-constrained, fair execution environments and periodic access to disks. Execution and retrieval of data for these applications have deadlines by which the application must get all the resources (data, CPU time etc.) to render video or audio. Most multimedia applications are soft real time applications, which mean that some loss is tolerable while delay and jitter can greatly reduce performance [2][3]. Although it is important that a certain amount of data be supplied to the application within a given period of time, it is not necessary that all the requests be satisfied in order to provide reasonable application performance. Skipping a few disk requests does not proportionally translate into degradation of quality. This is especially important in order to cater to the information-access needs of a large number of users.

Recent flourish of the Internet has caused a huge increase in the amount of content accessed from other computer systems (content providing servers). In order to meet their service requirements of local and remote applications efficiently, the operating system needs to be aware of many different kinds of service patterns that the applications might have. For example, an FTP application requires that data get transferred as fast as possible. So throughput would be used to measure the performance of such an application. Interactive applications require a portion of the CPU or fetch data every now and then but do not necessarily complete their task quickly. Continuous media applications, on the other hand, require a guaranteed rate of delivery of data from the disk when playing back files. Service patterns for these applications are different and the operating system needs to have this knowledge incorporated in it to efficiently serve requests from these applications.

Another bottleneck that can be experienced in multimedia applications is the network delay (propagation, serialization, switch, and quantization). The technological development for hard disk seems to be slower than networking components. The invention of fiber optic networks provides higher transfer rate (terabit per second instead of megabit per second). Although the latency of network can not be overcome fully, recent technologies provide higher transmission rate through the network, but it can not be ignored totally while we consider multimedia transmission over the network. In this thesis, we considered I/O bandwidth of the disk only for the admission control algorithm. Consideration of the network bandwidth is out of scope of this thesis.

One of the most ubiquitous components of a computer system is the hard disk, which is used for storage and retrieval of programs and data (both application and system dependent) etc, as well as for essential operating system functions like virtual memory management and more. Therefore, one of major factors impacting application performance is how fast data can be stored and retrieved from the hard disk. More importantly, disk access is orders of magnitude slower than memory access, and it is a bottleneck to overcome in order to provide good application performance by retrieving data quickly and efficiently. In order to speed up disk accesses efficient algorithms sequence disk requests in a way that would minimize time spent in retrieving data. The gap in the operating speeds of hard disks and memory has only widened with increase in computing power and faster memory access, making the problem of optimizing disk access even more critical. Compounding this problem is the fact that the applications developed for present day systems have significantly different service requirements [1].

Since a media stream consists of a sequence of media quanta, such as video frames or audio samples, which convey meaning only when presented continuously in time, a multimedia server must ensure that recording and retrieval of media stream to and from disks proceed at their real-time rate. Whereas designing a dedicated, single-client multimedia server does not offer many design choices and is relatively straightforward, the design of a multimedia sever that is capable of serving multiple clients simultaneously poses interesting research challenges. This is because, given the maximum rate of data transfer from disks, a multimedia server can only serve a limited number of clients. Hence, before admitting a new client, a multimedia server must employ admission control algorithms to decide whether a new client can be admitted without violating the continuity requirements of any of the clients already being served. Analysis and improvement of such algorithms is the subject matter of this thesis.

1.2 Background and Present State of the Problem

Recent advances in computing and communication technologies have made it feasible as well as economically viable to provide on-line access to a wide variety of information sources such as reference books, journals, newspapers, images, video clips, scientific data, etc, over high speed networks. The realization of such information management systems of the future, however, will require the development of high performance, scalable multimedia servers which can provide a wide range of services to a large number of clients [4]. The fundamental problem in developing such multimedia servers is that images, audio, video and other similar forms of data differ from numeric data and text in their characteristics, and hence they require totally different techniques for their organization and management.

A large-scale multimedia server has to serve a large number of clients simultaneously. Given the real-time requirements of each client, a multimedia server must employ admission control algorithms to decide whether a new client can be admitted for service without violating the requirements of the clients already being served. Two types of new services have been proposed in the literature to support real-time multimedia applications: guaranteed service [5] and predicted service [6]. In a guaranteed service model, client-specified a priori performance bounds are guaranteed to each connection regardless of the behaviors of other connections. In a predicted service model, a network dictated post facto delay bound and the service may be disrupted due to the network load fluctuation.

In [7], two types of guaranteed services are proposed: deterministic service and statistical service. In deterministic service, performance bounds are guaranteed for all packets on a connection even in the worst case. In statistical service, probabilistic performance bounds are guaranteed [8]. Although the quality of deterministic service is better, statistical service allows the network to achieve a higher utilization by exploiting statistical multiplexing. In the predictive service model the admission control criterion is defined using the measured characteristics of the current load on the server, rather than theoretical worst-case bounds. Most of the existing work on admission control algorithms for multimedia servers has been focused on developing techniques for providing deterministic service guarantees to clients (i.e., playback requirements are strictly met for the entire service duration) [9][10][11][12][13][14]. However, since human perception is tolerant to brief distortions in audio and video, providing deterministic guarantees to each client is superfluous. Furthermore, the worst-case assumptions that characterize most of these techniques may needlessly constrain the number of clients that are served simultaneously, and hence, may lead to severe under-utilization of server resources. This is because, the average time spent in accessing a media block from disk, in practice, and is significantly smaller than the corresponding worst-case times. Hence, in order to improve the utilization of server resources, a multimedia server must employ an admission control algorithm, which exploits the statistical variation in the access times of media blocks from disk.

To meet this purpose a new hybrid admission control algorithm is proposed in [15]. Here the server utilization is increased via dividing the clients into separate groups with different service requirements. This gives a relatively higher performance and utilization than Deterministic and Statistic Approach. This thesis proposes improvement and tries to find the validity of the findings of [15] in various conditions.

1.3 Scope and Focus of the Thesis

Typically, disk-scheduling mechanisms for multimedia applications reduce disk access times by only trying to minimize movement to subsequent blocks after sequencing based on Earliest Deadline First. We compared the results of using various well-known disk-scheduling algorithms as well as a relatively new algorithm on the performance of hybrid admission control algorithm. We find that our approach results in satisfactorily improved performance for multimedia and non-multimedia applications. The main focus of this thesis is to study the hybrid admission control algorithm that will increase number of clients for multimedia server as well as provide guaranteed service for the all clients who have made request for guaranteed service and accepted in to the server. We also propose improvement of performance of the algorithm by using multiple storage devices in the server with separate file systems for each of the storage devices. The thesis will focus mainly on the following objectives:

- Study the hybrid admission control algorithm using the previously mentioned hybrid approach that will decide whether a new client can be admitted for service without violating the requirements of the clients already being served in the server.
- Propose an improvement to the hybrid admission control algorithm where disk retrieval time is improved by introducing use of multiple storage devices with separate file systems.
- Analysis of the hybrid admission control algorithm and the multi-storage improvement of it.
- Comparison of performance with purely deterministic and purely statistical admission control algorithms.
- Study the decision making time and processing time of the algorithm and effect of using different disk scheduling algorithms on the performance of the hybrid admission control algorithm.

1.4 Organization of the Thesis

The thesis has been organized in different chapters, with each chapter discussing different aspects of the study. The areas covered by different chapters are briefly as follows:

Chapter 2:	Provides a literature review of relative works on different strategies that have been proposed in the context of admission control algorithm for multimedia server and necessary background information.
Chapter 3:	Describes the main aspects of the basic Hybrid Admission Control Algorithm.
Chapter 4:	Discussion and analysis of the multiple storage extension of the basic algorithm described in the previous chapter.
Chapter 5:	Simulation outcomes and analysis of the simulation results together with comparison between different admission control algorithms and effect of using different disk scheduling algorithm with hybrid admission control algorithm.
Chapter 6:	Concluding remarks and suggestions for future research work.

Chapter Two

Preliminaries

2.1 Introduction

In this chapter we will review the basics of multimedia, multimedia servers together with different perennial approaches to handle the admission control problem to multimedia servers. We will also take a brief look at the different disk scheduling algorithms used in our simulation study.

2.2 Multimedia

Multimedia means, from the user’s perspective, that computer information can be represented through audio and/or video, in addition to text, image, graphics and animation. The integration of these media into the computer provides additional possibilities for the use of computational power currently available. Furthermore, these data can be transmitted through computer and telecommunication networks, which imply applications in the areas of information distribution and cooperative work.

A multimedia system is characterized by computer-controlled, integrated production, manipulation, presentation, storage and communication of independent information, which is encoded at least through a continuous (time-dependent) and a discrete (time-independent) media.

From the networking perspective, all media types can be classified as either Real-time (RT) or Non-real time (NRT) depending on their end-to-end delay requirements. For example, text and image files do not have any delay constraints and hence classified as NRT media types. The RT media types are further classified as Discrete Media (DM) or Continuous Media (CM), depending on whether the RT data is transmitted in discrete quantum (as a file or message) or continuously (as a stream of messages with inter-message dependency). The real time discrete type of media has recently gained high popularity because of ubiquitous applications like MSN/Yahoo messengers (which are error-intolerant) and instant messaging services like stock quotes (which are error tolerant). The RT continuous type of media can further be classified as delay tolerant or delay intolerant. The term ‘delay intolerant’ only signifies that such media type can tolerate higher amounts of delay than the delay tolerant types, without significant performance degradation. Examples of RT, continuous media delay intolerant media are audio/video media used in audio/video conferencing systems, and remote desktop applications. Streaming audio/video media used in applications like Internet web cast are also the examples of delay-intolerant media types.

2.3 Multimedia Servers

Recently there has been an interesting growth in the demand for distributed multimedia applications operating over the Internet. Such applications have shown their value as a powerful technology that can allow people to remotely share resources or work collaboratively, thus saving time and money. Typical applications of distributed multimedia systems include video conferencing, video telephony, collaborative work, multimedia mail, and distance learning. Some recent applications include on-demand multimedia services, such as in entertainment, video news distribution services, and distribution of video rental services, interactive television, and digital multimedia libraries.

These applications demand certain constraints or service guarantees from the communication network, which must be satisfied to deliver an acceptable performance. The performance guarantee that a multimedia network can provide its applications is often referred to as *Quality of Service* (QoS). This may include guarantees on the throughput, network delays, delay jitter, and error rate. The current best-effort Internet, however, has been found to be inadequate to satisfy such requirements and enhancements are required to this basic Internet model to overcome this shortcoming.

Multimedia servers are content containing servers that ensure the necessary real-time QoS as described previously. It can be part of *Content Delivery Network* (CDN) that holds copies of similar data placed in different places all over the world. However, the volume of data that is typically stored on a multimedia server usually prohibits this form of complete server replication. In addition, since clients rarely request the majority of multimedia files, replication of low-demand files is wasteful. Multimedia servers try to improvise the shortcomings regarding service requirements by efficiently serving the incoming clients.

2.4 Admission Control and its Different Approaches

A multimedia server has to serve a large number of clients simultaneously. Given the real-time requirements of each client and fixed data transfer bandwidth of disks, a multimedia server must employ admission control algorithm to decide whether a new client can be admitted for service without violating the requirements of the clients already being served. Admission control is a mechanism that multimedia servers use to restrict service to limited number of clients while either having a mechanism to allow re-negotiation with session requesting clients or deny service to the other clients till such time that there is enough bandwidth available to serve them.

Multimedia clients typically negotiate using what are called Quality of Service (QoS) parameters to obtain a certain amount of service in terms of periodicity of data delivery. These QoS parameters are commonly expressed as bit rate, block rate and frame rate [16]. The server uses these performance parameters supplied by the clients for admission control and subsequent service. The server processes a client request based on QoS parameters and decides whether or not the performance guarantees for the client request can be met. This is essential to ensure acceptable deterioration in performance perceived by clients already being served [6][8][17][18]. There are three major approaches to carrying out admission control.

2.4.1 Deterministic Approach

The first approach is to provide deterministic guarantees to the clients [9], the strictest form of admission control, since it uses worst-case values for retrieving media blocks from the disk. The advantage of this approach is that all admitted clients receive all the blocks and no service agreements are violated. The obvious disadvantage is that it is overkill for soft real-time applications like continuous media applications.

2.4.2 Probabilistic Approach

The second approach is stochastic in nature [8]. This only provides a statistical guarantee that the deadlines for all the admitted clients will be met. This means that at least a fixed percentage of the blocks are retrieved for each client but not necessarily all of them. One of the biggest advantages of this approach is that it results in an increase in the number of clients that are admitted compared to the deterministic approach, since it is not important that all the blocks are retrieved for all the clients. The consequent problem with this approach is that applications that have very strict deadline and loss requirements cannot be accommodated. In the event that deadlines are violated, there must be a mechanism to determine the blocks that can be dropped and to distribute the violation of service guarantees among as many of the admitted clients as possible.

2.4.3 Observation Approach

The third approach is one based on observation [17]. In this approach, the times taken for retrieving various media blocks are recorded. When there is a request for admission, an extrapolation is made from current values for access times to obtain the time taken for the new client. This estimated time is used to either accept or decline a client’s request for admission. Although this does not provide very strict service guarantees like the deterministic approach, it still provides a fair amount of improvement over the deterministic approach. As we are proposing hybrid admission control algorithm, in the following sections we will describe different admission control algorithms in detail.

2.5 Statistical Admission Control Algorithm

Consider a multimedia server that is servicing n clients, each retrieving a different media strand (say, $S_1, S_2, \ldots, S_n$, respectively). Let service requirements of Client i be specified as percentage p_i of the total number of frames that must be retrieved on time. A multimedia server can serve these clients by proceeding in periodic rounds, retrieving a fixed number of frames for each client during each round. Let $f_1, f_2, \ldots f_n$ denote the number of frames of strands $S_1, S_2, \ldots, S_n$, respectively during each round.

Then assuming that R_{pl}^{i} denotes the playback rate (expressed in terms of frames/sec) of Strand S_i, the duration of a round, defined as the minimum of the playback durations of the frames accessed during a round, is given by,

$$R = \min_{i \in [1,n]} \left(\frac{f_i}{R_{pl}^{i}} \right) \qquad (2.1)$$

In such a scenario, ensuring continuous playback of each media strand requires that the total time spent in retrieving media blocks from the disk during each round (referred as service time τ) should not exceed R . That is:

$$\tau \le R \qquad (2.2)$$

The service time, however, is dependent on the number of media blocks being accessed as well as their relative placement on the disk. Since each media strand may be encoded using variable bit rate compression technique, the number of media blocks that contain f_i frames of S_i strands may vary from one round to another. This difference when coupled with the variation in the relative separation between blocks yields different service times across rounds. In fact, while serving a large number of clients, the service time may occasionally exceed the round duration (i.e. $\tau > R$). We refer to such rounds as overflow rounds. Given that each client may have requested a different quality of service (i.e., different values of p_i), meeting all of their service requirements will require the server to delay the retrieval of or discard media blocks of some of the more tolerant clients during overflow rounds. Consequently, to ensure that the statistical quality of service requirements of clients are not violated, multimedia server must employ admission control algorithms that restrict the occurrence of such overflow rounds by limiting the number of clients admitted for service.

To precisely derive an admission control criterion that meets the above requirement, observe that for rounds in which $\tau \le R$, none of the media blocks need to be discarded. Therefore, the total number of frames retrieved during such rounds is given by $\sum_{i=1}^{n} f_i$. During overflow rounds, however, since a few media blocks may have to be discarded or delayed to yield $\tau \le R$, the total number of frames retrieve will be smaller than $\sum_{i=1}^{n} f_i$. Given p_i, denotes the percentage of frames of strand S_i that must be retrieve on time to satisfy the service requirements of Client i, the average number of frames that must be retrieve during each round is given by $p_i * f_i$. Hence assuming q denotes the overflow probability (i.e., $P(\tau > R) = q$), the service requirements of the clients will be satisfied if:

$$q * F_0 + (1-q)\sum_{i=1}^{n} f_i \geq \sum_{i=1}^{n} p_i * f_i \qquad (2.3)$$

Here F_0 denotes the number of frames that are guaranteed to be retrieved during a overflow round. The left hand side of the equation 2.3 represents the lower bound on the expected number of frames retrieved during a round and the right hand side denotes the average number of frames that must be accessed during each round so as to meet the service requirements of all clients. Clearly, the effectiveness of this admission control criteria, measured in terms of the number of clients that can be admitted, is dependent on the values of q and F_0 [8].

Statistical admission control algorithm improves the utilization of server resources by exploiting the variation in the access times of media blocks from the disk. The main goals of this admission control algorithm are as follows:

- Allowing enough traffic to efficiently utilize server resources, while not accepting clients whose admission may lead to the violations of the service requirements of the clients.
- Providing statistical service guarantees to each client.

2.6 Observation Based Admission Control Algorithm

In observation based admission control algorithm a client is admitted for service only if the predicted extrapolation from the status quo measurements of the storage server utilization indicates that the service requirements of all the clients can be met satisfactorily. An observation-based admission control algorithm is based on the assumption that the amount of time spent in serving each of the clients already being served will continue to exhibit the same behavior, even after a new client is added into the system. Therefore, a new client will be admitted for service only if the prediction from the status quo measurements of the server performance characteristics indicates that the service requirements of all the clients can be met satisfactorily. A multimedia server that employs such an observation-based approach is referred to as providing predictive service guarantees to clients (A similar technique was presented by Clark et al. [6][19]for optimizing the utilization of network resource).

Notice that the observation-based admission control algorithm offers fairly reliable service, but no absolute guarantees. The admission control decisions are based on the measured characteristics of the current load on the server, rather than theoretical worst-case behavior. Hence, the key function of the admission control algorithm is to accept enough traffic to efficiently utilize the server resources, while not accepting clients whose admission may lead to the violation of the service requirements.

Finally, since the observation-based admission control algorithm provides fairly reliable service but no absolute guarantees, simultaneous serving of multiple clients may lead to occasional violation of the continuity requirements (i.e., media unit losses) of some of the clients. In order to enable a multimedia server to meet the requirements of as many clients as possible, observation based admission control algorithm proposed a technique for minimizing as well as distributing the media unit losses among multiple clients [17].

2.7 Literature Review of Disk Scheduling Algorithms

The main goals of disk scheduling algorithms are to achieve a high throughput and to provide fair disk access for every process. The additional real-time requirements introduced by multimedia systems make traditional disk scheduling algorithms inconvenient for multimedia systems. In the case of contiguous storage, scheduling is only needed to serve requests from multiple streams concurrently. In [20], a round-robin scheduler is employed that is able to serve hard real-time tasks. Here, additional optimization is provided through the close physical placement of streams that are likely to be accessed together. The time to read or write a disk block is determined by three factors: the seek time, the rotational delay, and the actual transfer time. For most disks, the seek time dominates, so reducing the mean seek time can improve system performance substantially. In this section we discuss some disk scheduling algorithms that either take rotational latency into account innately or modified to include that along with seek time.

2.7.1 First Come First Serve (FCFS)

If the disk driver accepts requests one at a time and carries them put in that order, that is, First-Come, First-Served (FCFS)[21], little can be done to optimize seek time. It works in constant time and in most of the cases it results in poor response time for the lack of any sophisticated optimization. We have extended it to include the rotational latency in decision making.

2.7.2 Shortest Seek Time First (SSTF)

Shortest seek time first [21] always handles the closest request next, to minimize seek time. The obvious problem is that in heavily loaded case it is not fair to requests located in further places from the head. The goals of minimum response time and fairness are in conflict here.

2.7.3 Scan/Elevator Algorithm (SCAN)

Most elevators use an algorithm to reconcile the conflicting goals of efficiency and fairness. They keep moving in the same direction until there are no more outstanding requests in that direction, and then they switch directions. This algorithm [22], known both in the disk world and the elevator world, as the elevator algorithm requires the software to maintain 1 bit: the current direction bit, UP or DOWN. When a request finishes, the disk or elevator driver checks the bit. If it is UP, the arm is moved to the next highest pending request, if any. If no requests are pending at higher positions, the direction bit is reversed. When the bit is set DOWN, the move is to the next lowest requested position, if any.

2.7.4 Near Optimal Disk Scheduling Approach (NODSA)

Finally, another disk scheduling algorithm is considered named NODSA [23] or Near Optimal Disk Scheduling algorithm that considers a request as a vertex $z_i = (x_{i,}, y_i)$ in a planar graph where x_i and y_i denote the track number and the block number on that track, respectively. So the problem of finding an optimal sequence of block-requests reduces to the traveling-salesman problem, a classical graph theory problem known to be NP-complete [24]. Then an approximate solution using Prim's minimum spanning tree algorithm together with triangle inequality principle is used to find an approximately optimal sequence of accessing disk-block requests with a ratio bound of 2 [24].

2.7.5 Analysis

The computational complexity of different disk scheduling algorithms is analyzed in this section.

- First Come First Server (FCFS) does its operation in constant time, as there is nothing to be done for preprocessing.
- The computational time of both Shortest Seek Time First (SSTF) and SCAN is $O(n \lg n)$, where n denotes the number of blocks in the queue to be retrieved.
- But the computational complexity of near optimal disk scheduling algorithm is $\theta(n^2)$, as this method uses a complete graph and construct minimum spanning tree by using MST_PRIM algorithm[24].

2.8 Summary

The objective of admission control algorithm is to control the usage and allocation of disk resources for various applications requiring service. Admission control is a key component in multimedia servers, which need to allow the resources to be used by the clients only when they are available. This assumes great significance when the server needs to maintain a certain promised level of service for all the clients being served. If the admission control admits too few clients, it results in wastage of system resources. On the other hand, if too many clients are allowed to contend for resources, then the performance of clients already degrades rapidly in the presence of new clients. Therefore, judicious decision making mechanisms for allocating resources disk bandwidth to clients are needed.

Chapter Three

Hybrid Admission Control Algorithm

3.1 Introduction

A continuously recorded sequence of audio samples or video frames is called a strand. A multimedia server must organize the storage of such media strands on disk (in terms of media blocks). During each round, the multimedia server retrieves a sequence of media blocks for each strand. The number of blocks of each media strand retrieved during a round is dependent on its playback rate requirement, as well as the available buffer space at the client. Consequently, ensuring continuous retrieval of each strand requires that the service time (i.e., the total time spent in retrieving media blocks during a round) does not exceed the minimum of the playback durations of the sequences of blocks for each strand retrieved during the round. Since service time is a function of the number of blocks retrieved during a round, a server can serve only a limited number of clients simultaneously. Hence, before admitting a new client, a multimedia server must employ admission control algorithms to decide whether a new client can be admitted without violating the continuity requirements of any of the clients already being served [8]. Hybrid admission control Algorithm by Tanvir [15] is such an algorithm that combines both deterministic and statistical approaches. In this chapter we will look into the basics of HACA.

3.2 Formulation of Admission Control Problem

Let a multimedia server is serving n clients, each of which is retrieving a different media strand (let, $S_1, S_2, \ldots, S_n$, respectively). Let $R_1, R_2, \ldots, R_n$, denote the playback rates (in terms of bytes/sec) of strands $S_1, S_2, \ldots, S_n$, respectively.

Furthermore, let $k_1, k_2, \ldots, k_n$, denote the number of blocks of strands $S_1, S_2, \ldots, S_n$, retrieved during each round. The total service time in serving n requests during a round is dependent on the total seek time and rotational latencies incurred while accessing $(k_1 + k_2 + \ldots + k_n)$ media blocks from the disk. In the worst case, the disk head may have to be repositioned onto new track at most $(k_1 + k_2 + \ldots + k_n)$ times during each round.

Suppose,

T = Number of tracks/cylinders on disk

M = Disk block size, in bytes

a, b = Seek time parameters, they are constants

$l_{seek}(t_1, t_2) =$ Seek time (a + b * |t$_1$ – t$_2$|), in sec

$l_{rot}^{\max}$ = Maximum rotational latency, in sec

R = Minimum playback rate

τ = Service time

It is quite clear in Figure 3.1 how to determine the minimum duration of a round. Say, multimedia server is serving *n* clients. Server provides a number frames to each client and client consumes these frames. In order to ensure continuous playback condition server must provide a new set of frames before the client consumes earlier set of frames.

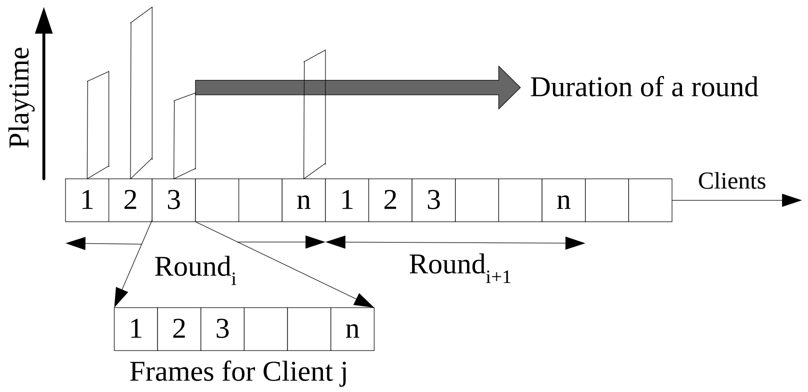

Figure 3.1: *Duration of a Round*

Duration of a round is the minimum time to consume a set of frames among the all clients. To ensure continuous playback of a media stream for each client, multimedia server must serve new set of frames before the minimum duration of a round. The equation $\tau \le R$ ensures that the playback requirements of all the clients are strictly met for the entire service duration.

So, total seek time incurred during a round is $a * \sum_{i=1}^{n} k_i + b * T$. Again, maximum rotational latency incurred during a round is $l_{rot}^{\max} * \sum_{i=1}^{n} k_i$. Hence total service time for each round is bounded by:

$$\begin{aligned} \tau &= a * \sum_{i=1}^{n} k_i + b * T + l_{rot}^{\max} * \sum_{i=1}^{n} k_i \\ &= (a + l_{rot}^{\max}) * \sum_{i=1}^{n} k_i + b * T \qquad (3.1) \end{aligned}$$

Consequently, ensuring continuous retrieval of each strand requires that the total service time per round do not exceed the minimum of the playback durations of $k_1, k_2, \ldots, k_n$, blocks. So the admission control criteria can be formally stated as:

$$\tau = (a + l_{rot}^{\max}) * \sum_{i=1}^{n} k_i + b * T \le \min_{i \in [1,n]} \left(\frac{k_i * M}{R_i} \right) \qquad (3.2)$$

Equation (3.2) indicates that service time should be less than or equal to the minimum duration of a round and ensures that the playback requirements of all the clients are strictly met for the entire service duration. Hence, a multimedia server that employs such an admission control criteria is said to provide *deterministic service guarantees* to each client.

The worst case assumptions of the deterministic techniques is that it may needlessly constrain the number of clients that are served simultaneously and thus leads to severe under-utilization of the server resources. This is because, the average time spent in accessing a media block from disk, in practice, and is significantly smaller than the corresponding worst-case times. So we must exploit the statistical variation in access times. Hybrid Admission Control Algorithm does just that.

3.3 Hybrid Admission Control Algorithm

Hybrid Admission Control Algorithm (HACA) is an Admission Control Algorithm that effectively incorporates the approaches of both Deterministic and Statistical Admission Control Algorithms. Here, these two types of admission policies are used to serve two different categories of clients and thus the performance of the multimedia server is improved.

3.3.1 The Formulation of the Algorithm

Consider a multimedia server that is serving n clients, each retrieving a media strand (say $S_1, S_2, \ldots, S_n$, respectively). Let n_s and n_n denote the number of clients that require deterministic and non-deterministic service, we call them super user (SU) and normal user (NU) respectively (i.e., $n = n_s + n_n$). Without loss of generality, let us assume that clients retrieving strands $S_1, S_2, \ldots, S_{n_d}$ require deterministic service guarantees, and those retrieving $S_{n_d+1}, S_{n_d+2}, \ldots S_n$ are tolerant to brief distortions or loss of information.

Let, minimum duration of a round, $R = \min_{i \in [1,n]} \left(\frac{k_i * M}{R_i} \right)$ sec. General rule of admission control is $\tau \leq R.$. We are defining a new term *safe guard*; it is the reduced percentage of minimum duration of a round. Say, if the admission control technique uses 90% of R in the admission control equation we called it 10% safe guard. On the other hand, the total time spent in retrieving the media blocks from disk during each round (referred to as service time τ) is dependent on their relative placement on disk as well as the disk-scheduling algorithm. Since the relative placement of blocks can vary from one round to another, the service times may also vary across rounds. Consequently, in some rounds, τ could be greater than R. Such rounds are called *overflow rounds*. Since maintaining continuity of playback for intolerant clients requires the service time to be smaller than the duration of a round, blocks of some of the tolerant clients may have to be discarded (i.e., not retrieved) during such overflow rounds.

Let, K denotes number of media blocks accessed during a round. So, $K \leq \sum_{i=1}^{n} k_i$. Hence, average time of retrieving a media block is:

$$\eta = \frac{\tau}{K} \qquad (3.3)$$

The admission control algorithm that we are presenting is based on the assumption that the average amount of time spent for the retrieval of each media block (i.e., the value of η) does not change significantly even after the server admits a new client. In fact, to enable the multimedia server to accurately predict the amount of time expected to be spent while retrieving media blocks during a future round, we will maintain a history of the values of η observed during the most recent W rounds. If η_{avg} and σ, respectively, denote the average and the standard deviation of η over the last W rounds, then the time required to retrieve a block in future rounds can be estimated as:

$$\hat{\eta} = \eta_{avg} + \in * \sigma \tag{3.4}$$

Here $\in$ is a constant. It plays an important role in controlling the number of overflow rounds. And HACA uses $\hat{\eta}$ to take decisions about future admissions.

In HACA, we form two groups according to the service requirements. The clients who demand guaranteed service fall in super user (SU) group and other clients are in normal user (NU) group. We apply different admission policies based on the service requirements. Say,

$\tau_s(n)$ = Service Time of SU group of n members

$\tau_n(n)$ = Service Time of NU group of n members

n_s = Number of super users

n_n = Number of normal users

p_i = % of blocks to be retrieved on time for Client i

So, Service time of super users,

$$\tau_s(n_s) = (a + l_{rot}^{\max}) \times \sum_{i=1}^{n_s} k_i + b \times T \tag{3.5}$$

Service time of normal users,

$$\tau_n(n_n) = \hat{\eta} \times \sum_{i=1}^{n_n} k_i \times p_i \tag{3.6}$$

Also, $\tau = \tau_s(n_s) + \tau_n(n_n)$ and $n = n_s + n_n$. Hence, at every instant following condition must be true,

$$\tau_s(n_s) + \tau_n(n_n) \le \min_{i \in [1,n]} \left(\frac{k_i \times M}{R_i} \right) \tag{3.7}$$

3.3.2 Description of the Algorithm

In the previous subsection we have formulated the criterion for the working of the Hybrid Admission Control Algorithm. We have shown that there are two types of user groups, namely super user group and normal user group, each of which has separate service real-time requirements.

When a new client requests for accessing the resources of a multimedia server, according to the group we may need either to calculate the new service time τ_s or to estimate the new service time τ_n. If the summation of these two service times is less than or equal to the minimum duration of a round, request is accepted otherwise it is declined. The exact formulation is presented in the following part of this chapter. A flow chart of the Hybrid Admission Control Algorithm is given in Figure 3.2. In the subsequent sections (sections 3.4 & 3.5) we will discuss the specific conditions for admission into the server and measures that are taken to ensure service quality respectively.

The major steps in taking the decisions are briefly described below.

Receive client request: At first, a client request is received into the multimedia server. In this step, client's user group is identified which will be used later in the process of finding whether or not to accept the client.

Take decision about admission: In this step, decision about accepting or discarding the request is taken on the basis of the client's user group, which decided in the previous step. This process and the specific conditions are described the following section (Section 3.4).

Select disk blocks to be retrieved in this round: In this step the disk blocks that are to be sent to the clients are selected, discarding those which will result in failure of service guarantees. The policies to discard are discussed in Section 3.5.

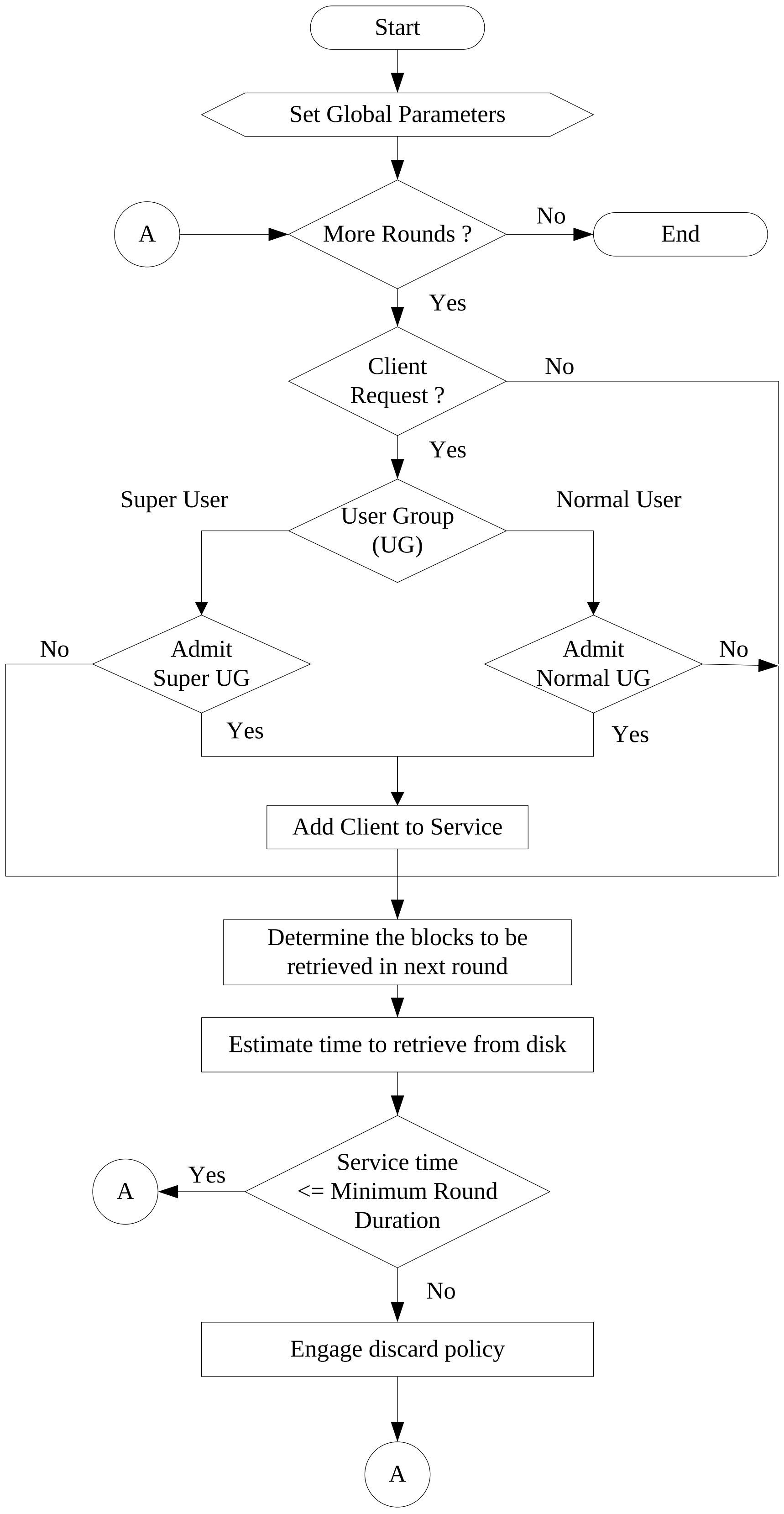

Figure 3.2: *Flow chart of Hybrid Admission Control Algorithm*

3.4 Admitting a New Client

When a new client requests to access the resources of a multimedia server, according to its group we may need either to calculate the new service time of super users $\tau_s(n_s+1)$ or to estimate the new service time of normal clients $\tau_n(n_n+1)$. If the summation of these two service times is less than or equal to the minimum duration of a round, request is accepted otherwise it is declined.

In order to precisely formulate the admission control criteria, consider a scenario in which a multimedia server receives a new client request for the retrieval of strand S_{n+1}. Let, R_{n+1} denotes the playback rate requirement for the new client, and k_{n+1} denotes the number of blocks of strand S_{n+1} that need to be retrieved during each round.

3.4.1 Deterministic Criteria

If the new client desires deterministic service, then before admitting the client, the multimedia server must ensure that neither super users nor normal users are being suffered after the new client is admitted. Thus admission control criterion in deterministic case can be found from Equation (3.7),

$$\tau_s(n_s+1)+\tau_n(n_n) \le \min_{i\in[1,n+1]}\left(\frac{k_i \times M}{R_i}\right) \qquad (3.8)$$

3.4.2 Non Deterministic Criteria

On the other hand, if the new client belongs to normal user group then admission control criterion be,

$$\tau_s(n_s)+\tau_n(n_n+1) \le \min_{i\in[1,n+1]}\left(\frac{k_i \times M}{R_i}\right) \qquad (3.9)$$

The performance of such an approach is dependent on the values of η_{avg} and σ; smaller the values of η_{avg} and σ, greater is the number of clients that can be served simultaneously by the server. Hence, the multimedia server must employ disk-scheduling algorithms that minimize the total time spent in retrieving media blocks during each round. Also, multimedia server has to employ policies for determining the minimum number of media blocks, which when discarded will yield service time less than or equal to the duration of a round (i.e. $\tau \leq R$).

3.5 Ensuring Service Guarantees

In this section we describe some policies to ensure service guarantee for the multimedia clients that are used in HACA. This techniques of ensuring service guarantees was first used by A. Goyal [8][19].

3.5.1 Overflow Rounds

Recall that the admission control algorithm presented in Section 4.4 admits a new client if the client is able to meet the admission control criteria in which it belongs. Due to the aggressive nature of this admission control criteria in case of normal user, the total time spent in retrieving media blocks during a round (i.e., service time τ) may occasionally exceed the duration of a round *R*, yielding an overflow.

3.5.2 Policy of Discarding Blocks

Observe, however, that since the retrieval sequence for Round i is pre-computed during Round $(i-1)$ an overflow can be detected before actually initiating Round i. Hence, in order to ensure that the deterministic service guarantees provided to clients are not violated, a multimedia server must discard (i.e., not retrieve) sufficient number of media blocks of normal clients so as to maintain the service time within the duration of the round (i.e., $\tau \leq R$). However, in doing so, the multimedia server must minimize the number of blocks discarded, as well as distribute the set of discarded blocks among the normal clients so as not to violate any of their requirements.

3.6 Analysis of the Algorithm

In Hybrid Admission Control Algorithm there are two types of user. We classify them into two groups; super user group and normal user group. Each group has an average rate of inter-arrival times. In simulation, client requests were generated using Poisson process.

- After classifying client request, multimedia server tests this request to admit it for accessing server resources by using hybrid admission control algorithm. The computational time to test admission condition, i.e. whether the request is accepted or declined, is linear. So computational complexity of hybrid admission control algorithm is $O(n)$, where n, is the number of clients that are currently accessing server resources.

- Next step of the multimedia server is to determine the blocks that are needed to retrieve in the next round. After determining the blocks, disk-scheduling algorithm is responsible to generate the order of blocks to be retrieved. From our experiments we found that SSTF gives the best performance with $O(n \lg n)$ complexity, where n is the number of blocks in the queue.

- The computation complexity of deterministic admission control algorithm is constant. But block retrieval time depends on the disk-scheduling algorithm it uses. In contrast, the computation complexity of statistical admission control algorithm is $O(n)$, where, n is the number of clients that are currently accessing server resources.

So the complexity of HACA is dependent on the disk scheduling algorithm employed, which is SSTF with rotational latency taken into account in this case.

3.7 Summary

The problem with Deterministic Admission Control Algorithm (DACA) is that it results in under-utilization of server resources, whereas Statistical Admission Control Algorithm (SACA) suffers from the problem of accepting more clients than the server can afford to handle at a time. HACA effectively solves these problems by incorporating the ideas of both algorithms and eventually resulting in a much more efficient technique to handle the problem of admission control. Though its performance and complexity is dominated by the disk scheduling algorithm used, we can always find one that will suit it most and give the maximum utilization.

Chapter Four

Hybrid Admission Control Algorithm with Multiple Storage

4.1 Introduction

Hybrid Admission Control Algorithm obviates the occurrence of severe under-utilization of the server resources and ensures that real-time requirements of clients from both super and normal user groups are met by accepting clients whose requests, if served, will not disrupt the service guarantees to the pre-existing users. It also uses its sophisticated discard policy that is optimized to ensure that maximum clients can be served in case an overflow round pops up. But from the analysis of the complexity of HACA, presented in the previous chapter we see that its performance is mainly dominated by the disk access time to retrieve requested disk blocks and the performance of the disk scheduling algorithm in the server. So it is obvious that if we can decrease disk access times we can considerably improve the performance of this admission control algorithm. In this chapter, we discuss the effect of using multiple storage devices in the multimedia server in order to introduce parallelism in disk access and thus reduce the service time resulting in reduced number of overflow rounds. We have also experimentally found that this extension effectively increases the number of clients that is admitted using the Hybrid Admission Control Algorithm.

4.2 Multi-Storage File System

With the use of multiple storage devices, we can effectively decrease the time to access a fixed number of disk blocks than the time it needs when there is only single storage device to serve all the requested blocks. This type of parallel access file system has already been implemented in GPFS (General Parallel File System)[25], which imitates the general POSIX file system of a single computer on cluster computers. GPFS evolved from the Tiger Shark multimedia file system [26]. GPFS successfully satisfies the needs for throughput, storage capacity, and reliability of the largest and most demanding problems. From that experience we have been intrigued to implement similar sort of file system and use it together with HACA to look for a considerable amount of improvement in performance of HACA.

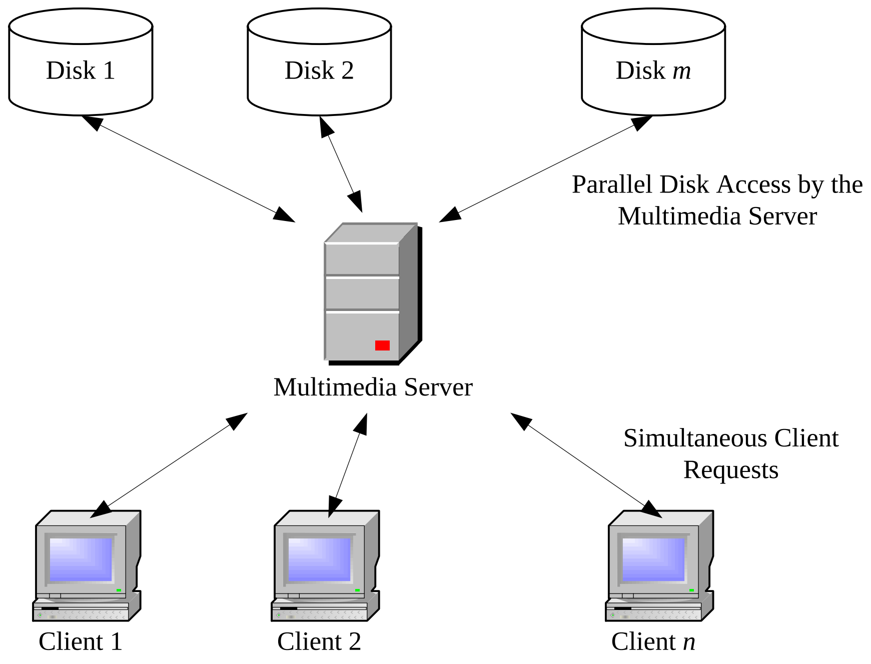
Disk 1
Disk 2
Disk *m*
Parallel Disk Access by the Multimedia Server
Multimedia Server
Simultaneous Client Requests
Client 1
Client 2
Client *n*

Figure 4.1: *Multimedia Server with Multi-Storage*

With multiple devices in hand, we have two options to distribute the audio-visual files saved in the server among the separate storage devices. We can split a single file into pieces and place its chunks into separate disks. Again on the contrary, we can keep files intact and distribute separate files among different disks. In case of multimedia server, the distribution of files is more effective than splitting files as there is no need to randomly access the files and supplying in sequence is far more important than supplying out of sequence, even if very quick. So the basic idea of the experimented file system is to use equal storage area divided into multiple storage devices and place intact files sequentially in separate disk blocks. As a result, when requests from different clients arrive, the multimedia server can retrieve necessary disk blocks for the requested data by all the clients simultaneously from different storage devices and decrease the total service time (Figure 4.1). Consequently, the number of overflow rounds decreases significantly.

4.3 Multi-Storage Hybrid Admission Control Algorithm

Now that we have a file system capable of working with multiple storage devices, we focus on the necessary changes in the standard Hybrid Admission Control Algorithm to accommodate and exploit the updated condition.

4.3.1 The Updated Algorithm

Consider the multimedia server described in the previous Chapter with an exception that it has *m* disks (say $D_1, D_2, ..., D_m$ respectively) instead of a single one to serve the incoming client requests in a single round. Let the multimedia server is serving n clients, each of which is retrieving a different media strand (say $S_1, S_2, \ldots, S_n$ respectively). Now all these *n* requests are divided into *m* disks that are present in the multimedia server and each of these *m* disks will be accessed simultaneously resulting in parallel disk access.

Furthermore, let $k_1, k_2, \ldots, k_n$, denote the number of blocks of strands $S_1, S_2, \ldots, S_n$, retrieved during each round. In single disk case, total service time in serving n requests during a round is dependent on the total seek time and rotational latencies incurred while accessing $(k_1 + k_2 + \ldots + k_n)$ media blocks from the disk. But in the current condition, the total time will not be directly proportional to $(k_1 + k_2 + \ldots + k_n)$; instead, it will be based on the maximum time that a single disk takes to serve the blocks retrieved from it.

Now, let us assume that each of the *m* disks $D_1, D_2, ..., D_m$ has to serve $C_1, C_2, ..., C_m$ blocks in a particular round, where

$$\sum_{i=1}^{m} C_i = \sum_{j=1}^{n} k_j \tag{4.1}$$

And if the service time of disk *i* is denoted by τ_i, then

$$\begin{aligned}\tau_i &= a * C_i + b * T + l_{rot}^{\max} * C_i \\ &= \left(a + l_{rot}^{\max}\right) * C_i + b * T \end{aligned} \tag{4.2}$$

So, the real service time of the total system with *m* disks would be bounded by the maximum of all the service times of *m* disks; which is

$$\begin{aligned}\tau &= \max_{1\le i\le m}(\tau_i)\\ &= \max_{1\le i\le m}\left(\left(a + l_{rot}^{\max}\right) * C_i + b * T\right)\end{aligned} \tag{4.3}$$

We know that the acceptation criterion is $\tau \le R$, where minimum duration of a round, $R = \min_{j\in[1,n]}\left(\frac{k_j * M}{R_j}\right)$.So by using Equation (4.3) the admission control criterion in case of multiple-disk storage system can be formally stated as:

$$\tau = \max_{1\le i\le m}\left(\left(a + l_{rot}^{\max}\right) * C_i + b * T\right) \le \min_{j\in[1,n]}\left(\frac{k_j * M}{R_j}\right) \tag{4.4}$$

4.3.2 Description of the Algorithm

In the previous sub-section we have discussed the admission criterion of the Hybrid Admission Control Algorithm in multi-storage situation. Except for the admission criterion as shown in Equation (4.4) all other steps in the algorithm are almost similar to the standard procedure as shown in Figure 3.2. As a consequence, the only difference is in the 2nd step as shown in Sub-section 3.3.2 previously. We will not increase the volume of the description by unnecessarily reiterating through the same steps. However, the main change will be needed in the implementation side of the algorithm. It is much more complex in this case as we have to implement a parallel access file system to simulate the environment of multiple disks.

4.4 Analysis of the Algorithm

The analysis in this case is almost similar to that of the basic algorithm except for the disk accessing portion of the decision making process. As usual, we classify the clients into two groups; super and normal user group. Each group has an average rate of inter-arrival times. In simulation, client requests were generated using Poisson process.

- After classifying client request, multimedia server tests whether or not to admit it. The computational time to test admission condition, i.e. whether the request is accepted or declined, is linear. So computational complexity of is $O(n)$, where n, is the number of clients that are currently accessing server resources.

- The actual improvement corresponding to the procedure introduced in this chapter occurs in this step. Here, multimedia server determines the blocks that are to be retrieved in the next round. After determining the blocks, disk-scheduling algorithm is responsible to generate the order of blocks to be retrieved. With a single disk, we found previously that SSTF gives the best performance with $O(n \lg n)$ complexity, where n is the number of blocks in the queue. Now with the use of multiple disks, this time is effectively reduced by factor proportional to the number of parallel disk usage.

- As in previous case, the computation complexity of deterministic criterion is constant and that of statistical criterion is $O(n)$, where, n is the number of clients that are currently accessing server resources.

So, except for the disk access time all other times are equal in this case. But from the simulation results presented in the next chapter it is evident that this only one improvement improvises the outcome greatly.

4.5 Summary

The Multi-Disk extension gives a major boost to the performance of the Hybrid Admission Control Algorithm. Although it is practically not related to HACA in terms of its integration to the algorithm, it is important because it works as an ancillary improvement that works very efficiently when used with the standard algorithm. It also paves the way to further experiments that would reduce disk access time effectively. In a word, anything that would reduce disk access time i.e. service time of the multimedia server should result in improvement of the admission control decision time and Multi-Disk extension is no exception to that fact.

Chapter Five

Simulation Results

5.1 Introduction

In the last few chapters, we have described the basic Hybrid Admission Control Algorithm and proposed a multi-disk extension to it. In this chapter, we demonstrate their viability by analyzing the performance of the basic and extended algorithm through simulations. We also present relative performance measurement of disk scheduling algorithms. The performance measures of different admission control schemes may help us to understand how various systems can be better utilized to serve multimedia clients. The following metrics were used to evaluate our hybrid admission controllers and compare their performance with deterministic and statistical admission control schemes.

- **Number of Clients Admitted**: The total number of clients that were admitted for a multimedia session out of a large number of requesting clients.
- **Number of Overflow Rounds**: In serving the multimedia clients, there could be a number of rounds that may exceed the duration of a round.

5.2 Simulation Steps

The main simulation steps of our experiments are shown in Figure 5.1. We have coded the program including the HACA and simulated file system with C++ programming language using MS VC++ 6. The simulation was carried on a 2.4 GHz single processor Pentium-IV PC with 1GB of RAM. After the initialization of the program, the file system is generated and distributed according to the simulation parameters set beforehand. After the file system with multiple disks is in place, we randomly generate clients of different user groups and let them go through the actual process that would have occurred in real case. For each client necessary operations of disk access is calculated by using the simulated file system. After a predefined number of iterations the simulation is stopped.

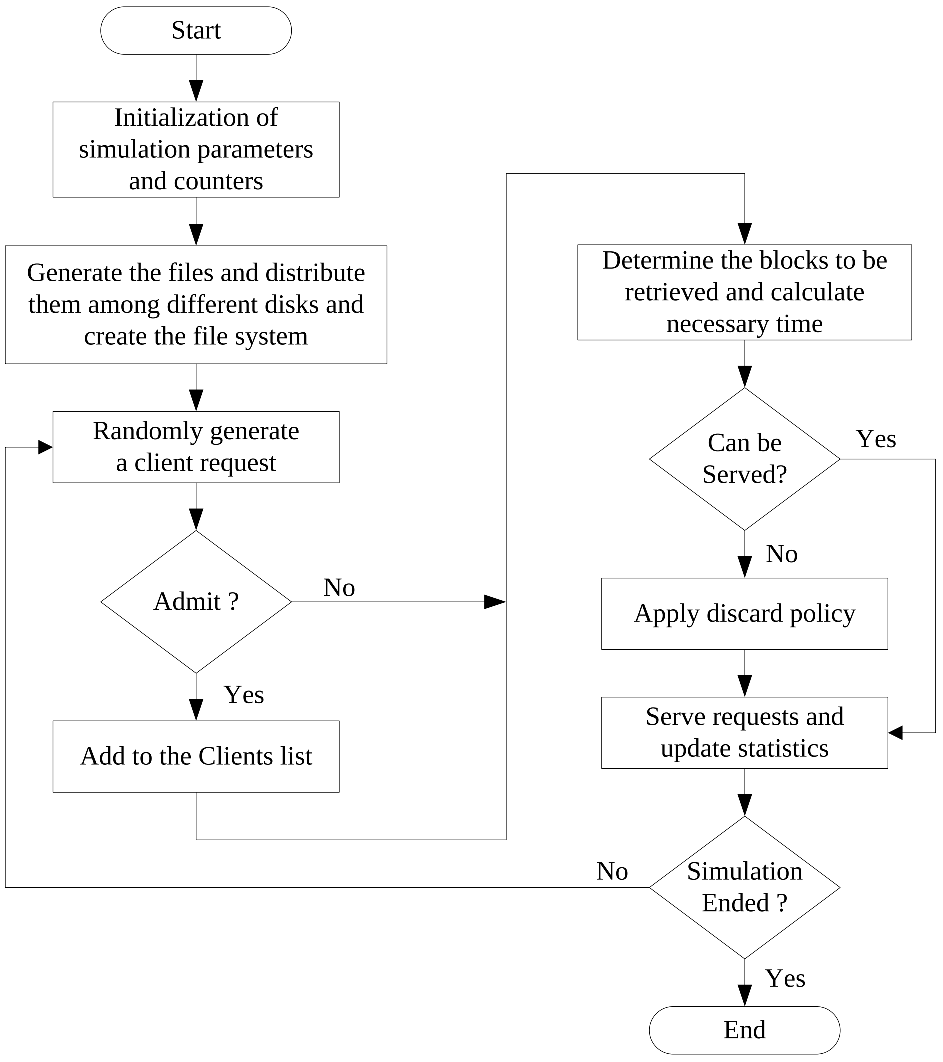
Start
Initialization of simulation parameters and counters
Generate the files and distribute them among different disks and create the file system
Randomly generate a client request
Admit ?
No
Yes
Add to the Clients list
Determine the blocks to be retrieved and calculate necessary time
Can be Served?
Yes
No
Apply discard policy
Serve requests and update statistics
No
Simulation Ended ?
Yes
End

Figure 5.1: *Simulation Steps*

5.2.1 Imitating the Real Environment

In practice there will be all kinds of audio and video files in the multimedia server. And for each of them there would have been different real-time service requirements. To imitate that, we have randomly generated files of different sizes and logically placed them in different disks. We have also defined necessary parameters for each file. For the ease of our experiment we have not literally generated the files; instead we have only made FAT entries and for each file we have created chain of blocks that would have hold the file if it really existed.

Because of these assumptions, there might be a slight difference with the real implementation of the system but we have tried to incorporate all the parameters regarding disk and file system so that the discrepancy remains to a tolerable limit.

5.3 Simulation Parameters

The simulations were carried out in an environment consisting of a synchronous disk array (Access arms/heads move in unison); unless otherwise specified. The characteristics of disk are shown in Table 5.1 with units. For our simulation, we assume two categories of client requests, namely Super user and Normal user where only a Normal user has a tolerance level.

Table 5.1: *Disk Parameters assumed in the simulation*

Parameter	Relevant Information	Unit
Disk Capacity	60	GB
Number of disks in the array	15	
Number of tracks per disk	1024	
Number of blocks per track	128	
Disk block size	32	KB
Rate of disk rotation	5400	RPM
Seek formula	$4+0.02*\|C_1 - C_2\|$	ms.
Max seek time	24.48	Ms
Max rotational latency	11.11	Ms

Arrivals of client requests of each category were generated using a Poisson process, with average inter-arrival times of 3 seconds and 1 second, respectively. During each round, exactly one block from each of the disks was retrieved for each client. Since the block size is assumed to be 32 KB, total amount of media information retrieved during each round is 512 KB. Assuming that the playback rate of each strand is 512 KB/second yields R = 1 second (R is the minimum duration of a round). The simulation parameters for admission control algorithm are given in Table 5.2.

Table 5.2: *Simulation parameters of Hybrid Admission Control Algorithm*

Parameter	**Values**
Playback rate	30 frames/sec
Frame Size	16.25 KB
Average inter-arrival times of Super user	3 sec
Average inter-arrival times of Normal user	1 sec
Average no of frames retrieve on time, P	80%
No of previous rounds used to determine average block retrieval time	20
Epsilon/ Overflow control parameter	0.0 – 1.0
Safe Guard	0 %– 10%
Disk Scheduling Algorithm	SSTF-E

5.4 Simulation Results

In this section we describe and discuss the results of all the simulation experiments that we have done. Extensive simulation of the performances of each of the disk scheduling algorithms has been done before choosing the best for using with the admission control algorithms. Also we have simulated the behavior and performance of all the admission control algorithms using the selected disk scheduling algorithm. We have also experimented on the effects of using multiple disks alongside HACA to find out its effect. All of these experimental results are presented in the following subsections.

5.4.1 Selection of Disk Scheduling Algorithm

As we have mentioned in previous chapters, the performance of any admission control algorithm largely depends on the readiness with which it can fetch requested disk blocks from its storage device. So the effect of disk scheduling is of paramount importance and we have to find out the best among the available ones. We have used the extended versions standard disk scheduling algorithms like FCFS, SSTF and SCAN with rotational latency taken into account and we denote the extended versions by FCFS-E, SSTF-E and SCAN-E respectively. We have also taken into our consideration, a new algorithm NODSA, which considers rotational latency as well.

5.4.1.1 Effect on Service Time

If we consider the service times of all the disk scheduling algorithms we find that in terms of service time NODSA performs the best and FCFS-E the worst. That is if we employ NODSA to schedule the disk requests the outcome would be the fastest possible retrieval. From Figure 5.2 we can see that the 2nd best is the SSTF-E algorithm. It performs just over NODSA and SCAN-E is also tolerable. If *n* is the number of blocks then for all values of *n* this observation holds. In fact, as *n* increases FCFS-E gets much more intolerable than others.

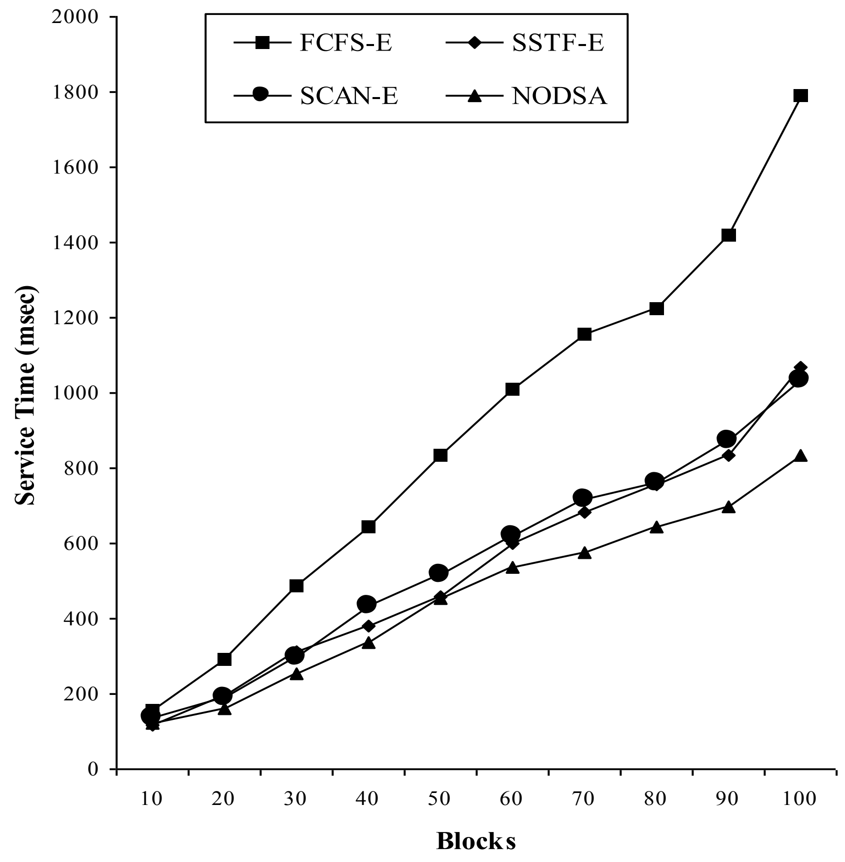

Figure 5.2: *Comparative service times of different disk scheduling algorithms*

5.4.1.2 Effect on Computation Time

In [15], it was assumed from the result of Figure 5.2 that NODSA will be the best for the purpose of disk scheduling alongside HACA. But we also have to consider the time it takes to take the decision. Because if that gets too large then no matter how fast it will retrieve data later will be undermined by its initial sluggishness. In Figure

5.3 we see the computation times of all the disk scheduling algorithms in a logarithmic scale. And it is evident that, NODSA requires too much of initial computation time which can not be tolerated. Here FCFS-E is the best as it has nothing to do to take the decision. On the other hand SCAN-E and SSTF-E, with their almost similar time complexity, performs almost similar. But SSTF-E is slightly better.

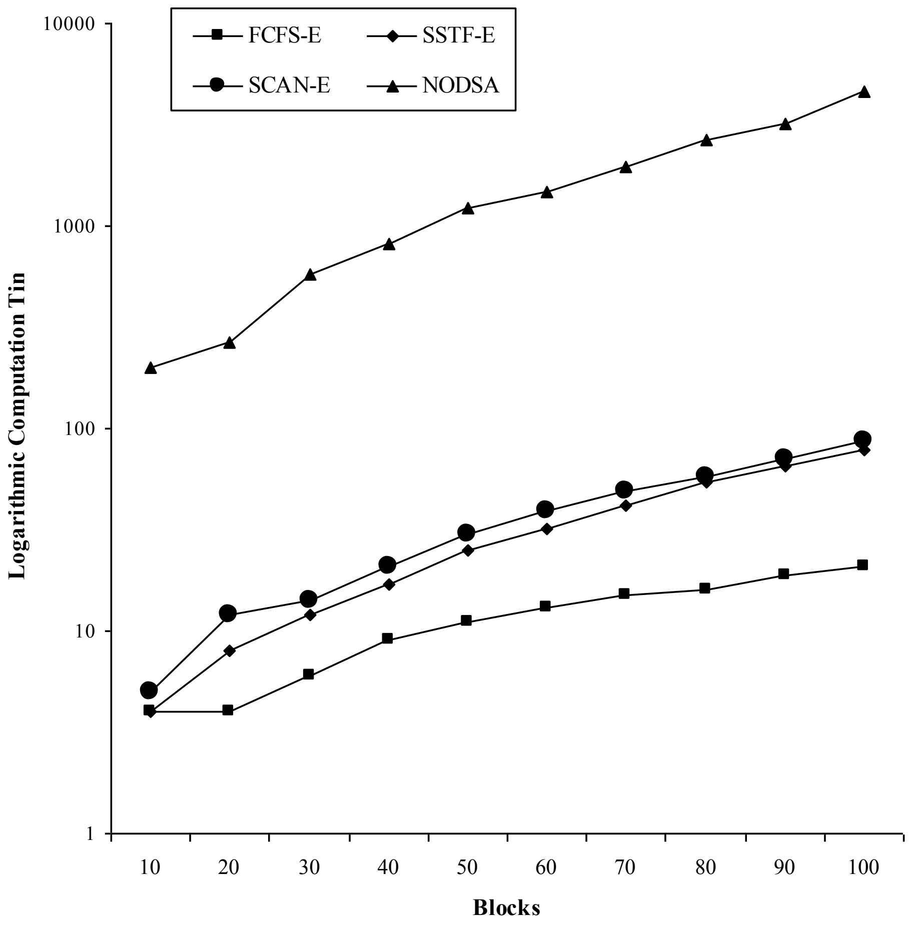

Figure 5.3: *Comparative computation times of different disk scheduling algorithms in Logarithmic Scale*

5.4.1.3 Selecting the Best Disk Scheduling Algorithm

From the previous two sections it is evident that SSTF-E is the only algorithm that performs best in terms of both Service time and Computation time. So in our simulation we have chosen it as the default disk scheduling algorithm to be used with all the admission control algorithms for further experiments. The validity of our choice will also be justified in sub-section 5.4.2

5.4.2 Comparison of Different Admission Control Algorithm using Different Disk Scheduling Algorithms

In the previous sub-section we have chosen SSTF-E as the best disk scheduling algorithm to be used in the multimedia server. To justify our choice further and to check the effects of different disk scheduling algorithms over different admission control algorithms we have carried on extensive experiments. We discuss the findings in terms of number of clients and number of overflow rounds – the main two performance parameters of admission control algorithms.

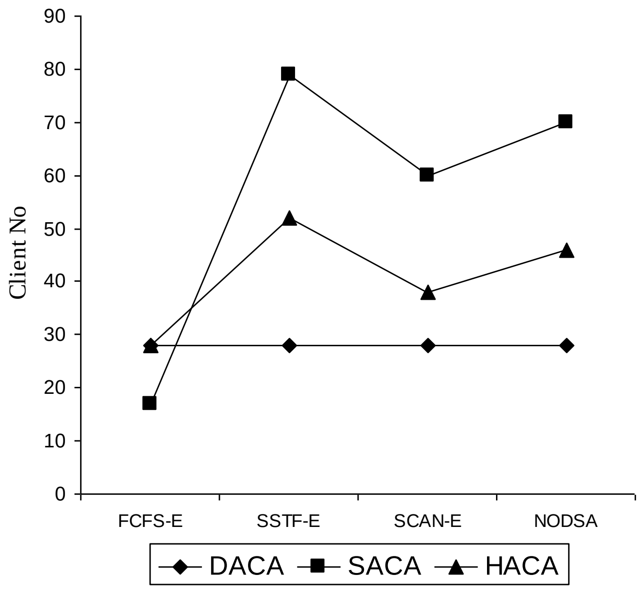

Figure 5.4: *Influence of using different disk scheduling techniques on the Number of Clients for different ACA*

5.4.2.1 In terms of Number of Clients

From Figure 5.4 we can see that SSTF-E ensures that the maximum numbers of clients are admitted into the multimedia server for any of the three admission control algorithm. For DACA it is constant as we know that its performance is independent of any statistical variation. Another point to note is that FCFS-E performs the worst in all cases. SACA admits the maximum number of clients much more than its other two counter parts i.e. HACA and DACA.

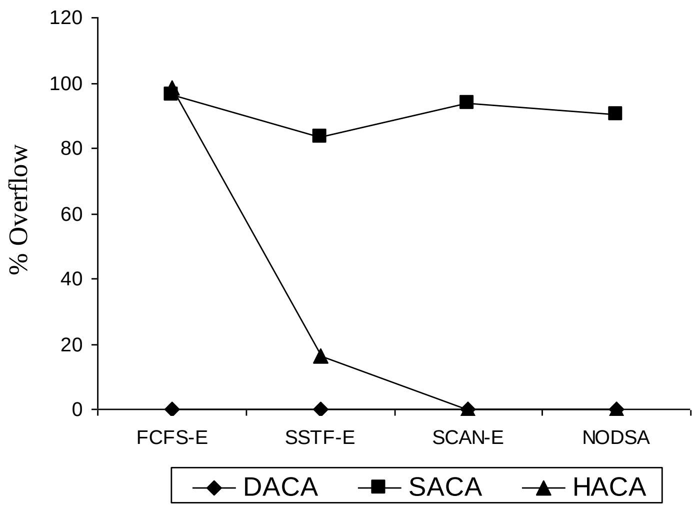

Figure 5.5: *Influence of using different disk scheduling techniques on the number of Overflow Rounds for different ACA*

5.4.2.2 In terms of Overflow Rounds

Continuing from the previous paragraph, we see in Figure 5.5 that both SCAN-E and NODSA performs well if we consider number of overflow rounds. SSTF-E also performs closer to those but FCFS is the best of all – no overflow at all. We also see that DACA causes no overflow rounds and HACA is almost free of those. The advantage that SACA gained in terms of number of clients is diminished here as it causes huge overflow rounds which means huge failure in maintaining real-time service requirements of the admitted clients.

5.4.3 Decision Making Time of Different Admission Control Algorithms

Another important factor to consider is the time each admission control algorithm to take its decision. It is important as if any one takes too much time it will cause an initial lag and will eventually fail afterwards as the lag time will accumulate with each incoming request. For DACA the decision making time is *O(1)* and it is *O(n)* for SACA. As HACA is effectively a combination of the two, its time requirement is also linear i.e. *O(n)*. And our simulation shows although HACA takes linear time it is lower than SACA by a certain factor as shown in Figure 5.6.

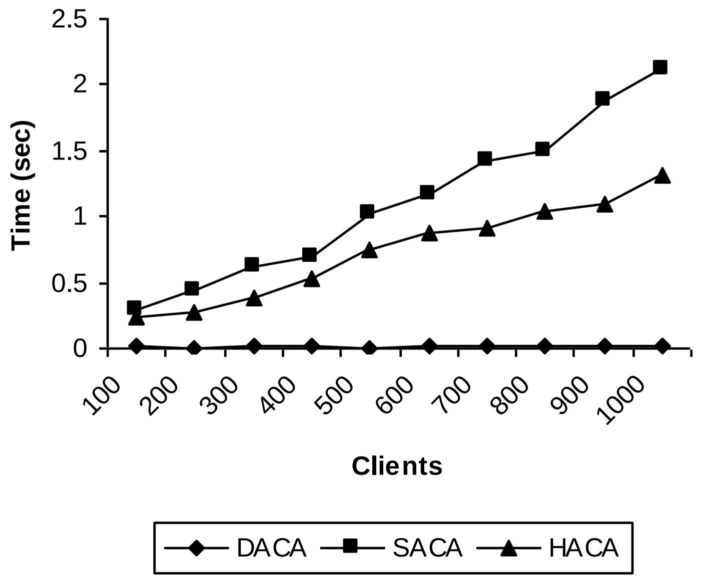

Figure 5.6: *Relative decision making time of different admission control algorithms*

This observation ensures that HACA is better than SACA also in terms of quickness. Though it is a little slower than DACA, but the previous findings regarding its acceptance rate and tolerable overflow rounds makes HACA superior to both of its counter parts.

5.4.4 Controlling the Performance – Epsilon and Safe Guard

In the previous chapters we have proposed two parameters to control the number of overflow rounds and consequently the performance of the multimedia server. These are safe guard and $\in$. In our experiments we have varied safeguard from 0% to 10% and $\in$ from 0.0 to 1.0. The results are discussed in the following.

5.4.4.1 Influence of Safe Guard

As we vary the value of safe guard from 0% to 10% we see significant change in both number of clients and overflow rounds. As safeguard increases overflow rounds decreases very sharply when $\in$ is constant. The change in the number clients is non-increasing with the increase of the value of safe guard. The exact figures are presented in Table 5.3.

Table 5.3: *Effect of using Safe Guard on the Number of Clients and the Overflow Rounds for ∈ = 0 and ∈ = 1*

Safe Guard %	**∈ = 0**		**∈ = 1**	
	Max Clients	**% of Overflow**	**Max Clients**	**% of Overflow**
0	52	16.5	54	6.54
2.5	50	2.34	52	2.93
5	54	1.48	51	2.93
7.5	51	1.36	48	0
10	50	2.83	42	0.15

5.4.4.2 Influence of Epsilon (∈)

Again if we change the value of ∈ with safeguard remaining fixed we find that number of overflow rounds is less when ∈ = 1. We also find that there is a little or no influence of ∈ on the number of clients as we change ∈ from 0 to 1. This is because epsilon has little influence in admission policy; it only adjusts the average block retrieval time for the next round. The outcome of the experiment regarding values of ∈ is shown in Figure 5.7.

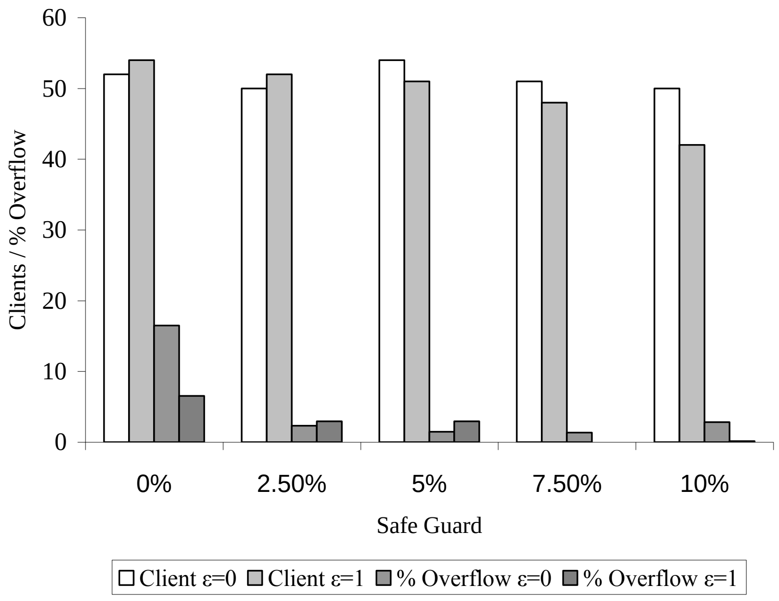

Figure 5.7: *Influence of ∈ on the Number of Clients and Overflow Rounds*

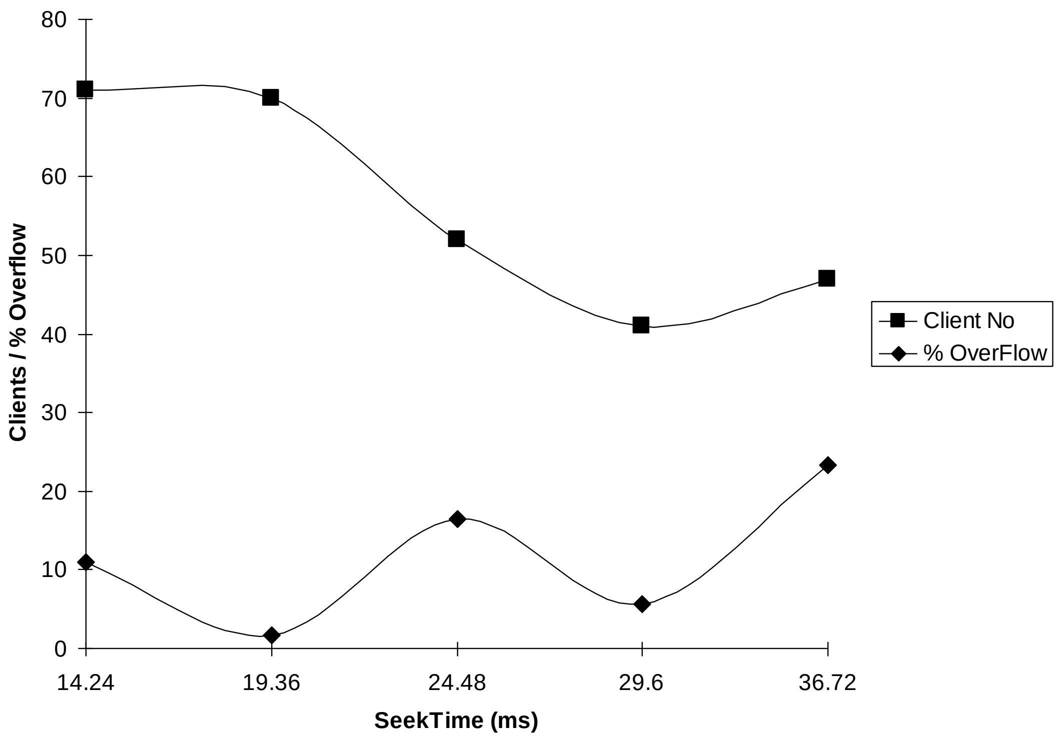

Figure 5.8: *Influence of Seek time on the Number of Clients and Overflow Rounds*

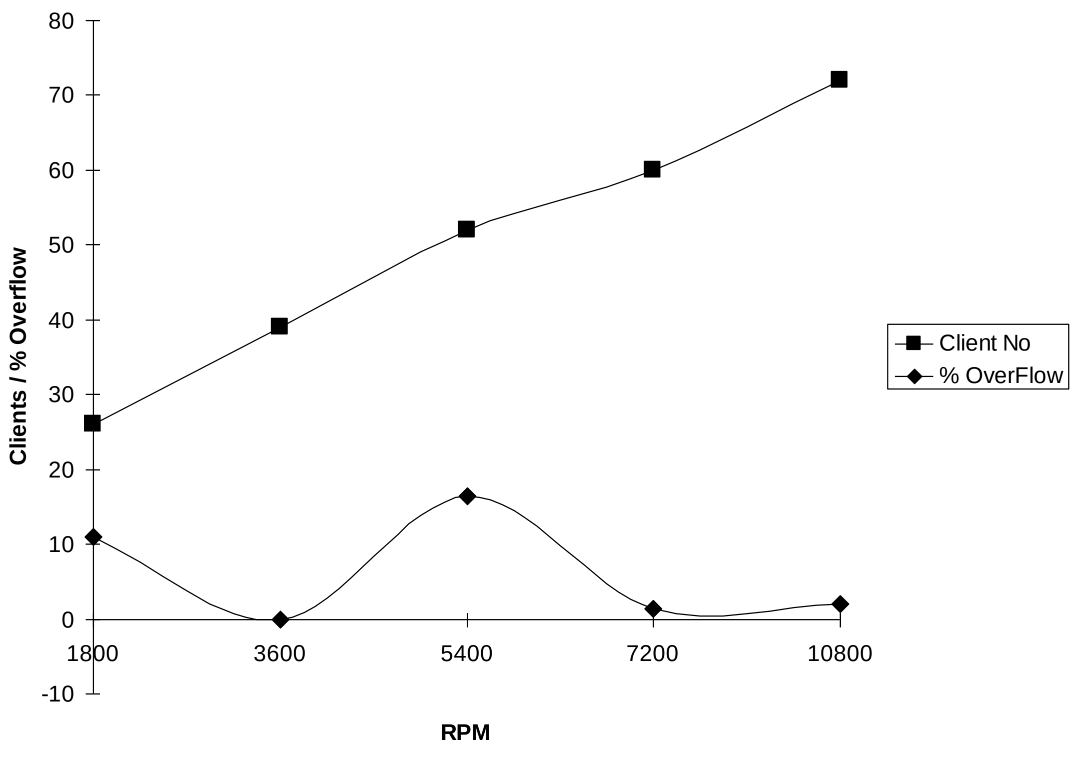

Figure 5.9: *Influence of Disk RPM on the Number of Clients and Overflow Rounds*

5.4.5 Effect of Disk Seek Time over Clients and Overflow Rounds

Theoretically, with the increase of seek time i.e. rotational latency of the disk it will take longer to retrieve same number of disk blocks than before. From our experiment we have found that with increasing seek time number clients decrease significantly but the increase of overflow rounds is not very clear. But the trend is increasing for overflow rounds and ultimate result is that increasing Seek time decreases performance of the Hybrid Admission Control Algorithm. This fact is presented in Figure 5.8.

5.4.6 Effect of Disk RPM over Clients and Overflow Rounds

Like the effect of Seek time, Disk RPM also plays a similar kind of role. With the decrement of Disk RPM it will take longer time to retrieve disk blocks and the result is decreased performance. From Figure 5.9, we can see that increasing RPM increases number of Clients admitted linearly but the effect on number of overflow rounds is erratic. This erratic behavior is an interesting finding of the simulation.

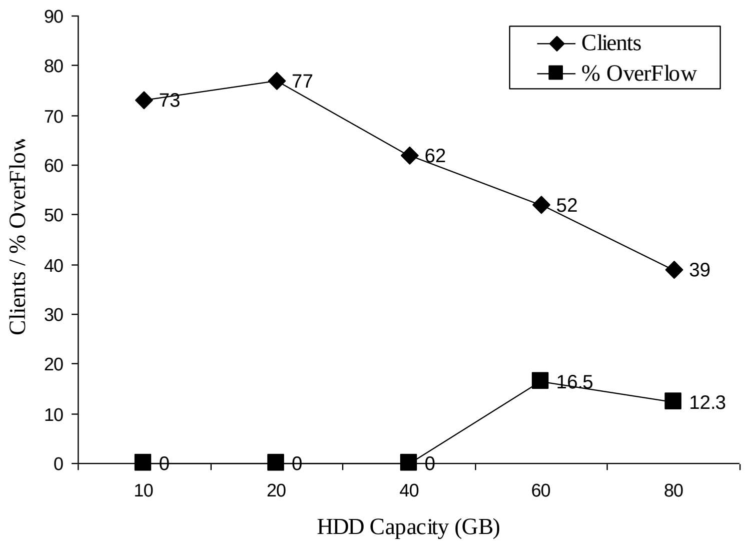

Figure 5.10: *Effect of HDD capacity on the Number of Clients and Overflow Rounds*

5.4.7 Effect of Disk Capacity over Clients and Overflow Rounds

In Figure 5.10 we present the performance of hybrid admission control algorithm by varying disk capacity. Larger hard disks adversely affect the outcome. With the increase of hard disk capacity, files tend to be distributed in wider range of tracks resulting in increase of retrieval time. With the increase of capacity client number decreases considerably and percentage of overflow rounds increases accordingly, ultimately resulting in poor performance.

5.4.8 Effect of Using Different HDD configurations and numbers

As we have mentioned in Chapter 4, if we can increase parallelism in disk accessing the obvious result is the decrement of total service time. As a result, it will take much less time to serve the available requests and which will consequently reduce the number of overflow rounds. It will also increase the number of clients that can be admitted into the server. To check this, we have used different configurations of storages with total size remaining the same. In this experiment the total size is fixed to 60GB and we have divided the total capacity in different numbers of disks for each case. The result is as described before and shown in Figure 5.11.

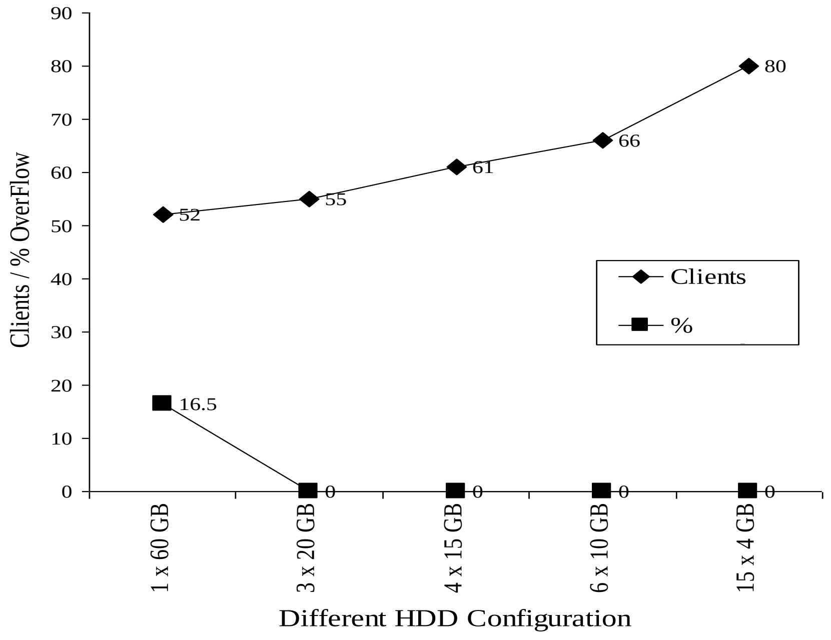

Figure 5.11: *Effect of using different multi-disk configurations on the Number of Clients and Overflow Rounds*

5.5 Summary

This chapter presented the simulation of the hybrid admission control algorithm. We have also simulated the behavior of the algorithm when multiple-disk is used. All the simulation parameters, its environment and results have been described and analyzed. We have explained the results in each section where the simulation result is shown. The most important thing is that, through this simulation experiments we have found some changes from the basic work of D.T. Ahmed [15]. All the relative changes have also been analyzed properly. The next chapter concludes this thesis with a brief description of our contribution and some points related to the future study.

Chapter Six

Conclusion

6.1 Major contributions

In this thesis we have studied and simulated the aspects of the Hybrid Admission Control Algorithm for multimedia server as a continuation of the work in [15]. This is an admission control algorithm that accepts enough traffic to efficiently utilize server resources, while not accepting those clients whose admission may lead to the violations of the service requirements of the clients. We have also introduced the idea the using multiple-storage extension of the algorithm that results in considerably reasonable increase in the performance of the algorithm – both in terms of number of clients and overflow rounds. We have simulated and demonstrated the validity of our techniques.

To recapitulate, the main ideas and observations of the research are as follows:

1. The Hybrid Admission Control Algorithm was presented to ensure that a multimedia server admits a client only if its admission does not violate the service requirements of all the clients already admitted. We have studied the results of the original work and have found out some differences between our results. We have tried to explain the discrepancies and suggested the changes that are required to alleviate the problem.

2. We have found that the performance of the algorithm largely depends on the time it takes to retrieve requested disk blocks from its storage devices. So we have proposed an improvement to the system that implements it. If the algorithm is used in conjunction with a file system where there are a large number of disks instead of a single one, the service time would decrease sharply resulting in a better acceptance rate.

3. The effectiveness of the admission criterion as well as the extension is illustrated through extensive simulations. By using hybrid admission control

algorithm multimedia server can serve more clients than deterministic admission control algorithm simultaneously. And using the multi-disk extension further boosts the number. As we have found, admitted number of clients can be increased from 52 to 80 by using the extension, whereas percentage of overflow rounds can be decreased to zero from 16.5% in the original algorithm.

4. Unlike the previous work, we have found that SSTF-E gives the best result if we take decision making time into account. NODSA fails when its fast service time is offset by the time it takes to find that fast solution. We have also shown the relative comparisons of different disk scheduling algorithms in general and found SSTF-E to be the best. We have verified the fact, by using it also in the multi-disk case.

6.2 Recommendation for Future Research

In order to understand and exploit the existing framework of operating systems and for building new ones, we believe that there are certain areas that require further research. In this section we suggest the following research plans to further excel the performance of the present state of the algorithm and corresponding multimedia system.

1. Some research can be done on the use of already well established Parallel file systems. This is because that there are many quality ones out there and we have only experimented with only a primitive one. Further research can be done on comparing between the performances of different parallel file systems and choose the best among the whole gamut of quality file systems.

2. Discarding policy can be further examined. Up to this point we have managed to ensure full guarantee to super user group and clients of normal user group are axed whenever necessary. Though the result is not very bad, but there is still scope to improve the quality of service of clients belonging to the normal user group. Balancing the failure between clients and ensuring fairness during

discarding is a very important factor that must be addressed in future research works.

3. Network delay can be classified as end-to-end delay and delay at resource. The delay “at the resource” is the maximum time span for the completion of a certain task at the resource. The end-to-end is the total delay for a data unit to be transmitted from the source to its destination. We have not paid much attention in determining their percentage. A precise work on classifying and identifying the delays and finding out actually how much delay is caused by the server will be an interesting thing to study.

4. A parallel processor multimedia server can be implemented to see the effect of pacing up the processing in the server side. In that case, NODSA might be able to shrug off its sturdiness in processing the decision and gain an advantage over all other disk scheduling algorithms. Also, the parallel implementation of the algorithm will also be of great interest to theoretical researchers.

References

[1] P. R. Barham, “A Fresh Approach to File System Quality of Service”, in *Proceedings of NOSSDAV’97,* (St. Louis, Missouri), pp. 119–128, May 1997.

[2] M. L. Claypool and J. Reidl, “End-to-End Quality in Multimedia Applications”, in *Handbook on Multimedia Computing,* ch. 40, Boca Raton, Florida: CRC Press, 1999.

[3] M. L. Claypool and J. Tanner, “The Effects of Jitter on the Perceptual Quality of Video”, *ACM Multimedia Conference,* October 1999.

[4] G. Miller, G Baber, and G. Gilliland, “News on-Demand for Multimedia Networks”, In *Proceedings of ACM Multimedia ’93,* Anaheim, CA, pages 383-392, August 1993.

[5] D. Ferrari and D.Verma, “Aschemefor real-time channel establishment in wide-area networks”, *IEEE Journal on Selected Areas in Communications*, 8(3): 368–379, April 1990.

[6] D. Clark, S. Shenker, and L. Zhang, “Supporting real-time applications in an integrated services packet network: Architecture and mechanism”, In *Proceedings of ACMSIGCOMM’92*, pages 14–26, Baltimore, Maryland, August 1992.

[7] D. Ferrari, “Client requirements for real-time communication services”, *IEEE Communications Magazine*, 28(11): 65–72, November 1990.

[8] H. Vin, P. Goyal, A. Goyal, “A statistical admission control algorithm for multimedia servers”, *Proceedings of the second ACM international conference on Multimedia*, p.33-40, October 15-20, 1994, San Francisco, California, United States.

[9] D. Anderson, Y. Osawa, and R. Govindan, “A File System for Continuous Media”, *ACM Transactions on Computer Systems*, 10(4): 311–337, November 1992.

[10] J. Gemmell and S. Christodoulakis, “Principles of Delay Sensitive Multimedia Data Storage and Retrieval”, *ACM Transactions on Information Systems*, 10(1): 51–90, 1992.

[11] P. Venkat Rangan and Harrick M. Vin, “Designing File Systems for Digital Video and Audio”, In Proceedings of the 13th Symposium on Operating Systems

Principles (SOSP'91), *Operating Systems Review,* Vol. 25, No. 5, pages 81–94, October 1991.

[12] F.A. Tobagi, J. Pang, R. Baird, and M. Gang, “Streaming RAID: A Disk Storage System for Video and Audio Files”, In *Proceedings of ACM Multimedia'93,* Anaheim, CA, pages 393–400, August 1993.

[13] Harrick M. Vin and P. Venkat Rangan, “Designing a Multi-User HDTV Storage Server”, *IEEE Journal on Selected Areas in Communications,* 11(1): 153–164, January 1993.

[14] P. Yu, M.S. Chen, and D.D. Kandlur, “Design and Analysis of a Grouped-Sweeping Scheme for Multimedia Storage Management”, *Proceedings of Third International Workshop on Network and Operating System Support for Digital Audio and Video,* San Diego, pages 38–49, November 1992.

[15] D. T. Ahmed, “A Hybrid Admission Control Algorithm for Multimedia Server”, *Master of Science Thesis,* CSE, BUET, September 2004.

[16] S. Childs, “Portable and Adaptive Specification of Disk Bandwidth Quality of Service”, *Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV ’99),* June 1999.

[17] H. M. Vin, A. Goyal, A. Goyal, and P. Goyal, “An Observation-Based Admission Control Algorithm for Multimedia Servers”, in *Proceedings of the First IEEE Inter-*135 *national Conference on Multimedia Computing and Systems (ICMCS’94),* (Boston), pp. 234–243, May 1994.

[18] P. Mohapatra and X. Jiang, “An Aggresive Admission Control Scheme for Multimedia Servers”, in *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pp. 620–621, 1997.

[19] S. Jamin, S. Shenker, L. Zhang, and D.D. Clark, “An Admission Control Algorithm for Predictive Real-Time Service (extended abstract)”, In *Proceedings of Third International Workshop on Network and Operating Systems Support for Digital Audio and Video,* San Diego, CA, pages 308–315, November 1992.

[20] P. Lougher and D. Shepherd, “The Design of a Storage Service for Continuous Media”, in *the Computer Journal*, 36(1): 32-42, 1993.

[21] M. Hofri. Disk scheduling: FCFS vs. SSTF revisited. *Communications of the ACM*, 23(11):645–53, November 1980.

[22] P. J. Denning. Effects of scheduling on file memory operations. Proceedings of *AFIPS Spring Joint Computer Conference* (Atlantic City, New Jersey, 18–20 April 1967), pages 9–21, April 1967.

[23] D. T. Ahmed, M. M. Rahaman, M. M. Akbar, “Near Optimal Disk Scheduling Algorithm” in *Proceedings of the 8th ICCIT Conference*, 2005.

[24] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, “Introduction to Algorithms”, Prentice Hall, 1998.

[25] Jason Barkes, Marcelo R. Barrios, Francis Cougard, Paul G. Crumley, Didac Marin, Hari Reddy, Theeraphong Thitayanun, “GPFS: A Parallel File System”, *IBM - International Technical Support Organization*, April 1998.

[26] Roger L. Haskin, “Tiger Shark - a scalable file system for multimedia”, *IBM Journal of Research and Development*, Volume 42, Number 2, March 1998, pp. 185-197.