University of California
Santa Cruz

Performance Evaluation of Systems with Restricted Overlap of Resources

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Engineering

by

David E. Levy

September 1994

The dissertation of David E. Levy is approved:

Alexandre Brandwajn

Patrick E. Mantey

Edwin J. Lau

Dean of Graduate Studies and Research

Contents

List of Figures

Performance Evaluation of Systems

with Restricted Overlap of Resources

David E. Levy

ABSTRACT

In this dissertation, we investigate the performance evaluation of systems with restricted overlap of resources. The impetus of this study is derived from the constrained parallelism in the service of I/O requests in the current generation of cached disk controllers. Among other features, these cached controllers offer improved performance through an increase in the degree of transfer concurrency. In particular, we consider controllers that have the ability to service the next request for a given logical device, if it can be satisfied by the cache, in parallel with "hidden" disk activity, such as staging. We present a queueing model to represent the restricted overlap problem. The model is developed under exponential assumptions and then expanded to account for variability in service. We investigate the efficiency of three solution techniques. The model is also expanded to represent a shared logical device.

Following the development of the restricted overlap model in the "classical" I/O setting, we apply the model to a disk array architecture and a multimedia kiosk storage architecture. The restricted overlap model is extended to represent a storage subsystem with a set of logical devices that map to a group of physical disks in a disk array. The bulk of recent efforts to characterize the performance of disk arrays appears to have concentrated on the effects of the aggregate bandwidth in a non-cached configuration. Our goal is to provide an analysis of a cached disk array architecture.

We also investigate the performance of a distributed kiosk storage architecture, where a set of multimedia stations are networked in an application group and each group is connected to an electronic library. Audio-video information-dispensing computers, known as kiosks, are becoming a popular technology for businesses and government agencies. Kiosks offer individuals a convenient means of information retrieval while reducing the demands placed on service organizations, by automating many routine processes. We present a model of an electronic library utilizing an optical jukebox with cyclic service of the storage platters. We also extend the restricted overlap model to represent a kiosk application group which uses preliminary sequence caching to improve the start-up latency.

Acknowledgements

I would like to thank my parents for their constant love and support.

I am grateful for the thoughtful comments and suggestions of my thesis committee, Alexandre Brandwajn, Patrick E. Mantey, and Edwin J. Lau. Professor Alexandre Brandwajn introduced me to the art of performance evaluation, in general, and the restricted overlap problem, in particular. I am indebted to him for his tutelage and guidance. I appreciate the generosity of Professor Patrick E. Mantey in allowing me to be a part of the electronic library and multimedia kiosk investigations, supported by the State of California Health and Welfare Agency Data Center, Sacramento, CA and IBM. Since I have been a student of the computer engineering department at UC Santa Cruz from its inception, I must also express my deepest thanks to Professor Mantey for his tireless efforts and vision in building the research and teaching content of the department. His commitment to excellence has been an inspiration. I would like to express my thanks to Dr. Edwin J. Lau for his time in reviewing my thesis. I am very grateful to him, as well as my other colleagues at NetFRAME, for their many insights drawn from practical experience.

In addition, I would like to thank Professor Anujan Varma for two quarters of support in working on his disk array destaging algorithm. Finally, I would like to thank the faculty (past and present), especially Professors Daniel Hellman, David Huffman, Harwood Kolsky and Glen Langdon, for contributing to a rewarding educational experience at UC Santa Cruz.

1. Introduction

The development of high performance computer and communication architectures over the past three decades has been the beneficiary of the simultaneous maturation of performance evaluation techniques. Computer performance evaluation is an engineering art borne out of the desire to optimize the performance of existing systems and future designs. The methods draw on mathematical models, system measurements, and intuition gained from a comprehensive understanding of the dynamics of the system or components to be evaluated.

Historically, the analytical branch of performance and capacity planning was created with the development of queueing theory by A.K. Erlang in the early 1900s [21]. His work was motivated by improving the design process of telephone exchanges. From the early part of this century, the techniques have been advanced by the mathematics, telephony engineering, and operations research communities. However, in the early 1960s with the emergence of the computer technologies, analytical modeling via queueing systems found a new domain of applicability and a new group of researchers. Scherr's representation of a computer time-sharing system [60] proved to be an effective performance model and provided a seed for the research of computer performance evaluation which continues to impact current system development.

This dissertation investigates performance evaluation methods for analyzing systems with restricted overlap of resources. The research is motivated by the type of constrained parallelism of I/O operations found in the contemporary generation of cached disk controllers. The controllers are designed to allow the simultaneous service of cache hits and the staging of track data. However, if a request is generated for a physical device during a staging period, the request is delayed until the staging completes. We refer to this constraint

on simultaneous service as *restricted overlap*. The contributions of this thesis include

- a representation of restricted overlap of resources which treats variability in service,
- a study of the efficiency of the solution techniques which are applied to the restricted overlap model,
- a disk array performance model, developed as an extension of the restricted overlap model,
- an application of the restricted overlap model to a multimedia kiosk storage architecture,
- a study of an electronic library, utilizing an optical jukebox with cyclic service of the storage platters, in a multimedia kiosk storage architecture.

In the following sections of this introduction, we present the origin of the restricted overlap problem which serves as the topic of this thesis, the relation of this work to previous literature, our approach to analysis, and the complete thesis structure.

1.1 Problem Origin

The benefits of caching, utilized throughout the memory hierarchy in computer systems design, depends on the well known locality of reference principle [18, 19]. In the late 1980's, a new generation of mainframe I/O controllers were introduced [33, 2] which capitalize on an intelligent multiported cached design. When a desired record is not found in the the cache, the cache storage buffers the disk request along with the remaining records on the track following the desired record. The aim of buffering the remaining records on the track, also referred to as *staging*, is to improve the cache read hit rate by prefetching data which may exhibit spatial locality.

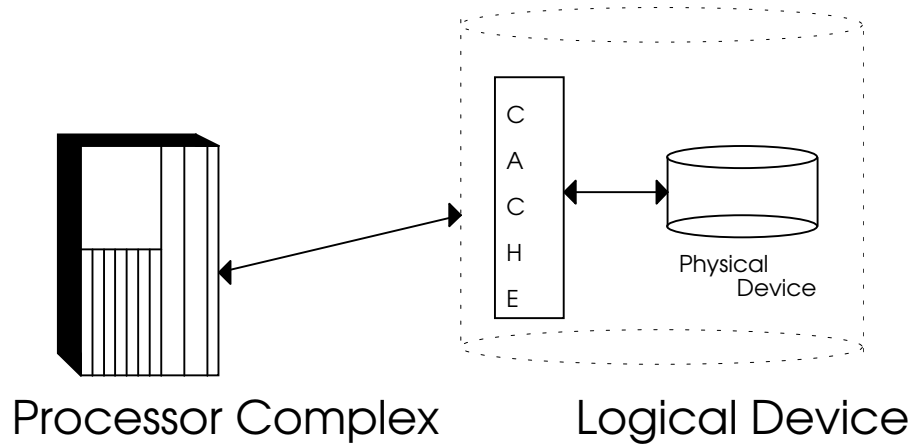

Figure 1.1: Diagram of processor complex and storage subsystem

Among other features, these cached controllers offer improved performance through an increase in the degree of transfer concurrency. The concurrency is facilitated by the multiple ports available on the cache of modern controllers. Thus, following a request for a given logical device which results in staging, the next request for the same logical device can be accepted. The advantage is that no delay is incurred if the subsequent request can be satisfied from the cache. We refer to the simultaneous service as *overlapped* service, and the staging activity as *hidden* since requests resulting in cache hits do not experience any delay. Naturally, if the next request requires an access to the physical device, the service experiences a delay until the staging activity is complete. Therefore, the overlap of service is restricted in the sense that a request for a record not in the cache, which occurs during a staging period, results in the serialization of the request (must wait on the single physical device).

Note that throughout this dissertation, we refer to the I/O devices as they appear to the operating system (including the effects of caching) and the actual storage devices as *logical devices* and *physical devices*, respectively. A model of restricted overlap was introduced in [8] to describe the *hidden* disk activity which occurs during the staging of a track. In the first part of this thesis, a variation of the model, based on a queueing model with vacations, is presented. This model and its extensions are developed in the body of this dissertation.

1.2 Related Work

Aside from [8], discussed in the prior section, there is little research, to the best of our knowledge, which is directed at the problem of restricted overlap of resources. The previous work in the literature which has the closest semblance to modeling restricted overlap includes the topics of modeling read/write access to shared resources, locking issues

in database theory, and fork-join queueing. However, their formulations do not seem to offer extensions which are readily applicable to modeling restricted overlap.

Since databases are typically a shared resource, the problem of concurrency control arises. It is popularly solved with locking methods. Thus, any read or write transactions must acquire some kind of token (or lock) before being allowed to proceed. Shared read/write access [16, 37, 51, 58] and locking models [69, 68, 71, 50, 30] have been developed to analyze the performance of such systems. Although the models seem similar to the idea of restricted overlap, their emphasis is on modeling the acquisition of the locks prior to the service of a request. Thus, a request is either admitted or blocked, but the effective service is serialized. (This is an important aspect of database management since a recovery procedure must allow the restoration to a consistent state in the event of an error). Our approach explicitly deals with the parallel service of the cache and staging activity subject to the operating system constraints.

The fork-join queueing systems [3, 51] are also applied to solving parallel processing problems. However, these models typically deal with breaking, *forking*, a task into subtasks which individually queue at separate servers. The task is then considered complete when all of the subtasks complete service, *join*. These systems involve a restriction on the network of "subtask" queues as opposed to constraining the service stages of an individual queue as in our approach to the restricted overlap problem. We note the fork-join queueing systems have been used effectively to model RAID-5 devices [43, 72], which will be discussed further in Chapter 3.

1.3 Analysis Methodology

The thrust of this research centers around an analytical representation of restricted overlap and its application to pragmatic performance issues. Typically, performance modeling efforts are approached from simulation and/or analytic techniques. The advantage of simulation is that the system to be modeled can be expressed in arbitrary detail. However, the virtue of generality is tempered by possibly costly run times. Thus, in this thesis, we investigate analytic models with a level of abstraction that captures the system's salient features and corresponds to a representation which can be solved efficiently.

In this thesis, the role of simulation is as a tool of validation for the analytic models developed. Details of simulation theory can be found in such related texts as [23, 47]. Some of the simulations in this thesis were written in *CSIM*, a process-oriented simulation language [61], and others using a simple discrete event simulator written in C and based on *SMPL* [47]. The approximate confidence intervals for the simulation runs were estimated at a 95% confidence level. In our application, we attempted to maintain confidence intervals half-widths within 10% of the generated average values. The need for efficient analytic models is underscored when long simulation run times are experienced due to increased service variability. (Although not investigated in this dissertation, variance reduction techniques may provide a means to reduce run times, see for example [42, 10].)

1.4 Thesis Structure

This thesis is structured into three main chapters plus a conclusions chapter. Chapter 2 investigates modeling restricted overlap of resources as it applies to the current generation of cached disk controllers in a "classical" mainframe I/O environment (described in the

previous section, *Problem Origin*). We start by presenting three solution approaches to the restricted overlap model under exponential assumptions. The efficiency of each technique is assessed and the most efficient method of the three is adopted for the subsequent extensions of the model. The extensions of the model which appear in Chapter 2 include representations which account for variability in service and pending times due to resource sharing.

Chapter 3 presents a further extension of the restricted overlap model as it applies to storage systems with logical devices that map to a group of physical disks in a disk array configuration. In particular, we investigate the representation of a RAID-5 device which includes the effects of a cached controller. The work in Chapters 2 and 3 are a product of a joint research effort with Professor Alexandre Brandwajn.

A multimedia application is the subject of Chapter 4. Audio-video information dispensing computers, known as kiosks, are becoming a popular technology for businesses and government agencies. Kiosks offer individuals a convenient means of information retrieval while reducing the demands placed on service organizations, by automating many routine processes. Today, kiosks are typically autonomous machines, delivering audio and video in an analog format. In this chapter, we consider a digital, distributed system, consisting of a group of kiosk clients and an electronic library, all linked by a network. We investigate a model for representing a cyclic service scheme of the storage platters in an optical disk jukebox, used as an archival store in the electronic library. Then, we apply a variation of the restricted overlap model to represent the kiosk group. The work in Chapter 4 is a product of a joint research effort with Professor Patrick E. Mantey.

Finally, in Chapter 5, we summarize the previous chapters and discuss possible directions for future research.

2. Modeling Restricted Overlap of Resources

As discussed in the introduction, the problem of restricted overlap of resources is approached from the perspective of the I/O subsystem architecture utilizing an intelligent, cached controller. Although the problem arises in this application, our model and its solution are a mathematical abstraction of the overlap problem which can be applied to a class of systems which map to the overlap model. In this chapter, we develop the restricted overlap model and use the current generation of cached disk controllers as a practical example of its applicability.

As previously mentioned, we use the terms *logical device* and *physical device* to describe an I/O device as it appears to the operating system and an actual disk storage device, respectively. When a request is generated for a particular record on a logical device at the operating system level, it will be serviced by the cache if it is available. Otherwise, if the physical device is not busy, the record is transferred from the disk. Typically, the rest of the track, following the requested record, is staged into the cache. Thus, the physical device remains busy for an extended period of time beyond the desired record transfer time. However, subsequent requests which are generated during the staging period and result in cache hits can be served simultaneously with the staging. This resulting overlap is facilitated by the multiported caches in contemporary cached controller designs.

Since requests are viewed from a single physical device, any read miss has the effect of restricting the overlap that can occur at the controller by serializing service until any hidden activity completes. Addtional constraints can be present due to operating system rules. For example, according to the current principles of operation of MVS/XA and MVS/ESA [32, 35], only a single I/O request can be outstanding for a given logical device, *i.e.* there are no

multiple exposures. Therefore, the overlap of service is restricted in the sense that a request for a record not in the cache must be serviced from the physical device, and signaled as complete, before any other requests are serviced by the cache on behalf of the same logical device.

In dealing with writes requests, there are typically two approaches that are considered: *write-through* and *write-back*. Write-through insures the data is written to stable storage by directing the write to the physical disk. Write-back operations write to a cache, signal the write complete, and then asynchronously write the data to disk in a destaging process. The DASD Fast Write operation [33] is similar to a write-back, except data is written to non-volatile storage (NVS) in order to guarantee it permanence. For the purpose of our treatment of restricted overlap, we assume a cached controller of sufficient NVS capacity to manage write requests as DASD Fast Writes. However, the model can represent a write-through management scheme by adjusting the model parameters (discussed later in this chapter).

A model of restricted overlap was first proposed in [8] to model device overlap found in the current generation of cached controllers in computer systems running MVS. The model, shown in Figure 2.1, accounts explicitly for service overlaps. In the model, stage I represents the time to service a request for a particular record on a physical device; stage II represents the time to service a cache request; and stage III represents the time to stage the remainder of a track, following the desired record. Under the state description presented in [8], the model operation can be described as follows:

1. Requests enter the queue as one of two classes:
   - *Class 1* - service from stage I (occurring with probability p)
   - *Class 2* - service from stage II (occurring with probability $q = 1 - p$)

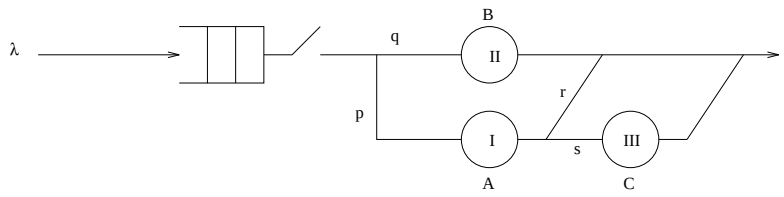

Figure 2.1: Queueing model of restricted overlap (initial representation)

2. First-come first-service (FCFS) queueing service discipline with the following restrictions:
   - stages I and II operated exclusively as do stages I and III
   - *Class 1* requests enter stage III for further service with probability, s
   - stage combination II and III can have simultaneous operation - *overlapped service*

Although not explicitly presented in [8], this queueing model, with exponential assumptions for each service stage, leads to a set of balance equations which can be solved using finite difference techniques. For future reference, we will call this model the "initial" model. Upon further study of the problem, a variation of this model, based on service with vacations, was proposed by [9]. This model has a closer correlation to the I/O subsystem dynamics and does not appear to have numerical difficulties with some iterative methods that were experienced with the initial model.

The restricted overlap vacation model and its associated state description provide the kernel of this thesis and we will develop several approaches to solving this model.

The vacation model state description differs from the "initial" model in basically a single bullet point listed above. In the vacation model, stage III is a vacation period for the stage I server, *i.e.* the duration stage I is unavailable. Therefore, where the "hidden" service was described as:

- *Class 1* requests enter stage III for further service with probability, s

we now replace with:

- stage I enters a vacation period (stage III) with probability, s

The model is shown in Figure 2.2 and the model's service states are shown in Figure 2.3.

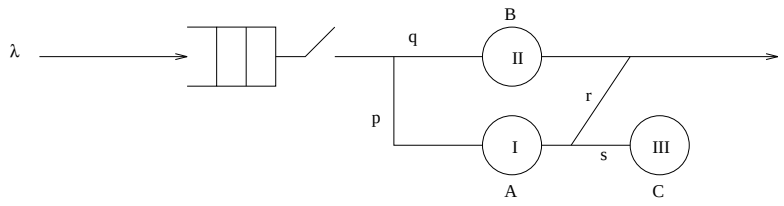

Figure 2.2: Queueing model of restricted overlap with vacation stage

The variables used to describe the probabilities of entering these stages are also shown in the figure and are defined as follows:

- λ - the Poisson rate of arrivals

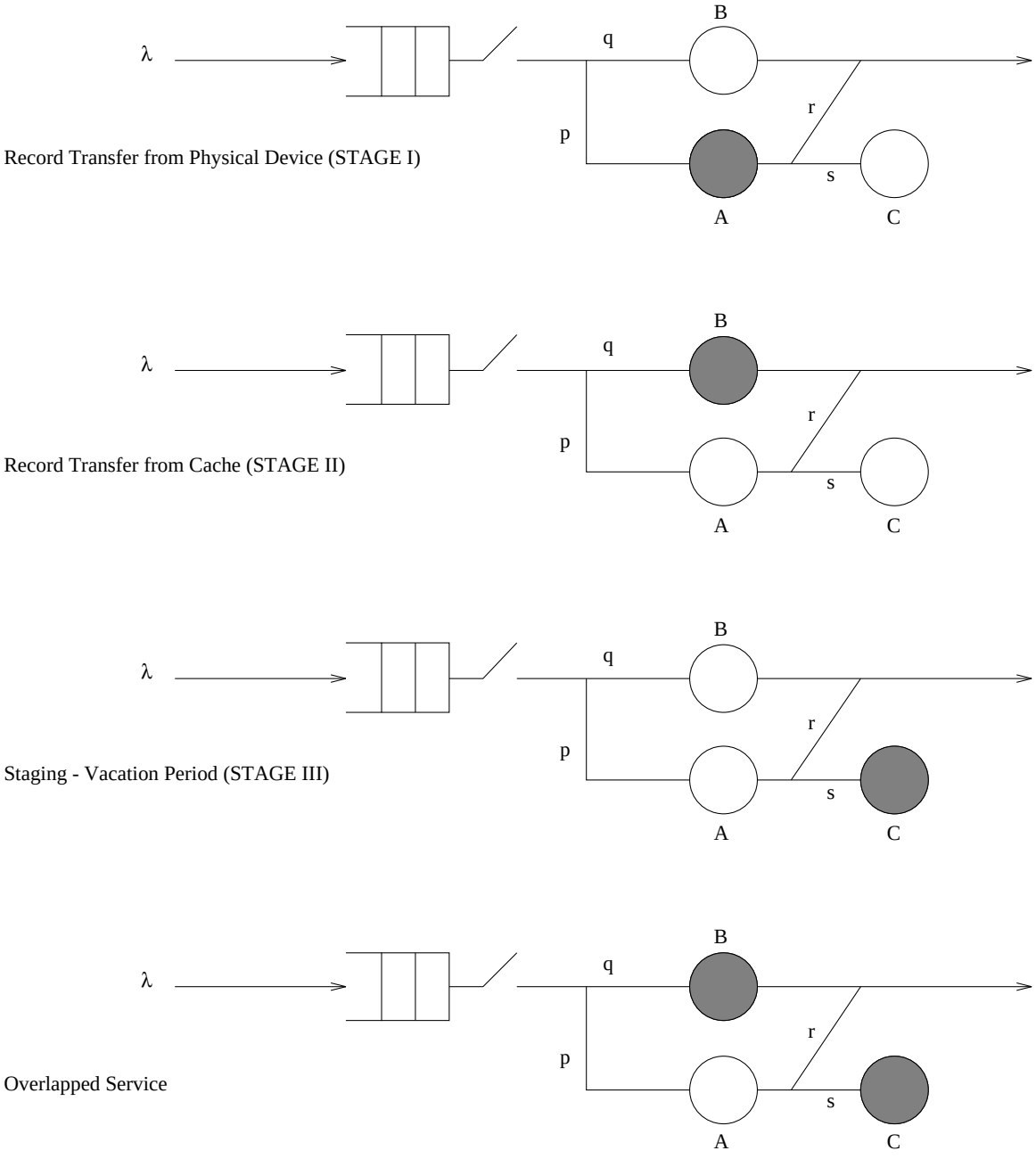

Figure 2.3: Service states of the restricted overlap model

- A, B, C - the random variables (RVs) representing the time spent at stage I, II, and III, respectively
- $\bar{a}, \bar{b}, \bar{c}$ - the mean of RVs A,B, and C
- $\bar{a^2}, \bar{b^2}, \bar{c^2}$ - the second moment of RVs A,B, and C
- α, β, γ - the rates of service at stage I, II, and III ($\alpha = 1/\bar{a}$, $\beta = 1/\bar{b}$, $\gamma = 1/\bar{c}$)

In the following sections, we develop several solution techniques to the restricted overlap model. The first section examines the problem under exponential assumptions in an effort to facilitate solutions to each method. We evaluate the efficiency of each technique, and adopt the best solution technique for subsequent extensions of the model.

2.1 Restricted Overlap Vacation Model under Exponential Assumptions

In this section, we develop a first order solution to the restricted overlap vacation queueing model by assuming each service stage is exponentially distributed. Even under exponential assumptions, the problem is challenging due to the presence of overlap. Certainly, a crude approximation can be generated by assuming no overlap and simply solving the resulting M/G/1 queueing system. However, our goal is a representation which accounts for the constrained parallelism.

The analytic techniques of vacation models developed in the literature, see monograph [67], do not readily appear to correspond to the restricted overlap problem. The "classical" vacation models typically view the server of a queue as either being in the state of service or vacation, *i.e.* unavailable. The models deal with various restrictions on service and vacation periods [20], *e.g.* exhaustive or limited service prior to vacations. However, they do not account for "partial" vacations, in which some service may be allowed, as with the simultaneous service of the cache in our model definition.

An exact solution to the queueing model's balance equations is generated using an iterative method based on equivalence [6, 7]. We start by choosing an appropriate state description, (n, S), which corresponds to the dynamics described in the introduction to this chapter. The variable n represents the number of customers in the system and $S \in \{I, II, III, \theta\}$ represents the state of service, where

$$
\begin{aligned}
I &\triangleq \text{server station I is busy exclusively} \\
II &\triangleq \text{server station II is busy exclusively} \\
III &\triangleq \text{server station III is busy exclusively} \\
\theta &\triangleq \text{server stations II and III are both busy (overlap).}
\end{aligned}
$$

Using the above state space definition, the balance equations can be expressed as:

$$
\begin{aligned}
p(n,I)(\lambda+\alpha) &= p(n-1,I)\lambda + p(n,III)\gamma + p(n+1,I)\alpha rp + p(n+1,II)\beta p \\
p(n,II)(\lambda+\beta) &= p(n-1,II)\lambda + p(n,\theta)\gamma + p(n+1,I)\alpha rq + p(n+1,II)\beta q \\
p(n,III)(\lambda+\gamma) &= p(n-1,III)\lambda + p(n+1,I)\alpha sp + p(n+1,\theta)\beta p \\
p(n,\theta)(\lambda+\beta+\gamma) &= p(n-1,\theta)\lambda + p(n+1,I)\alpha sq + p(n+1,\theta)\beta q
\end{aligned}
$$

where $n \geq 2$ and the boundary conditions can be written as

$$
\begin{aligned}
p(0,I\bar{I}I)\lambda &= p(1,I)\alpha r + p(1,II)\beta + p(0,III)\gamma \\
p(0,III)(\lambda+\gamma) &= p(1,I)\alpha s + p(1,\theta)\beta
\end{aligned}
$$

$$
\begin{aligned}
p(1,I)(\lambda+\alpha) &= p(0,I\bar{I}I)\lambda p + p(1,III)\gamma + p(2,I)\alpha rp + p(2,II)\beta p \\
p(1,II)(\lambda+\beta) &= p(0,I\bar{I}I)\lambda q + p(1,\theta)\gamma + p(2,I)\alpha rq + p(2,II)\beta q \\
p(1,III)(\lambda+\gamma) &= p(0,III)\lambda p + p(2,I)\alpha sp + p(2,\theta)\beta p \\
p(1,\theta)(\lambda+\beta+\gamma) &= p(0,III)\lambda q + p(2,I)\alpha sq + p(2,\theta)\beta q
\end{aligned}
$$

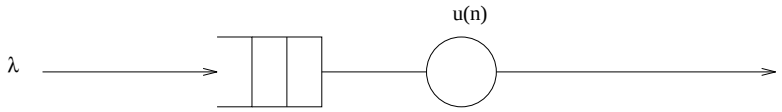

Figure 2.4: Equivalence model for restricted overlap service

Given these equations, we travel two paths toward a solution. The first is a semi-numerical method based on equivalence [6, 7]. Since the solution is iterative, we also investigate an alternative implementation to improve the speed of convergence which builds on the basic equivalence technique. The second path is based on the matrix-geometric method [53, 52], which capitalizes on the system structure resulting from the choice of state space.

2.1.1 Equivalence Approach - Simple Iteration

In applying equivalence techniques to the restricted overlap model, we start by considering the aggregate server shown in Figure 2.4. This queueing model has a service stage represented as a single server with aggregate service rate $u(n)$, where n is the number of jobs in the system. Assuming $u(n)$ is known, we can solve for the marginal probability of n customers in the system, $p(n)$.

$$p(n) \quad = \quad \frac{1}{G} \prod_{i=1}^{n} \frac{\lambda}{u(i)}$$

where G is a normalization constant. At this point, we are left with determining $u(n)$, which can be expressed as a function of the conditional probabilities, $p_n(S)$, defined as:

$$p_n(S) \triangleq \text{Prob}\{\text{service state is } S \mid n \text{ in the system}\}$$

Thus, we can write

$$\begin{aligned} u(n) &= \sum_S p_n(S) \cdot \mu_S \\ &= p_n(I) \cdot \alpha + (p_n(II) + p_n(\theta)) \cdot \beta \end{aligned}$$

and now turn to solving the conditionals. The solution is found in the balance equations which can be rewritten by substituting the conditional probabilities with their definition in terms of the joint probabilities, *i.e.*

$$p(n, S) = p_n(S) \cdot p(n),$$

see for example [1], and the definition of $p(n)$. Therefore, the general balance equations can be written as:

$$\begin{aligned} p_n(I)(\lambda + \alpha) &= p_{n-1}(I)u(n) + p_n(III)\gamma \\ &\quad + (p_{n+1}(I)\alpha rp + p_{n+1}(II)\beta p) \cdot \lambda / u(n+1) \\ p_n(II)(\lambda + \beta) &= p_{n-1}(II)u(n) + p_n(\theta)\gamma \\ &\quad + (p_{n+1}(I)\alpha rq + p_{n+1}(II)\beta q) \cdot \lambda / u(n+1) \\ p_n(III)(\lambda + \gamma) &= p_{n-1}(III)u(n) + (p_{n+1}(I)\alpha sp + p_{n+1}(\theta)\beta p) \cdot \lambda / u(n+1) \\ p_n(\theta)(\lambda + \beta + \gamma) &= p_{n-1}(\theta)u(n) + (p_{n+1}(I)\alpha sq + p_{n+1}(\theta)\beta q) \cdot \lambda / u(n+1) \end{aligned}$$

with the same substitutions applying to the boundary conditions. Since the equations above are expressed using $u(n)$, we apply a simple iteration, akin to the form of Jacobi's method [64]. The iteration calculates each successive iteration via the points calculated in the previous iteration. As a first level characterization of the model, this approach is sufficient.

However, in an effort to improve the iteration time to converge to a solution, we apply a modified iterative technique in the next section.

2.1.2 Equivalence Approach - Modified Iteration

In this section, we present a modified iterative method for solving the restricted overlap queueing model. We refrain from referring to this approach as "improved" until we quantify the results in the results section. The modified method, similar to the approach developed in [6], makes use of a recurrence relation found in the conditional probability equations, which were derived from the balance equations in the previous section. The basic idea is to generate new updates with the latest information available.

The first step in the modified iteration is to recognize that the conditional probabilities at a given iteration, i (denoted with a superscript), can be represented as:

$$p_n^i(S) \quad = \quad A_S p_n^i(III) + B_S u^i(n) + C_S$$

where A_S, B_S, and C_S are constants derived from the previous iteration. Using this observation, the definition of $u(n)$, and the conditionals' normalization constraint, the resulting recurrence relation determines the current iteration's values for $p_n^i(III)$ and $u^i(n)$. Therefore, the modified iterative method computes the i^{th} iteration for the general terms as follows:

$$\begin{aligned}
p_n^i(I)(\lambda+\alpha) &= p_{n-1}^i(I)u^i(n) + p_n^i(III)\gamma + \frac{\lambda}{u^{i-1}(n+1)}[p_{n+1}^{i-1}(I)\alpha rp + p_{n+1}^{i-1}(II)\beta p] \\
p_n^i(II)(\lambda+\beta) &= p_{n-1}^i(II)u^i(n) + p_n^{i-1}(\theta)\gamma + \frac{\lambda}{u^{i-1}(n+1)}[p_{n+1}^{i-1}(I)\alpha rq + p_{n+1}^{i-1}(II)\beta q] \\
p_n^i(III)(\lambda+\gamma) &= p_{n-1}^i(III)u^i(n) + \frac{\lambda}{u^{i-1}(n+1)}[p_{n+1}^{i-1}(I)\alpha sp + p_{n+1}^{i-1}(\theta)\beta p] \\
p_n^i(\theta)(\lambda+\beta+\gamma) &= p_{n-1}^i(\theta)u^i(n) + \frac{\lambda}{u^{i-1}(n+1)}[p_{n+1}^{i-1}(I)\alpha sq + p_{n+1}^{i-1}(\theta)\beta q]
\end{aligned}$$

with the boundary conditions following the same procedure. The extra cost of implementing this iterative method is in the added complexity of solving the recurrence relation on a per iteration basis. This cost and the resulting improvement in the number of iterations to reach convergence is presented in this chapter's results section.

2.1.3 Matrix-Geometric Approach

In this section, we apply another numerical method based on the matrix-geometric techniques, developed in [22, 53, 52]. Matrix-geometric solutions may exist for systems with state space descriptions of the form (i, j), where i and j are countable and only j is unbounded. Certainly, our state space description for the restricted overlap vacation model conforms to these constraints, *i.e.* for the (n, S) state space, n is integer valued and unbounded, and S is countable and bounded. Therefore, we investigate whether or not a matrix-geometric solution exists for the restricted overlap model and how the technique compares to the equivalence approach applied in the previous two sections.

The principal idea of the matrix-geometric method leverages the structural properties of the equations governing the system dynamics. The matrix-geometric property implies the existence of a matrix R such that

$$\vec{x}_{n+1} \quad = \quad R \cdot \vec{x}_n \ , \quad n \geq 0$$

where $\vec{x}_i = [p_{i0}, p_{i1}, \cdots, p_{ij}, \cdots, p_{iN}]^T$ is the steady state probability vector for i.

Considering our application, the balance equations from the previous sections can be represented as a matrix quadratic equation. The first step is recognize the restricted overlap balance equations can be expressed in matrix form as

$$A_0 \cdot \vec{p}_{n-1} + A_1 \cdot \vec{p}_n + A_2 \cdot \vec{p}_{n+1} \quad = \quad 0 \ , \quad n \geq 2$$

$$B_0 \cdot \vec{p}_0 + B_1 \cdot \vec{p}_1 + B_2 \cdot \vec{p}_2 = 0\,, \quad n = 1$$

$$C_0 \cdot \vec{p}_0 + C_1 \cdot \vec{p}_1 = 0\,, \quad n = 0$$

where the vector of state probabilities are defined as

$$\vec{p}_n = \begin{bmatrix} p_{nI} \\ p_{nII} \\ p_{nIII} \\ p_{n\theta} \end{bmatrix}$$

for $n \geq 1$ and

$$\vec{p}_n = \begin{bmatrix} p_{nIII} \\ p_{nI\bar{I}I} \end{bmatrix}$$

for $n = 0$. And, the matrices, $A_0, A_1, A_2, B_0, B_1, C_0$, and C_1 are defined as:

$$A_0 = \begin{bmatrix} \lambda & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & \lambda & 0 \\ 0 & 0 & 0 & \lambda \end{bmatrix}$$

$$A_1 = \begin{bmatrix} -(\lambda+\alpha) & 0 & \gamma & 0 \\ 0 & -(\lambda+\beta) & 0 & \gamma \\ 0 & 0 & -(\lambda+\gamma) & 0 \\ 0 & 0 & 0 & -(\lambda+\beta+\gamma) \end{bmatrix}$$

$$A_2 = \begin{bmatrix} \alpha rp & \beta p & 0 & 0 \\ \alpha rq & \beta q & 0 & 0 \\ \alpha sp & 0 & 0 & \beta p \\ \alpha sq & 0 & 0 & \beta q \end{bmatrix}$$

$$B_0 = \begin{bmatrix} 0 & \lambda p \\ 0 & \lambda q \\ \lambda p & 0 \\ \lambda q & 0 \end{bmatrix}$$

$$B_1 = A_1$$

$$B_2 = A_2$$

$$C_0 = \begin{bmatrix} \gamma & -\lambda \\ -(\lambda+\gamma) & 0 \end{bmatrix}$$

$$C_1 = \begin{bmatrix} \alpha r & \beta & 0 & 0 \\ \alpha s & 0 & 0 & \beta \end{bmatrix}$$

Next, applying the matrix-geometric property, we get the following form for the general equation:

$$A_0 R^{n-2} \cdot \vec{p}_2 + A_1 R^{n-1} \cdot \vec{p}_2 + A_2 R^n \cdot \vec{p}_2 \quad = \quad 0\,, \quad n \geq 2$$

Assuming the steady state solution exists and $\vec{p}_2$ is not identically zero, the equation above can be expressed as

$$A_0 + A_1 R + A_2 R^2 \quad = \quad 0$$

and can be solved numerically. We solve the matrix equation by applying the following recurrence:

$$R(i+1) \quad = \quad -(A_0 + A_2 R(i)^2) \cdot A_1^{-1}$$

With the solution to the matrix, R, and the two previous unused boundary equations, the corresponding probabilities are generated and thus the performance metrics can be determined.

As noted at the start of this chapter, it was our experience in developing a matrix-geometric solution to the initial model that the technique did not readily apply. Since we shifted to the vacation model definition, we did not further investigate what properties of the initial model definition resulted in the numerical difficulties.

The details of how the solution to the restricted overlap vacation model using the matrix-geometric method performs in comparison with the application of the equivalence solution methods are the topic of the next section.

2.1.4 Results

In the previous sections, three numerical solutions to the restricted overlap vacation model were presented. The first two were based on equivalence method and the third on matrix-geometric techniques. Now, we evaluate the accuracy and efficiency of these approaches. The efficiency of each approach is measured by the number of iterations to converge within a given accuracy and the number of arithmetic operations performed in the calculation. The following table outlines our findings with respect to the number of arithmetic operations:

In Table 2.1, n represents the depth at which the probabilities are truncated (n was chosen to be 40 in our examples). The table shows the modified method has a per iteration complexity close to double that of the simple iteration. The iteration complexity of the matrix-geometric method is written as the sum of two components: a factor which is dependent on n and a constant which is the number of arithmetic operations needed to

Method	$\pm$	$*$	$\div$
Simple Iteration	16n	20n	9n
Modified Iteration	36n	38n	19n
Matrix-Geometric	12n + 160	16n + 192	-

Table 2.1: Per Iteration Complexity

calculate $R(i)$. Given our choice of n, the matrix-geometric method has a per iteration complexity similar to the simple iteration method.

To comment on the efficiency of each method for a given set of parameters, we need to see how fast each method converges in combination with the arithmetic operation complexity of Table 2.1. In order to apply our model, we need to set the model parameters. At this point, we outline our methodology in selecting parameters. Our aim is choose a feasible set of parameters which represent current disk storage technology. We also want the model to experience overlap. Accordingly, we consider the operating specifications of Small Computer System Interface-2 (SCSI-2) [62] type devices, for example see [28], which currently offer a comparable capacity and performance to the general class of mainframe direct access storage devices (DASD, "disks" in IBM parlance), for example see [31, 34]. Note our choice of parameters are not meant to reflect any particular disk vendor but rather a set of feasible

values which reflect the current technology.

The main parameters we utilize to derive our model parameters are the transfer speed, the rotational speed, and the orientation time. We assume the disk transfer rate is 5 MB/sec and the rotational speed is 5400 rpm (which implies a full track transfer of 11.11 msec). We also assume a fixed orientation time of 6 msec. A cache transfer rate of 10 MB/sec is assumed. An overhead time of 1 msec and 0.7 msec is also assumed for the transfer setup of a block from the physical device and the cache, respectively. Therefore, given transfer block size, BLK, we set the averages a, b, and c as:

$$\begin{aligned} a &= 1 + 6 + BLK/5 \text{ msec} \\ b &= 0.7 + BLK/10 \text{ msec} \\ c &= 11.11 - BLK/5 \text{ msec} \end{aligned}$$

Note c is set to a full track minus the desired record transfer time. The staging is typically from the desired record to the end of the current track. However, we include a full track to account for any asynchronous background activity due to operations like write-backs from NVS.

The branching probabilities, p and s, are determined from three factors:

- Read to write ratio
- Miss probability
- Cache write policy

Throughout this chapter, we assume a 3:1 ratio of read to write requests. The miss probability is varied between 10% to 40% to demonstrate its effect on the response time. The cache write policy is assumed to be write-back, *e.g.* in IBM mainframe class disk controllers [33, 2], writes can be treated as DASD Fast Writes [33]. The write-back policy

implies all write requests experience the service of stage II and we agglomerate the time to do the asynchronous transfer to disk into stage III, as mentioned previously. Thus, the parameter, p, can be determined as the product of the miss probability and the probability of a read, $\frac{3}{4}$ in our examples.

Since the write policy is write-back and all read requests are followed by a staging period, the branching probability s is set to one. We consider this parameter setting as a reasonable mode of operation. It also increases the possibility of overlap by maximizing the hidden activity represented by stage III. If one wanted to model a write policy of write-through, where a write request only required a physical disk for the duration of a block transfer, the probability s could be set accordingly. We limit our attention to $s = 1$ in the following examples.

Now that we have outlined our assumptions for the values of the model parameters, we can generate a set of results from the three solution techniques. Table 2.2 shows the number of iterations needed for each method to reach convergence within a given error limit, ε. We define the error as the maximum absolute value of the difference between $p_n^i(S)$ and $p_n^{i+1}(S)$ over all n and S as defined in the previous sections.

Table 2.2 demonstrates the effectiveness of the modified method in reducing the number of iterations to convergence versus the simple method. The matrix-geometric method is also shown to be a more efficient solution than the simple method for the results in Table 2.2. Note, the efficiency of the matrix-geometric method versus the modified method is not as clear as the comparison with the simple method. For the purpose of this dissertation, we adopt the modified method as a solution technique for the subsequent extensions of the restricted overlap model. We leave the further investigation between the modified method and the matrix-geometric solution as an open problem.

Parameters	Method		
$\lambda, \bar{a}, \bar{b}, \bar{c}, p, s, \varepsilon$	Simple	Modified	M-G
$0.1, 7.78, 1.09, 10.33, 0.30, 1.0, 10^{-4}$	59	9	19
$0.1, 7.78, 1.09, 10.33, 0.22, 1.0, 10^{-4}$	68	10	18
$0.1, 7.78, 1.09, 10.33, 0.15, 1.0, 10^{-4}$	72	10	18
$0.1, 7.78, 1.09, 10.33, 0.07, 1.0, 10^{-4}$	76	10	16
$0.1, 8.56, 1.48, 9.55, 0.30, 1.0, 10^{-4}$	54	12	21
$0.1, 8.56, 1.48, 9.55, 0.22, 1.0, 10^{-4}$	60	9	18
$0.1, 8.56, 1.48, 9.55, 0.15, 1.0, 10^{-4}$	64	10	18
$0.1, 8.56, 1.48, 9.55, 0.07, 1.0, 10^{-4}$	66	10	10
$0.1, 9.34, 1.87, 8.77, 0.30, 1.0, 10^{-4}$	42	11	18
$0.1, 9.34, 1.87, 8.77, 0.22, 1.0, 10^{-4}$	50	10	19
$0.1, 9.34, 1.87, 8.77, 0.15, 1.0, 10^{-4}$	52	10	16
$0.1, 9.34, 1.87, 8.77, 0.07, 1.0, 10^{-4}$	52	13	15

Table 2.2: Iteration Efficiency

In Figure 2.5, Figure 2.6, and Figure 2.7, we present the results of our analytic solution and simulation at several test points. As discussed in the introduction, the simulation confidence level was 95% and the corresponding confidence halfwidths are shown around each of the simulation points. The three graphs illustrate the performance, in terms of average response time, for block transfer lengths of 4KBytes, 8KBytes, and 12KBytes. Each graph has four plots; one for each value of the branching probability, p ($p = 0.07, 0.15, 0.22, 0.30$ as derived from the previously discussion). The analytical results and simulation points appear to cross validate for the examples shown. Simulation points were run for every other analytical point with the analytical result falling within the simulation confidence interval.

In the next section, we relax the exponential constraint on the distribution of the service stages.

2.2 Restricted Overlap Vacation Model under General Assumptions

2.2.1 Coxian Equivalence

In an effort to extend the applicability of the exact solution to systems with service times of general distributions, each server in the model can be represented by a two stage coxian equivalent distribution [17]. The two stage coxian matches the first two moments of a general distribution with a coefficient of variation, C_v, greater than $1/\sqrt{2}$. The coxian equivalent model is shown in Figure 2.8. We consider a state space description similar to the exponential case, (n, S), with an expanded number of service states, $S \in \{I_i, II_i, III_i, \theta_{ij}\}$ where $i, j \in \{1, 2\}$ corresponding to the two coxian stages. Note the number of service states increased to $3 \cdot 2 + 2^2 = 10$ and, generally, a k stage coxian would result in $3 \cdot k + k^2$ states. Thus, the state space grows quite rapidly for a resulting representation of C_v's greater than

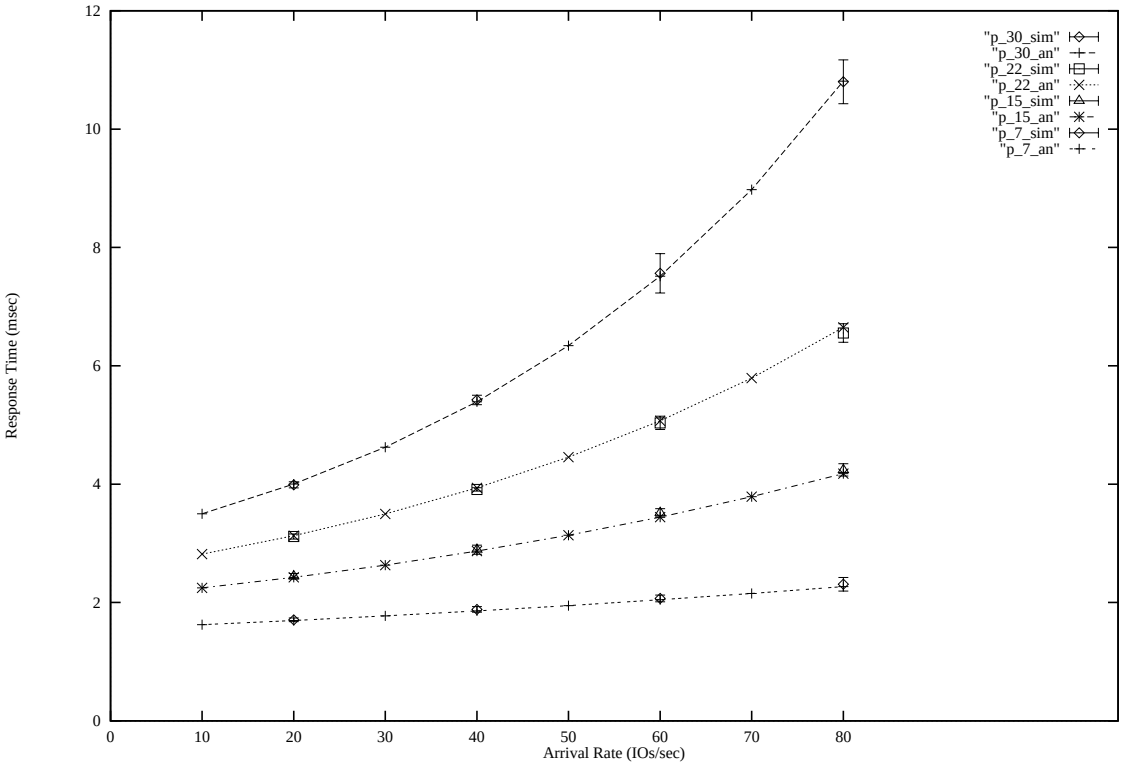

Figure 2.5: Logical Device Performance with 4KByte Block Transfers, Simulation and Analytical Results for p=0.07,0.15,0.22,0.30

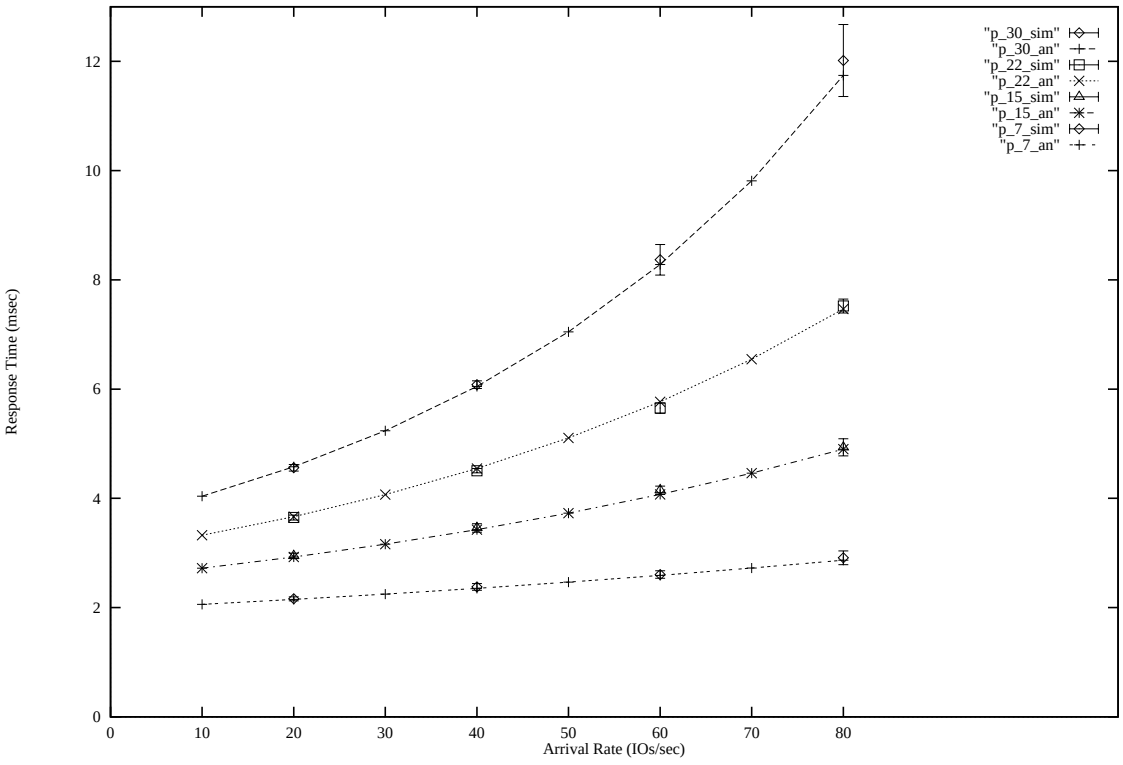

Figure 2.6: Logical Device Performance with 8KByte Block Transfers, Simulation and Analytical Results for p=0.07,0.15,0.22,0.30

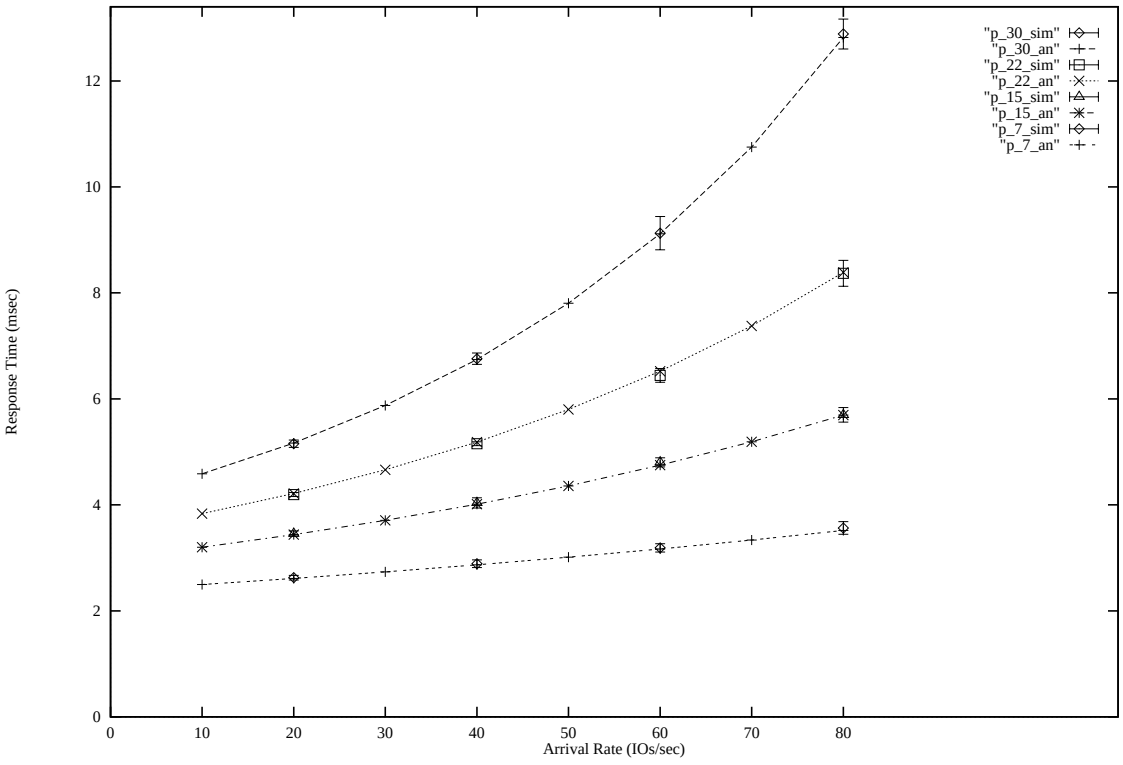

Figure 2.7: Logical Device Performance with 12KByte Block Transfers, Simulation and Analytical Results for p=0.07,0.15,0.22,0.30

$1/\sqrt{k}$, *i.e.* the law of diminishing returns becomes evident in representing low variability distributions.

The two stage coxian is constructed from two exponential stages. Service begins with the first exponential stage and the second stage is entered with some branching probability. The service rates and branching probability can be determined such that the first two moments match the desired general distribution. This leads to the service states, S, described above, where each exclusive (non-overlapped) server is in one of the two coxian stages and the overlapped service must account for all combinations of overlap, *i.e.* $II_1 + III_1 \Rightarrow \theta_{11}$, $II_1 + III_2 \Rightarrow \theta_{12}$, $II_2 + III_1 \Rightarrow \theta_{21}$, and $II_2 + III_2 \Rightarrow \theta_{22}$.

A new, expanded set of balance equations for the restricted overlap model can written for the general case ($n \geq 2$) as follows:

$$
\begin{aligned}
p(n,I_1)(\lambda+\alpha_1) &= p(n-1,I_1)\lambda + p(n+1,I_1)\alpha_1 rp(1-l_a) + p(n+1,I_2)\alpha_2 rp \\
&\quad + p(n+1,II_1)\beta_1 p(1-l_b) + p(n+1,II_2)\beta_2 p \\
&\quad + p(n,III_1)\gamma_1(1-l_c) + p(n,III_2)\gamma_2 \\
p(n,I_2)(\lambda+\alpha_2) &= p(n-1,I_2)\lambda + p(n,I_1)\alpha_1 l_a \\
p(n,II_1)(\lambda+\beta_1) &= p(n-1,II)\lambda + p(n+1,I_1)\alpha_1 rq(1-l_a) + p(n+1,I_2)\alpha_2 rq \\
&\quad + p(n+1,II_1)\beta_1 q(1-l_b) + p(n+1,II_2)\beta_2 q \\
&\quad + p(n,\theta_{11})\gamma_1(1-l_c)p(n,\theta_{12})\gamma_2 \\
p(n,II_2)(\lambda+\beta_2) &= p(n-1,II_2)\lambda + p(n,II_1)\beta_1 l_b \\
p(n,III_1)(\lambda+\gamma_1) &= p(n-1,III_1)\lambda + p(n+1,I_1)\alpha_1 sp(1-l_a) + p(n+1,I_2)\alpha_2 sp \\
&\quad + p(n+1,\theta_{11})\beta_1 p(1-l_b) + p(n+1,\theta_{21})\beta_2 p \\
p(n,III_2)(\lambda+\gamma_1) &= p(n-1,III_2)\lambda + p(n,III_1)\gamma_1 l_c + p(n+1,\theta_{12})\beta_1(1-l_b)p \\
&\quad + p(n+1,\theta_{22})\beta_2 p
\end{aligned}
$$

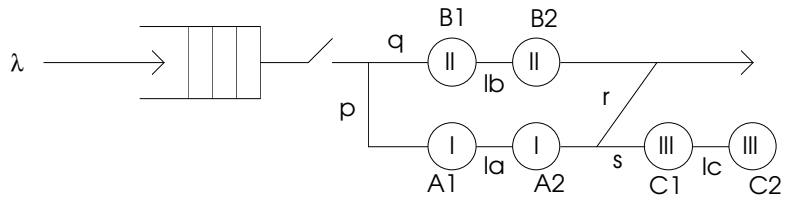

Figure 2.8: Queueing model of restricted overlap with general service represented by two stage coxian distribution

$$
\begin{aligned}
p(n,\theta_{11})(\lambda+\beta_1+\gamma_1) &= p(n-1,\theta_{11})\lambda + p(n+1,I_1)\alpha_1(1-l_a)sq + p(n+1,I_2)\alpha_2 sq \\
&\quad + p(n+1,\theta_{11})\beta_1(1-l_b)q + p(n+1,\theta_{21})\beta_2 q \\
p(n,\theta_{12})(\lambda+\beta_1+\gamma_2) &= p(n-1,\theta_{12})\lambda + p(n,\theta_{11})\gamma_1 l_c \\
p(n,\theta_{21})(\lambda+\beta_2+\gamma_1) &= p(n-1,\theta_{21})\lambda + p(n,\theta_{11})\beta_1 l_b \\
p(n,\theta_{22})(\lambda+\beta_2+\gamma_2) &= p(n-1,\theta_{22})\lambda + p(n,\theta_{12})\beta_1 l_b + p(n,\theta_{21})\gamma_1 l_c
\end{aligned}
$$

The corresponding boundary conditions follow the same extension.

The coxian equivalence approach appears to be an accurate representation of general distributions as shown in the following results section. However, as previously discussed, the applicability of this approach is limited by the variability which can be represented. In the following section, we consider an approximation technique which attempts to account for general distributions of service including lower variability. The method is an adjustment of the results generated from the model based on exponential assumptions. Thus, it does not suffer from an increase in the state space description. We outline the distributional adjustment approach in the next section. Then, we present some results on the accuracy of these two general variability approaches.

2.2.2 Distributional Adjustment

A useful approximation of queueing systems with general distributions is the application of a distributional adjustment, analogous to the M/G/1 solution. The idea is to adjust the average queueing time, calculated under exponential assumptions, by a factor which accounts for the variability. This technique, outlined in [1], is based on empirical observations of Allen and Cunneen, however [1] recognizes its formulation has been used by others.

We start by assuming the existence of an underlying "exponential form" for the average waiting time, $\bar{W}_q$, of the restricted overlap solution. Next, we express the average waiting time for the restricted overlap model with exponentially distributed service stages and with generally distributed service stages. Both are derived as an *M/G/1 like* adjustment of $\bar{W}_q$ where the adjustment is a function of the coefficient of variation of the service time. The calculation is as follows:

$$\begin{aligned} \bar{W}_{q_{exponential}} &= \bar{W}_q \cdot \frac{1 + C^2_{v_{exponential}}}{2} \quad \text{and} \\ \bar{W}_{q_{general}} &= \bar{W}_q \cdot \frac{1 + C^2_{v_{general}}}{2} \\ &\Rightarrow \quad \bar{W}_{q_{general}} = \bar{W}_{q_{exponential}} \cdot \frac{1 + C^2_{v_{general}}}{1 + C^2_{v_{exponential}}} \end{aligned}$$

Since $\bar{W}_{q_{exponential}}$ can be generated using the methods previously developed, we can approximate $\bar{W}_{q_{general}}$ as shown above. The accuracy of this method is explored in the next section.

2.2.3 Results

In this section, we outline some of our findings for the restricted overlap model under general distributional assumptions for the service stages. In Figure 2.9, Figure 2.10, and Figure 2.11, we demonstrate the accuracy of the coxian equivalence and distributional adjustment approaches. In each of the figures, the x-axis represents the variability in the distribution of stage I. The performance is measured in terms of the average response time. The four plots on each graph correspond to a varying arrival rate: 20, 40, 60, and 80 IOs/sec.

For the given runs, the coxian method and simulation results cross validate within the coxian method's region of applicability, $C_v \in \{1, 2, 3\}$. The distributional adjustment method also appears to be effective in capturing the performance behavior for most of the

coefficient of variation points plotted (including the constant case). For high variability and high arrival rates, the adjustment method slightly overestimates the response times as compared to the simulation. However, we have accurate coverage of these cases with the coxian approach. Note, the appeal of the distributional adjustment technique over the coxian equivalence method is the relatively simpler computational complexity. The "base" value for the distributional adjustment method is generated using the restricted overlap model under exponential assumptions. Thus, it is more efficient since the state space is effectively $\frac{2}{5}^{ths}$ the size of the coxian equivalence.

2.3 Shared Devices

In many computer installations, it is quite common for processor complexes to share storage devices (as well as other peripheral devices), see Figure 2.12. Sharing disks is a useful means to provide common data to multiple processor complexes and, as an ancillary effect, increase their utilization. However, performance may be adversely affected if higher utilization leads to larger *pending* times. An I/O request experiences pending delays when the shared device appears free to the issuing system, but service cannot start. Some typical causes for pending delays arise when no pathes are available to the desired device or the disk is busy on behalf of another CPU complex. We focus on the the effect of the latter as it applies to the restricted overlap model.

A queueing model, which accounts for pending time under exponential assumptions, was presented in [11]. A distribution for the pending time experienced by a single complex due to "extraneous" activity is generated. We adopt a similar approach in order to represent pending delays in the restricted overlap vacation model in a shared environment. In the restricted overlap model, we account for the pending times in a new stage, referred to as

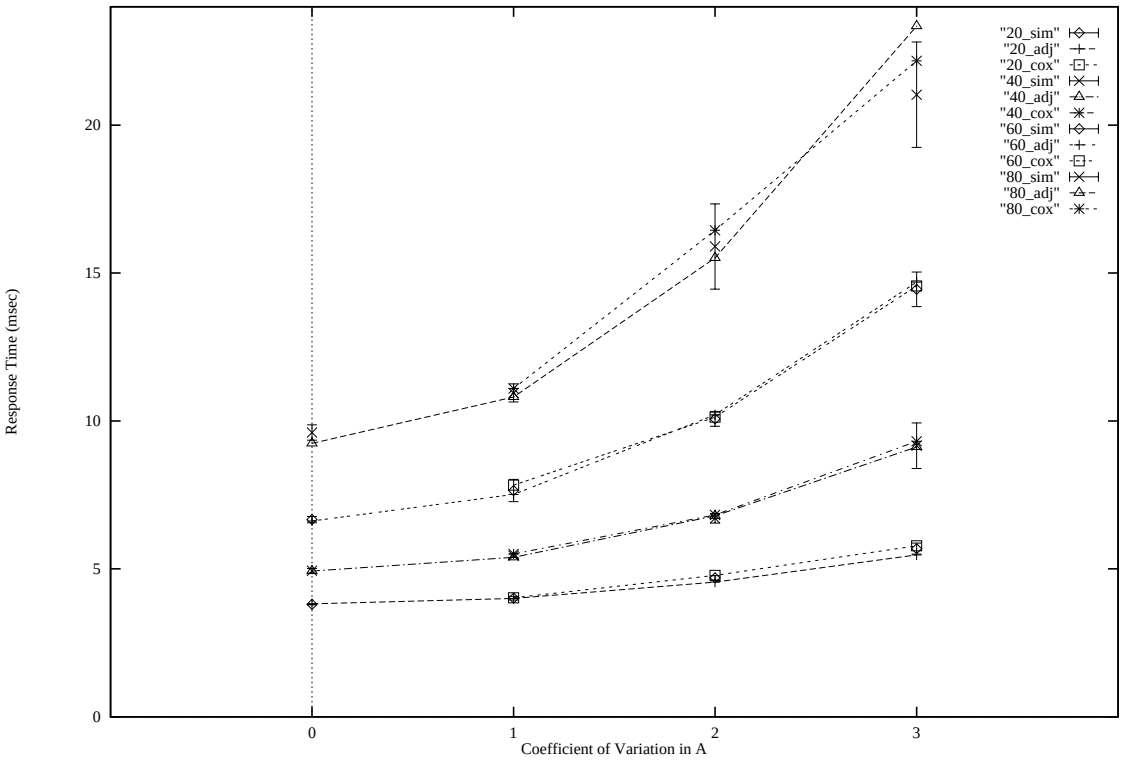

Figure 2.9: Logical Device Performance with 4KByte Block Transfers and Variability in Stage I, Simulation and Analytical Results for p=0.07,0.15,0.22,0.30

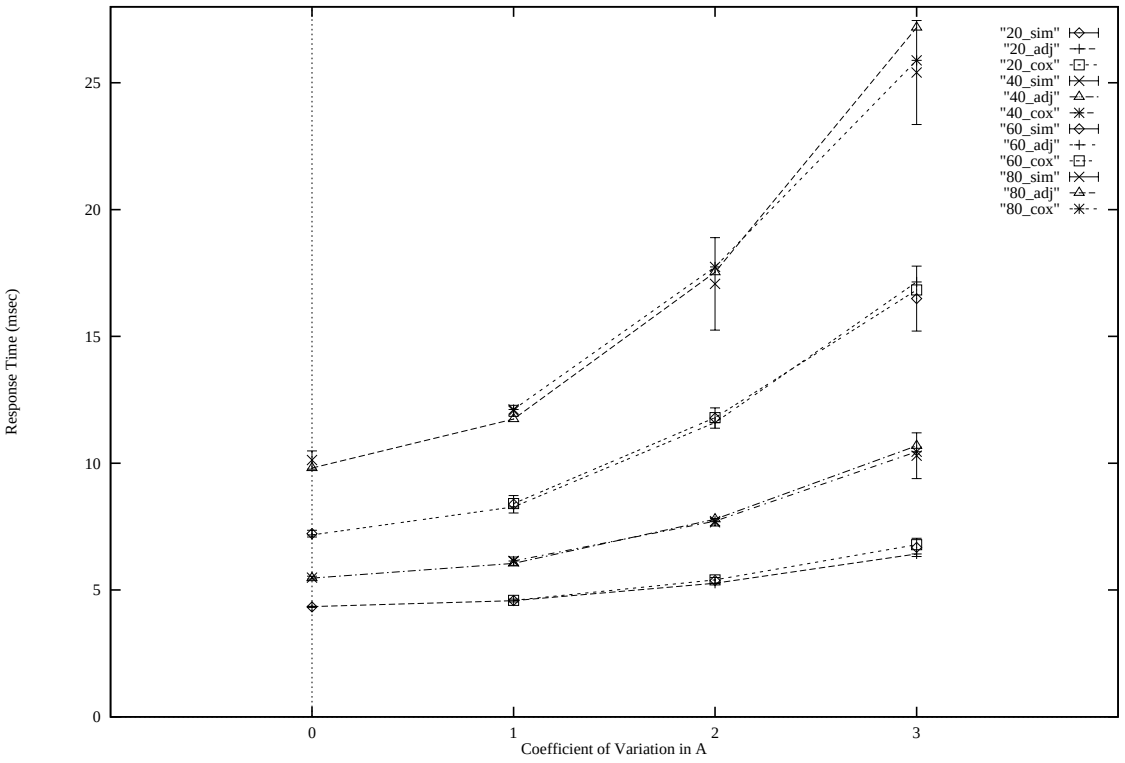

Figure 2.10: Logical Device Performance with 8KByte Block Transfers and Variability in Stage I, Simulation and Analytical Results for p=0.07,0.15,0.22,0.30

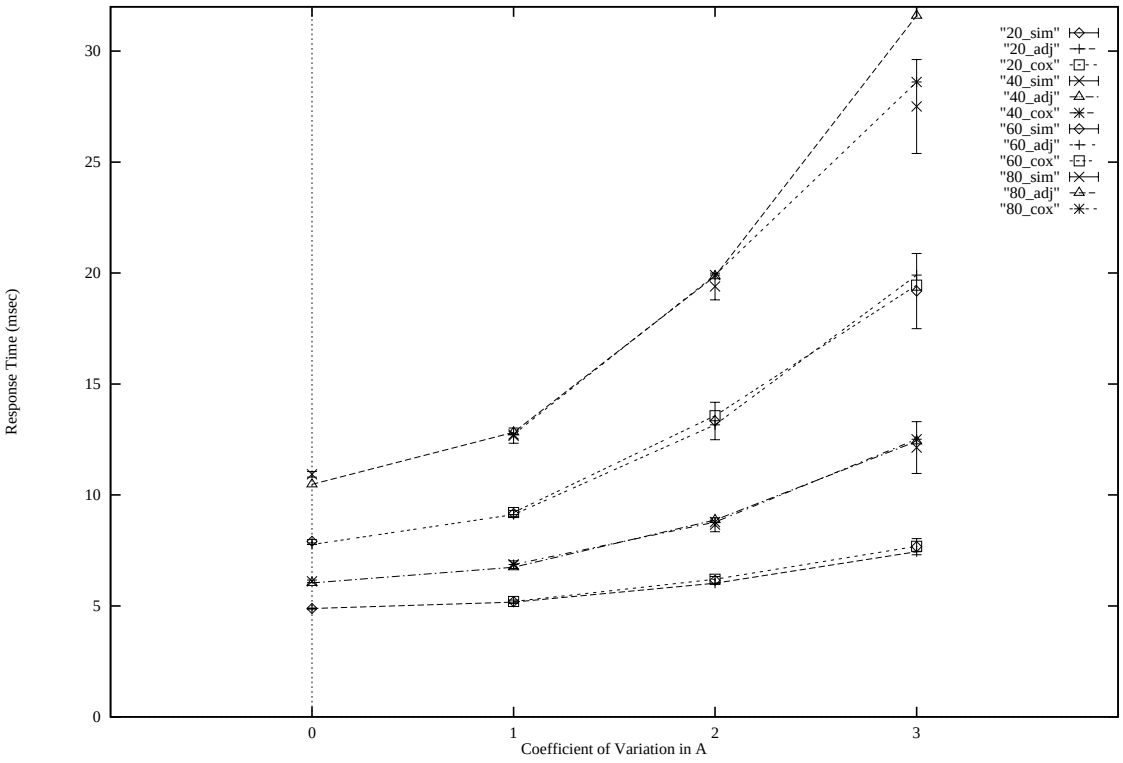

Figure 2.11: Logical Device Performance with 12KByte Block Transfers and Variability in Stage I, Simulation and Analytical Results for p=0.07,0.15,0.22,0.30

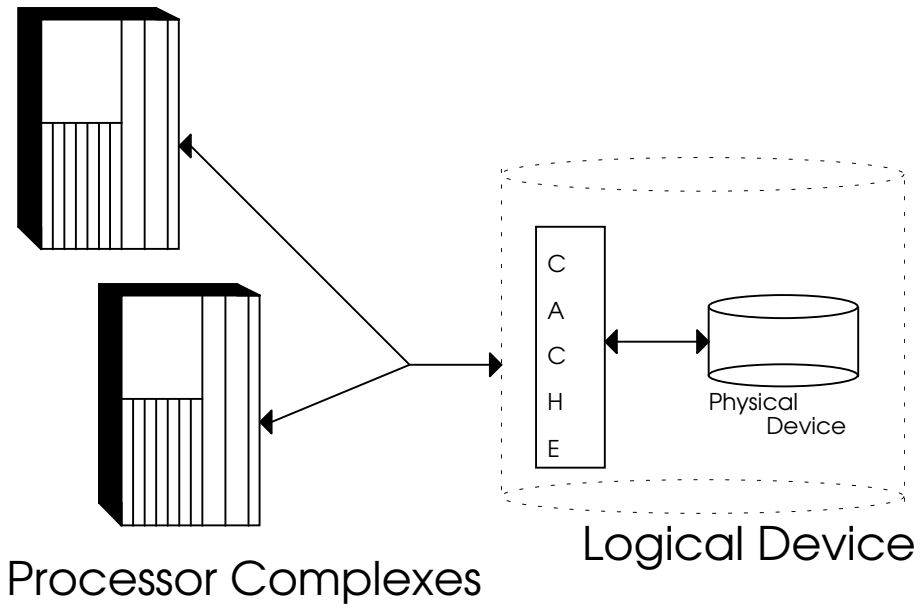

Figure 2.12: Diagram of shared storage subsystem

stage IV, prior to stage I see Figure 2.13.

The two stage coxian is constructed from two exponential stages. We calculate the distribution of stage IV as the residual of the physical device busy time $(I + III)$ times the probability the device is found busy. As a heuristic, the physical device busy time is treated as an exponential to facilitate the residual calculation.

Figure 2.14, Figure 2.15, and Figure 2.16 show the effectiveness of this approach. Each graph has four curves corresponding to the branching probability, $p = 0.07, 0.15, 0.22$ and 0.30. The method is quite accurate for higher cache hit rates, *i.e.* $p = 0.07$ and 0.15. As the probability of a request requiring a physical device increases and the arrival rate increases, the approximation suffers from a slight overestimate. Since the approximation falls within the simulation confidence intervals for the 4KByte block transfer size and begins to overestimate for the 8KByte and the 12KByte blocks, the slight exaggeration may be due to a distributional characteristic of stage IV which is lost with the exponential assumption. However, the model does capture the performance trend in average response times and we leave the resolution of the small differences as an open problem.

In the next chapter, we draw on the techniques developed in this chapter and extend them in order to represent a disk array storage architecture.

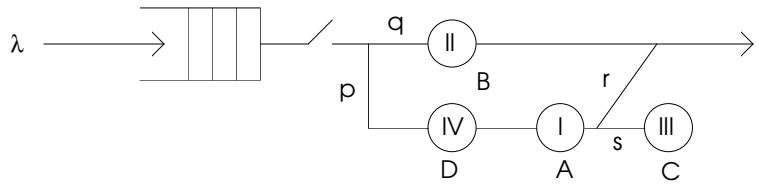

Figure 2.13: Queueing model of restricted overlap with pending time stage due to sharing

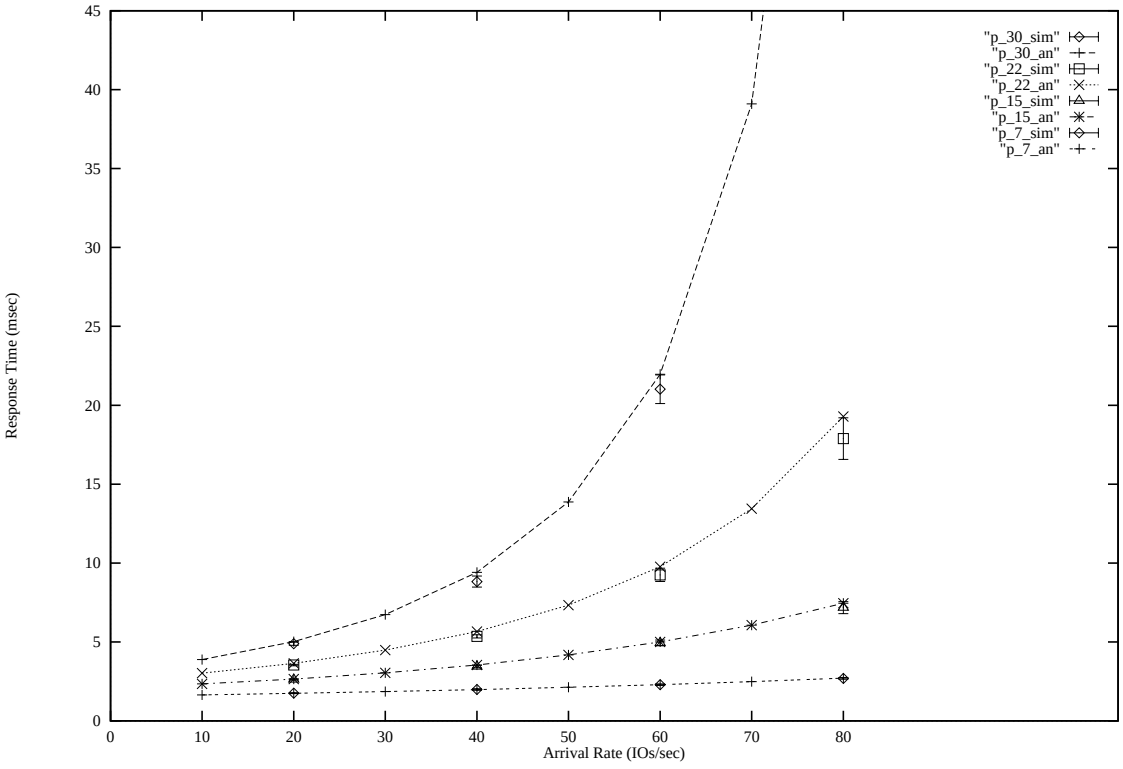

Figure 2.14: Shared Device Performance with 4KByte Block Transfers, Simulation and Analytical Results for p=0.07,0.15,0.22,0.30

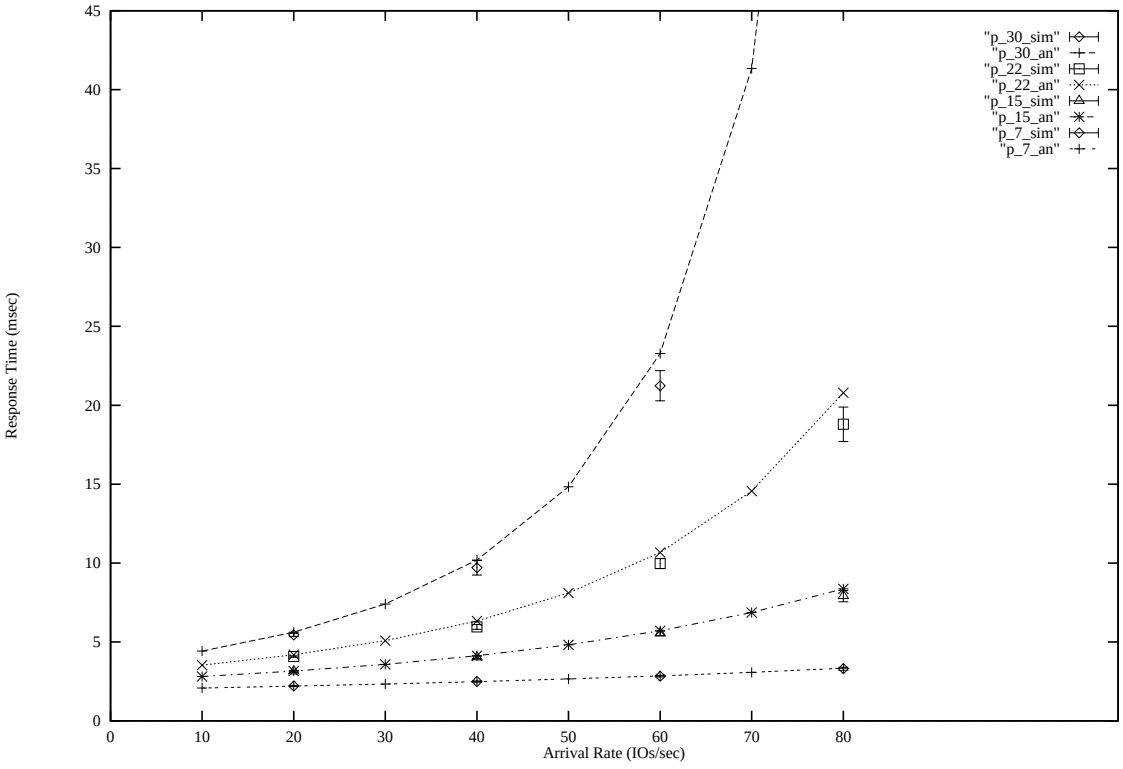

Figure 2.15: Shared Device Performance with 8KByte Block Transfers, Simulation and Analytical Results for p=0.07,0.15,0.22,0.30

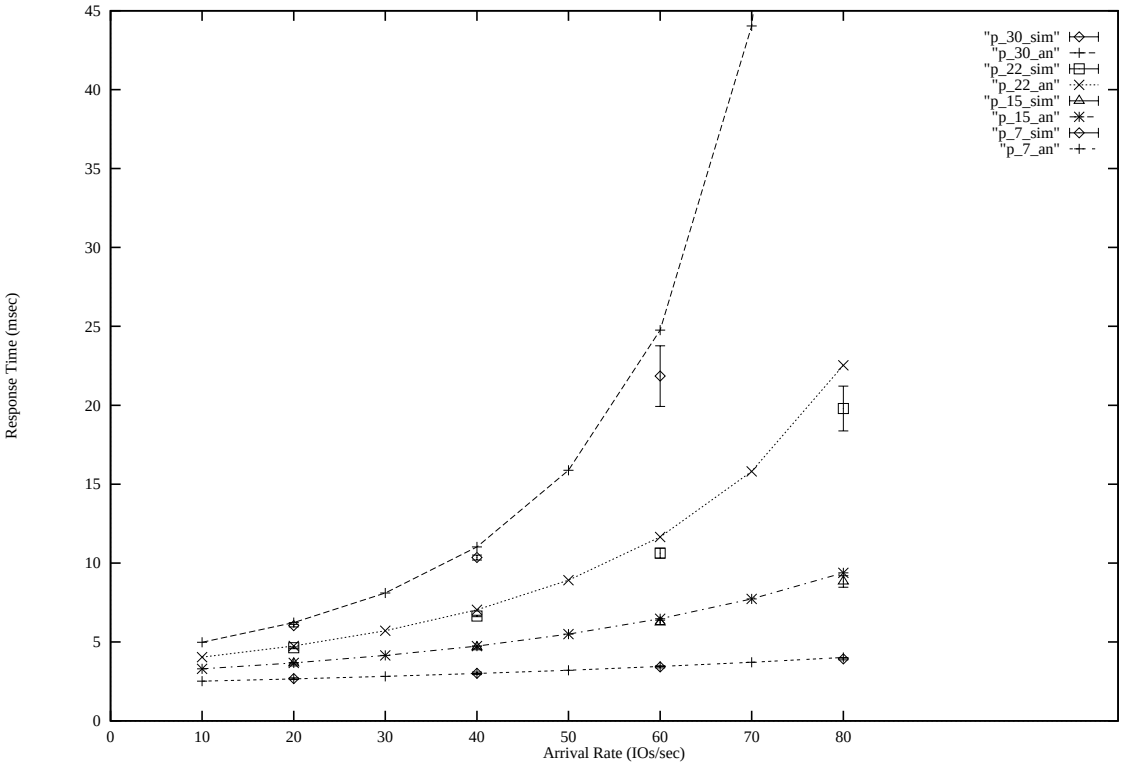

Figure 2.16: Shared Device Performance with 12KByte Block Transfers, Simulation and Analytical Results for p=0.07,0.15,0.22,0.30

3. Modeling Disk Array Performance

In this chapter, we focus on extending the restricted overlap model to represent disk arrays utilizing a cached controller. A disk array is an I/O subsystem architecture which organizes a group of physical disks into a set of logical devices by distributing data across the disks. The technique of striping or interleaving data over multiple disks [39, 59, 46] emerged as a solution to the increasing speed gap between secondary storage and processors. The RAID (*Redundant Array of Inexpensive Disks*) architectures [38, 55, 56] further popularized (and stimulated the commercial interest in) disk arrays by utilizing many small disks, organized for performance and reliability.

The need for tools and techniques which facilitate the understanding of the behavior of complex systems, such as disk arrays, increases as these systems become more readily available. Performance models can assist in the design, configuration, and procurement of disk arrays. As expressed in the introduction chapter, analytic models are typically preferable to other techniques because of efficient solution times, allowing multiple configuration scenarios to be evaluated.

3.1 Background

The RAID architectures as presented in [38] are categorized into five levels, RAID-1 through RAID-5. Each level positions the array construct at a different point in the characterization space of cost, capacity, performance, and reliability. A complete taxonomy of the RAID levels can be found in [38] and we briefly outline the levels as follows:

- *RAID-1* - mirroring complete disks. Dual copies of each disk are maintained.

- *RAID-2* - Hamming error code detection and correction. Data is bit-interleaved across a group of data disks and check disks.
- *RAID-3* - single check disk. Bit-interleaving of data, similar to RAID-2, with a single check disk. By reducing the number of check disks to one, single faults can be tolerated by reconstructing data via parity.
- *RAID-4* - dedicated parity disk. The interleaving of data across disks is conducted with a higher level of granularity than RAID3. This allows independent block transfers from multiple disk.
- *RAID-5* - rotated parity. Data and parity are striped across disks. This alleviates the single disk bottleneck of writes experienced by RAID-4.

Since the introduction of RAID, the marketplace and literature has grown richly with various enhancements and alternative methods of developing disk array architectures. Some examples are the parity striping architecture [27], EMC's Integrated Cached Disk Array (ICDA) [54], the TickerTAIP parallel RAID architecture [14], Network Appliance's NFS File Server Appliance [29], and Iceberg from Storage Technology.

In this chapter, the analysis focuses on a RAID-5 level disk array with a cached controller. We extend the idea of restricted overlap to account for the multiple spindles which comprise a RAID-5 device.

3.2 Related Work

The advancement of disk arrays has spurred the development of performance evaluation models. Previous work on analytic performance models of disk arrays, including RAID-5, have not accounted for the presence of a cached controller [43, 72, 15] or ignored the overlapped service which exists between the cache and data staging [49].

The model presented in [43] uses an approach based on fork-join queueing [3]. The model is a closed queueing system in which requests are generated by a set of processes; each process generating a request upon the completion of the previous request. This type of model is a better representation of scientific and time-sharing systems. As in the previous chapter, our approach considers an open queueing model, a closer representation to a transaction based environment.

A performance model of RAID-5 which accounts for the problem of write synchronization (the old data must be read in order to calculate parity) is demonstrated in [15]. The model maintains two request queues for each disk: one for read/write requests and one for parity requests. Upon the start of any write request at the head of the read/write queue, a parity request is generated. The parity queue has higher priority to ensure that the parity request is serviced as soon as possible since the corresponding write has already started. We assume the presence of a controller with sufficient non-volatile cache storage to take advantage of the fast write I/O operation and thus deemphasize the write problem of the non-cached environment modeled in [15].

The model of RAID-5 in [72] utilizes a vacation server model to specifically analyze the disk array in the presence of faults, *i.e.* the periods of time a disk is unavailable due to failure. It represents three modes of operation: normal, degraded (single drive fault requiring data to be reconstructed from parity), and rebuild (complete spare drive is reconstructed to regain normal operation). The focus of the model is on an accurate representation of the various modes of operation of a RAID-5 device and does not account for a cached controller or the periods of hidden activity.

In [49], the authors consider a model that accounts for a disk array with a cached controller. However, it is not clear how one might extend the model to represent the

overlapped service of cache hits and staging of track information which our model tries to capture. This model as well as the related work discussed to this point also tend to limit their representation of disk arrays to a single logical device.

An overview of how the current hard disk drive technology has shifted the original RAID hypothesis of performance via the "I/O bandwidth" of many actuators to accenting the need for a cached array is presented in [12]. In the next section, we extend the restricted overlap idea to model the behavior of a RAID-5 with a cached controller in an "MVS like" operating environment.

3.3 RAID-5 Model - Single Logical Device Representation

Our approach to modeling RAID-5 is to consider the kernel model presented in chapter 2 and extend the idea of restricted overlap in order to represent the added complexity of a logical device consisting of multiple physical devices. Figure 3.1 shows an example of a RAID-5 construct. For the purpose of our analysis, we assume a mapping of a large enough granularity, *e.g* cylinder level, to restrict staging to a single physical drive. We start by assuming the entire RAID-5 device forms a single logical device in order to test the accuracy of our model in a controlled case where all requests are on behalf of the same logical device, *i.e.* no interference.

The queueing model shown in Figure 3.2 is used to model the performance of a RAID-5. As with the "classical" I/O model, the service of the desired data record is represented by stage I and stage II represents the cache service time. However, with a disk array, we now have two main new features:

1. requests requiring service from a physical device are directed at one of D physical devices (where the RAID-5 device consists of D drives)

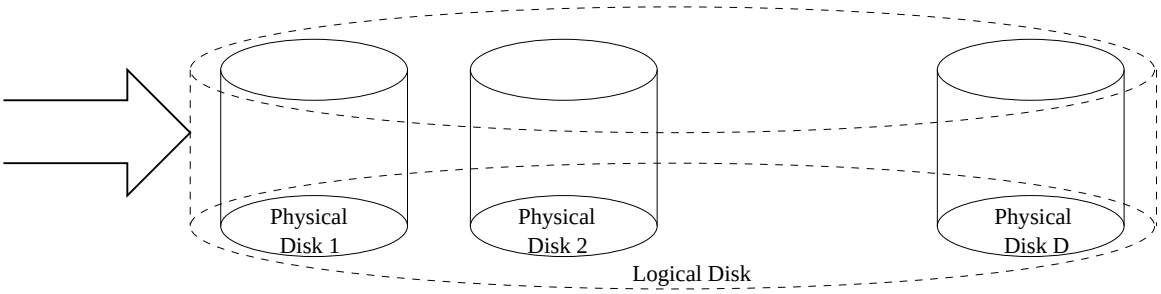

Figure 3.1: Schematic figure of RAID-5 device

2. requests requiring service from a physical device may also *overlap* with the staging of physical devices other than the one corresponding to the desired record

This observation leads to a state description of (n, S) with the variable n representing the number of customers in the system and new set of service states $S \in \{I, II, III_i, \theta_i, \zeta_j\}$ where

$$I \triangleq \text{server station I is busy exclusively}$$

$$II \triangleq \text{server station II is busy exclusively}$$

$$III_i \triangleq \text{server station(s) IIIi is (are) busy exclusively}$$

$$\theta_i \triangleq \text{server stations II and IIIi are both busy (overlap).}$$

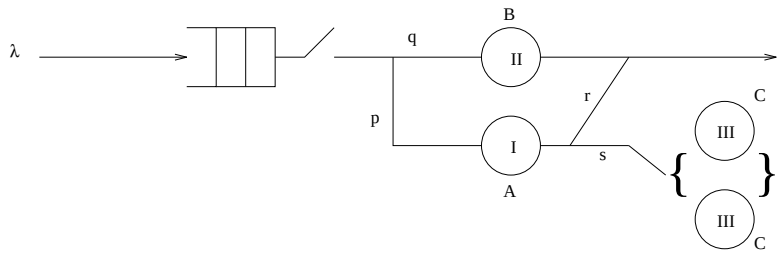

Figure 3.2: RAID-5 overlap queueing model representation

$\zeta_j \triangleq$ server stations I and IIIj are both busy (new overlap)

and $i \in \{1, \cdots, D\}$, $j \in \{1, \cdots, D-1\}$.

At this point, we note that the state description can be fairly large even for a small number of disks in the array. Therefore, instead of developing a full set of balance equations for D drives, we interject an approximation by limiting $i, j \in \{1, 2\}$. Experimental evidence indicates this approximation results in an accurate estimate of the I/O response times. The approximation accuracy will be discussed further after presenting the equations governing our queueing model.

Using the above state space definition, the balance equations can be expressed as:

$$\begin{aligned} p(n, I)(\lambda + \alpha) &= p(n-1, I)\lambda + p(n, III_1)\gamma + p(n+1, I)\alpha rp \\ &\quad + p(n+1, II)\beta p + p(n, \zeta_1)\gamma \end{aligned}$$

$$
\begin{aligned}
p(n,II)(\lambda+\beta) &= p(n-1,II)\lambda + p(n,\theta_1)\gamma + p(n+1,I)\alpha rq + p(n+1,II)\beta q \\
p(n,III_1)(\lambda+\gamma) &= p(n-1,III_1)\lambda + p(n+1,I)\alpha s\frac{p}{D} + p(n+1,\theta_1)\beta\frac{p}{D} \\
&\quad + p(n+1,\zeta_1)\alpha r\frac{p}{D} + p(n,III_2)2\gamma \\
p(n,\theta_1)(\lambda+\beta+\gamma) &= p(n-1,\theta_1)\lambda + p(n+1,\zeta_1)\alpha rq + p(n+1,I)\alpha sq \\
&\quad + p(n+1,\theta_1)\beta q + p(n,\theta_2)2\gamma \\
p(n,\zeta_1)(\lambda+\alpha+\gamma) &= p(n-1,\zeta_1)\lambda + p(n+1,I)\alpha sp\frac{D-1}{D} + p(n+1,\theta_1)\beta p\frac{D-1}{D} \\
&\quad + p(n+1,\zeta_1)\alpha rp\frac{D-1}{D} + p(n,\zeta_2)2\gamma \\
p(n,III_2)(\lambda+2\gamma) &= p(n-1,III_2)\lambda + p(n+1,\theta_2)\beta\frac{2p}{D} \\
&\quad + p(n+1,\zeta_1)\alpha s\frac{2p}{D} + p(n+1,\zeta_2)\alpha\frac{2p}{D} \\
p(n,\theta_2)(\lambda+\beta+2\gamma) &= p(n-1,\theta_2)\lambda + p(n+1,\theta_2)\beta q + p(n+1,\zeta_1)\alpha sq + p(n+1,\zeta_2)\alpha q \\
p(n,\zeta_2)(\lambda+\alpha+2\gamma) &= p(n-1,\zeta_2)\lambda + p(n+1,\theta_2)\beta p\frac{D-2}{D} \\
&\quad + p(n+1,\zeta_1)\alpha sp\frac{D-2}{D} + p(n+1,\zeta_2)\alpha p\frac{D-2}{D}
\end{aligned}
$$

where $n \geq 2$ and the boundary conditions can be written as

$$
\begin{aligned}
p(0,I\bar{I}I)\lambda &= p(1,I)\alpha r + p(1,II)\beta + p(0,III_1)\gamma \\
p(0,III_1)(\lambda+\gamma) &= p(1,I)\alpha s + p(1,\theta_1)\beta + p(1,\zeta_1)\alpha r + p(0,III_2)2\gamma \\
p(0,III_2)(\lambda+2\gamma) &= p(1,\zeta_1)\alpha s + p(1,\zeta_2)\alpha + p(1,\theta_2)\beta
\end{aligned}
$$

$$
\begin{aligned}
p(1,I)(\lambda+\alpha) &= p(0,I\bar{I}I)\lambda p + p(1,III_1)\gamma + p(2,I)\alpha rp \\
&\quad + p(2,II)\beta p + p(1,\zeta_1)\gamma \\
p(1,II)(\lambda+\beta) &= p(0,I\bar{I}I)\lambda q + p(1,\theta_1)\gamma + p(2,I)\alpha rq + p(2,II)\beta q \\
p(1,III_1)(\lambda+\gamma) &= p(0,III_1)\lambda\frac{p}{D} + p(2,I)\alpha s\frac{p}{D} + p(2,\theta_1)\beta\frac{p}{D} \\
&\quad + p(2,\zeta_1)\alpha r\frac{p}{D} + p(1,III_2)2\gamma
\end{aligned}
$$

$$
\begin{aligned}
p(1,\theta_1)(\lambda+\beta+\gamma) &= p(0,III_1)\lambda q + p(2,\zeta_1)\alpha r q + p(2,I)\alpha s q \\
&\quad + p(2,\theta_1)\beta q + p(1,\theta_2)2\gamma \\
p(1,\zeta_1)(\lambda+\alpha+\gamma) &= p(0,III_1)\lambda p\frac{D-1}{D} + p(2,I)\alpha s p\frac{D-1}{D} + p(2,\theta_1)\beta p\frac{D-1}{D} \\
&\quad + p(2,\zeta_1)\alpha r p\frac{D-1}{D} + p(1,\zeta_2)2\gamma \\
p(1,III_2)(\lambda+2\gamma) &= p(0,III_2)\lambda\frac{2p}{D} + p(2,\theta_2)\beta\frac{2p}{D} + p(2,\zeta_1)\alpha s\frac{2p}{D} + p(2,\zeta_2)\alpha\frac{2p}{D} \\
p(1,\theta_2)(\lambda+\beta+2\gamma) &= p(0,III_2)\lambda q + p(2,\theta_2)\beta q + p(2,\zeta_1)\alpha s q + p(2,\zeta_2)\alpha q \\
p(1,\zeta_2)(\lambda+\alpha+2\gamma) &= p(0,III_2)\lambda p\frac{D-2}{D} + p(2,\theta_2)\beta p\frac{D-2}{D} \\
&\quad + p(2,\zeta_1)\alpha s p\frac{D-2}{D} + p(2,\zeta_2)\alpha p\frac{D-2}{D}
\end{aligned}
$$

As in Chapter 2, we apply a semi-numerical technique based on equivalence methods to solve for the probabilities and ultimately derive the performance measures of interest. The accuracy of this model is outlined in the next section.

3.3.1 Results

In this section, we outline the accuracy of the RAID-5 model treated as a single logical device. Figures 3.3, 3.4, and 3.5 show the results of our approximation versus simulation. In these graphs, we consider a disk array consisting of six physical devices. The model parameters are set to the same values as those in Chapter 2. We also assume a uniform access pattern across the disk array. Note "hot spots", in which the disk utilizations are skewed, can be represented by skewing the branching probabilities.

The curves on each graph are generated for varying probability, $p = 0.07, 0.15, 0.22$ and 0.30. Note that, in the examples shown, truncating the multi-server station at stage III to two stations is an effective approximation of the disk array performance, compared with the simulation. For each of the examples, the simulation results demonstrated the number

of active devices in the array was two or less for 98% of the simulation run time.

In order to gain some more insight into the applicability of the two station approximation, we considered an upper bound on the number of simultaneously vacationing stage III server stations. We use a classical multi-server station queue with Poisson arrival rate, λp. The type q requests are ignored since they are serviced in parallel. Also, the stage I time is ignored to facilitate our analysis. We consider this a valid assumption since we are interested in bounding, from above, the number of simultaneously active staging stations and the stage I time only decreases the parallel activity by allowing an opportunity for any of the active devices to complete staging.

Now, we can use the solution to a multi-server queue [1] to solve for the probability of n in the system:

$$p_n = \frac{(\lambda pc)^n}{n!} \cdot p_0$$

for $D \geq n > 1$ and

$$p_n = \frac{(\lambda pc)^n}{D^{i-D} D!} \cdot p_0$$

for $n > D$ and where

$$p_0 = [\sum_{i=0}^{D-1} \frac{(\lambda pc)^i}{i!} + \frac{(\lambda pc)^D}{D!(1 - \lambda pc/D)}]^{-1}$$

Note for our parameter choices, p_n is a rapidly decreasing function. From an engineering perspective, potential trouble spots (in which the two station approximation may no longer be effective) are avoided based on the following constraints on the parameter space:

- practical arrival rates $\Rightarrow$ constrains λ from growing too large
- cache designs target relatively high hit rates to justify addition to the architecture $\Rightarrow$ constrains p from increasing too high

- increases in the average staging period, c, are constrained since trends in drive transfer rates are increasing

Thus, given the feasible cache hit rates and the physical disk busy period distributions, the two station approximation appears to be reasonable. One can design a choice of parameters which forces more simultaneous activity among the array devices. However, our experimentation shows such parameter choices leads to saturation and unreasonable response times.

3.4 RAID-5 Model - Multiple Logical Devices Representation

Typically, commercial RAID-5 configurations organize the array of physical disks as a set of multiple logical devices, see Figure 3.6. Each logical device of the RAID-5 architecture presents the image of a "standard" device to the operating system. Thus, the underlying disk organization is transparent to the processor complex. This architecture differs from the previous section since a request for a particular logical device may now find the device busy on behalf of another logical device which maps to the same physical device. The delay is an external pending time similar to the delay described in Chapter 2 for shared devices by multiple processor complexes.

Since an I/O request for a given logical device, which maps to a particular physical disk, may be delayed due to activity on the same physical disk on behalf of another logical device, we introduce a new stage IV, similar to the shared devices case of Chapter 2. Stage IV, described with a distribution, D, is added to account for this pending time. Determining D is not a straightforward task since one must evaluate the distribution of a residual based on the activity of a physical device due to requests being serviced by the other logical devices. We avoid this complexity by approximating the distribution, D, as the residual of

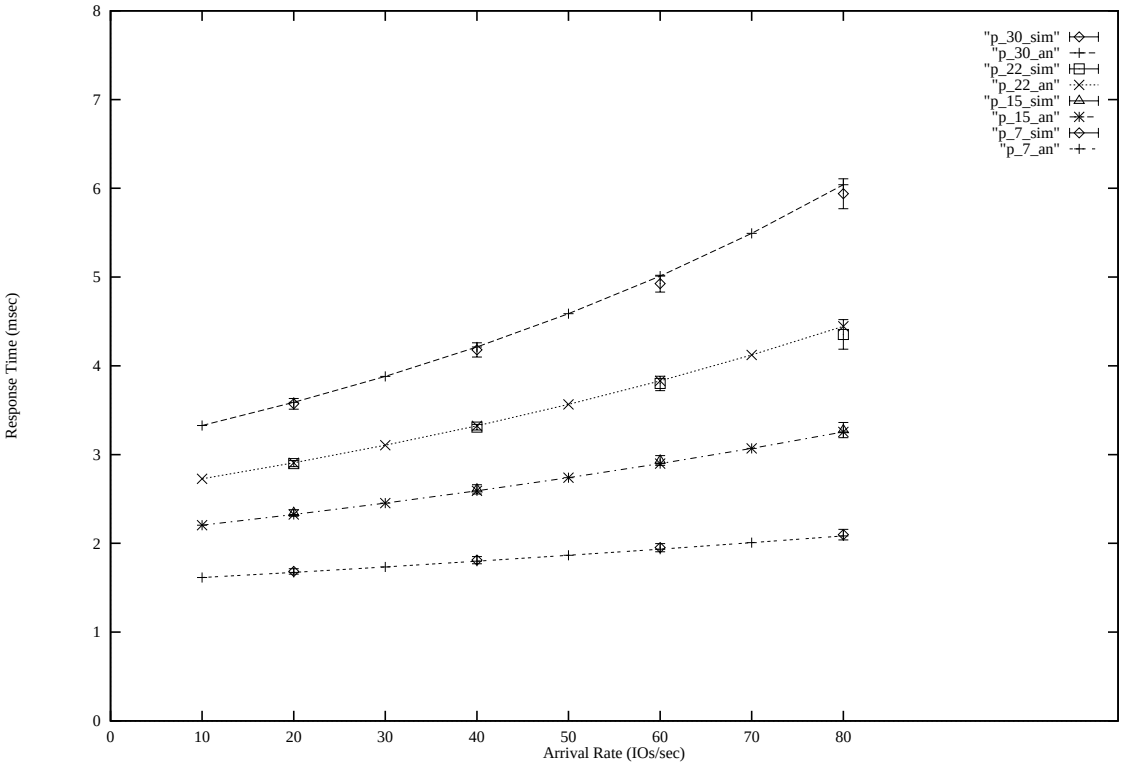

Figure 3.3: 4K block requests of RAID-5 (1 ldevs, 6 physical disks)

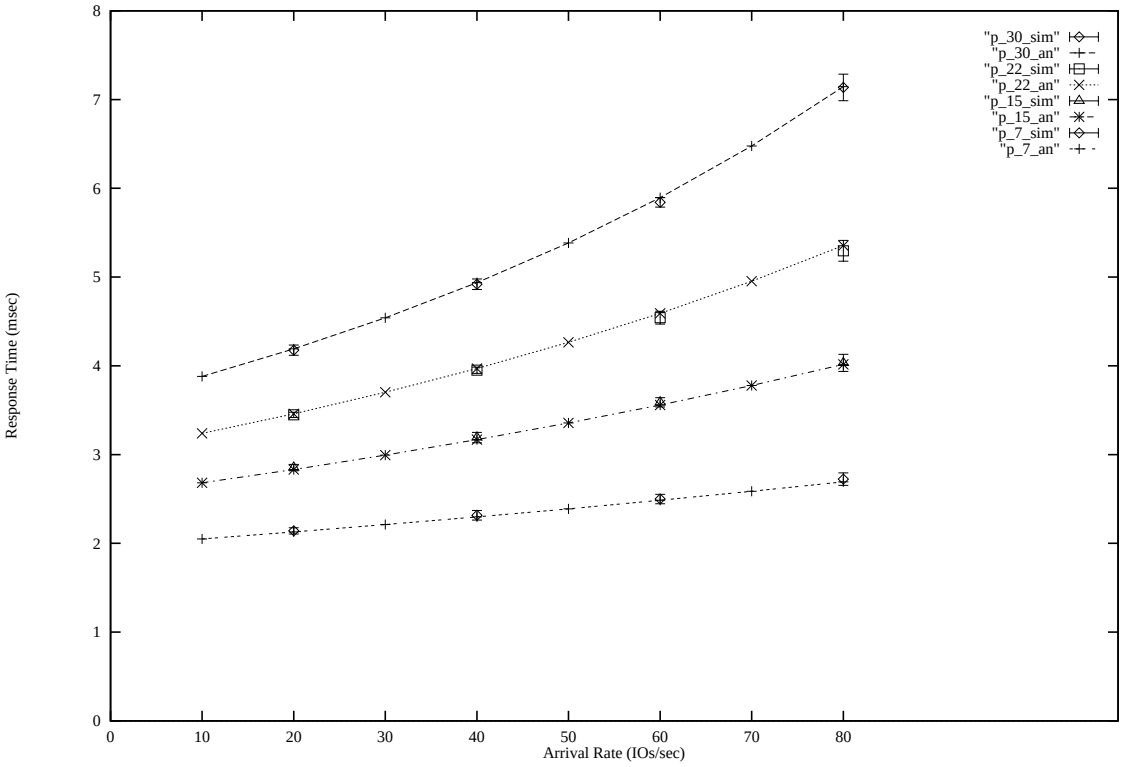

Figure 3.4: 8K block requests of RAID-5 (1 ldevs, 6 physical disks)

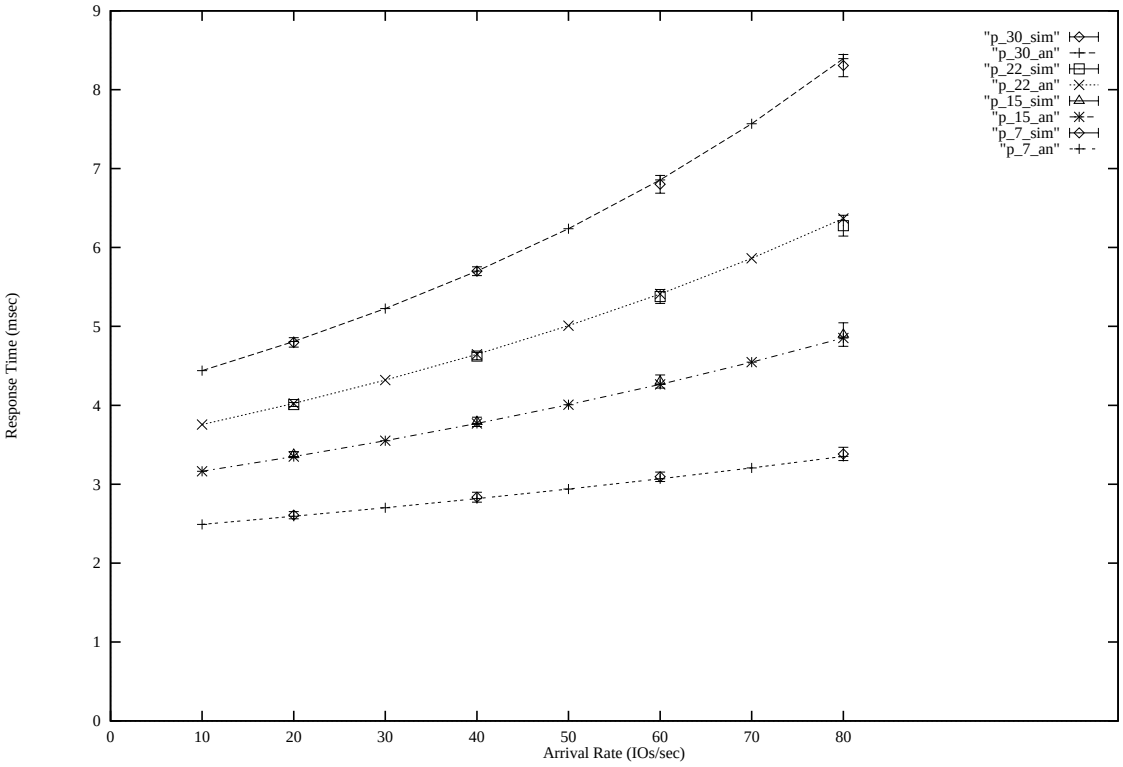

Figure 3.5: 12K block requests of RAID-5 (1 ldevs, 6 physical disks)

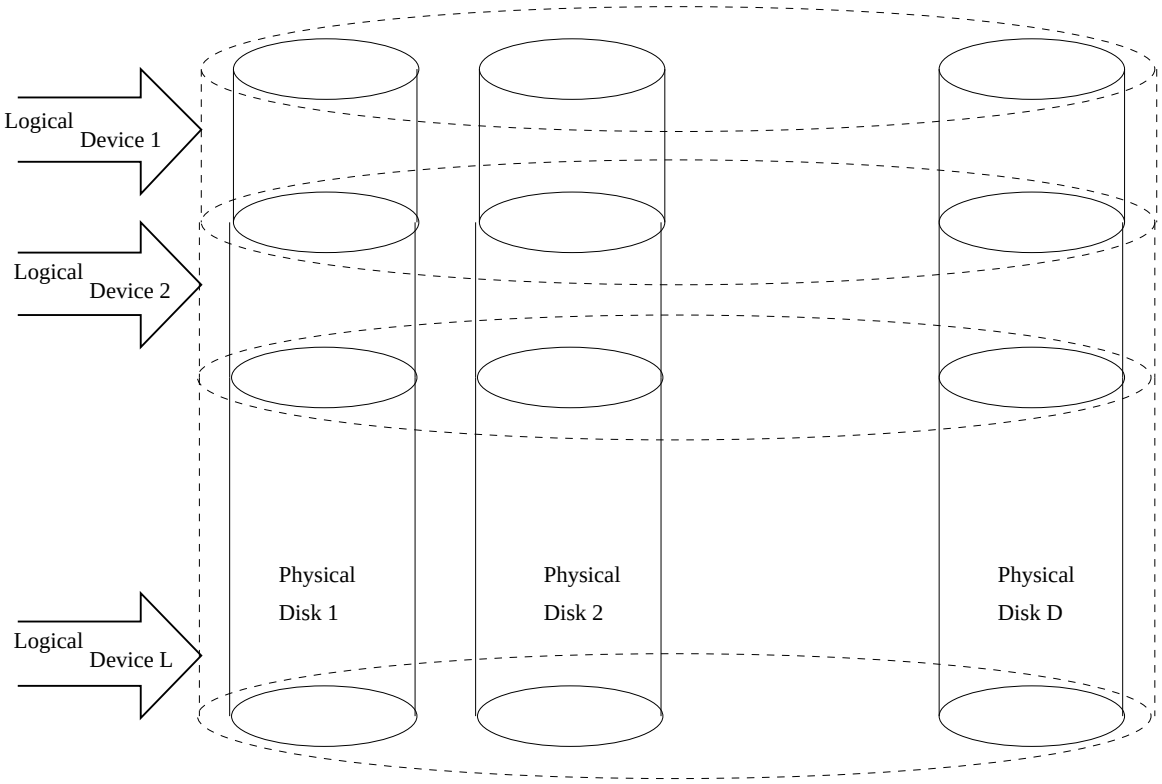

Figure 3.6: RAID-5 Device Partitioned into Multiple Logical Devices

an exponential generated using the device busy time and the probability the device is found busy. The queueing model is shown in Figure 3.7.

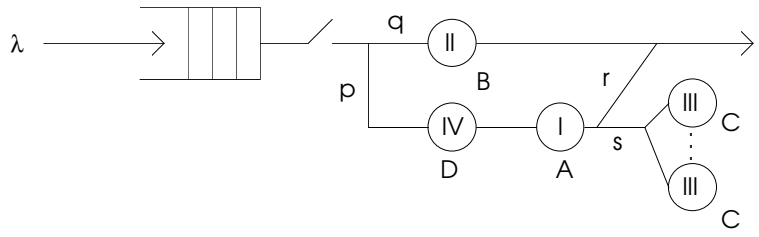

Figure 3.7: Queueing model of a RAID-5 device, accounting for multiple logical devices

3.4.1 Results

In this section, we outline the accuracy of the RAID-5 model partitioned into multiple logical devices. As in the previous section, we consider a disk array with six disks. We also assume the RAID-5 device is organized as six logical devices. Figures 3.8, 3.9, and 3.10 show the results of our approximation versus simulation. The approximation accurately follows the performance trends, falling within the confidence intervals.

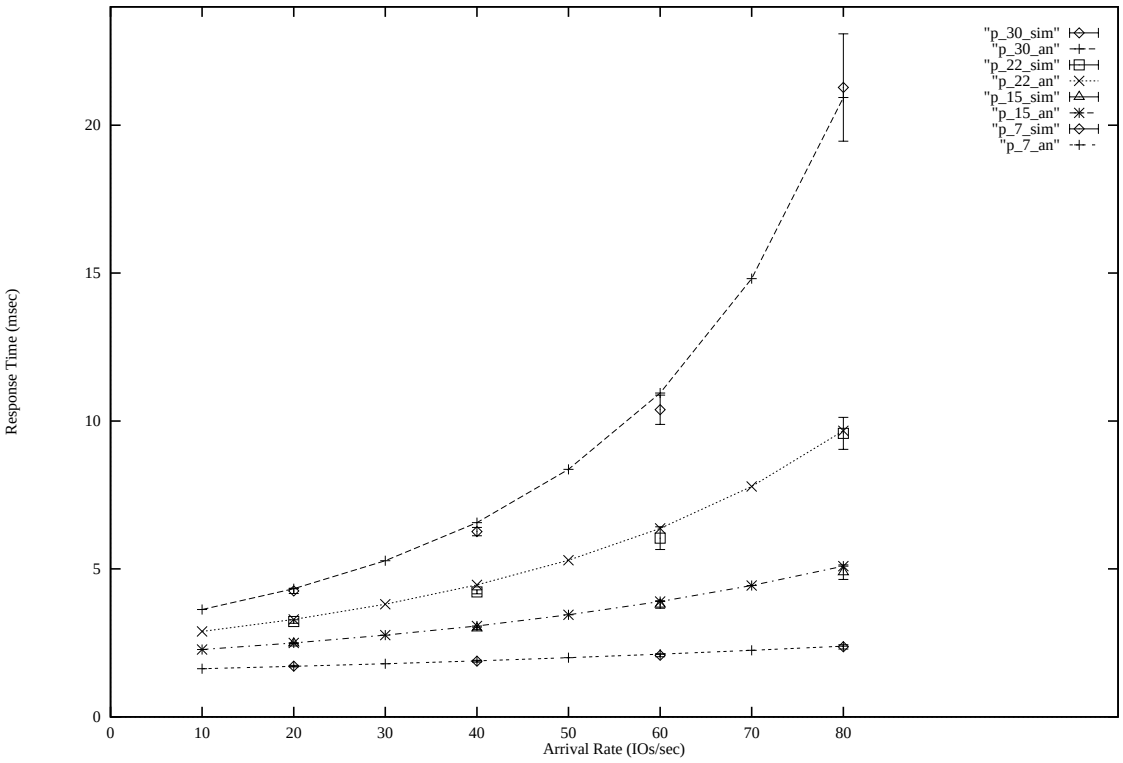

Figure 3.8: 4K block requests of RAID-5 (6 ldevs, 6 physical disks)

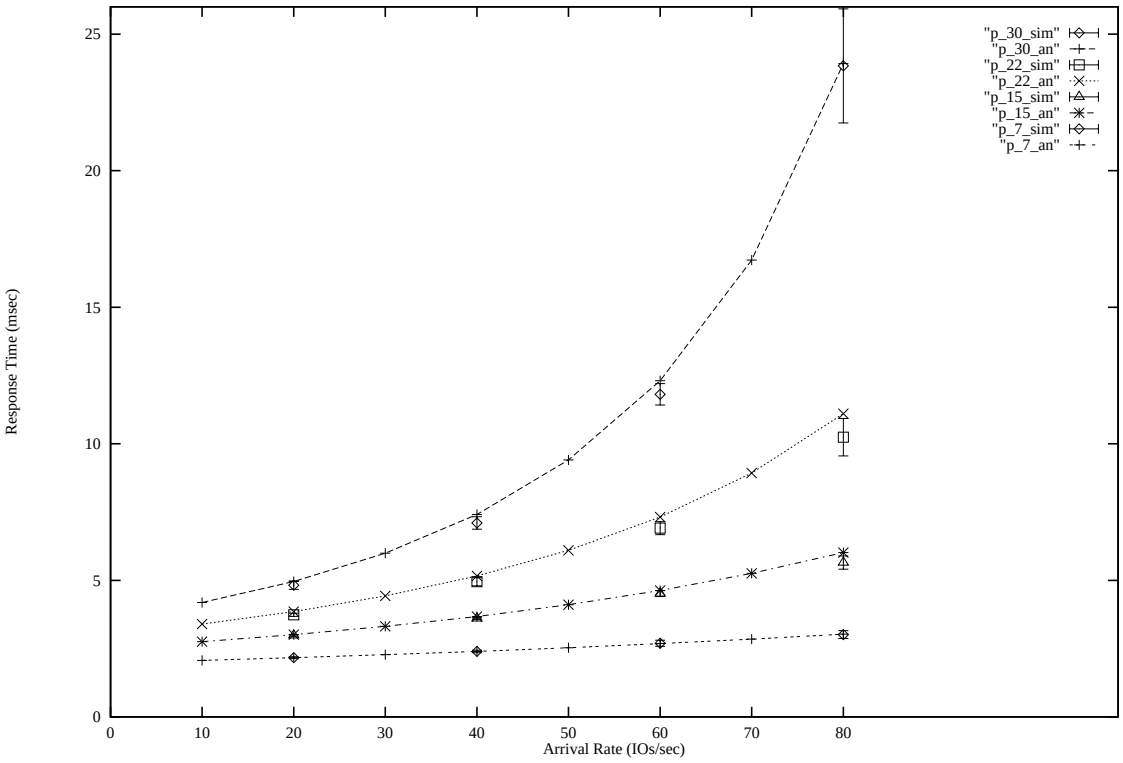

Figure 3.9: 8K block requests of RAID-5 (6 ldevs, 6 physical disks)

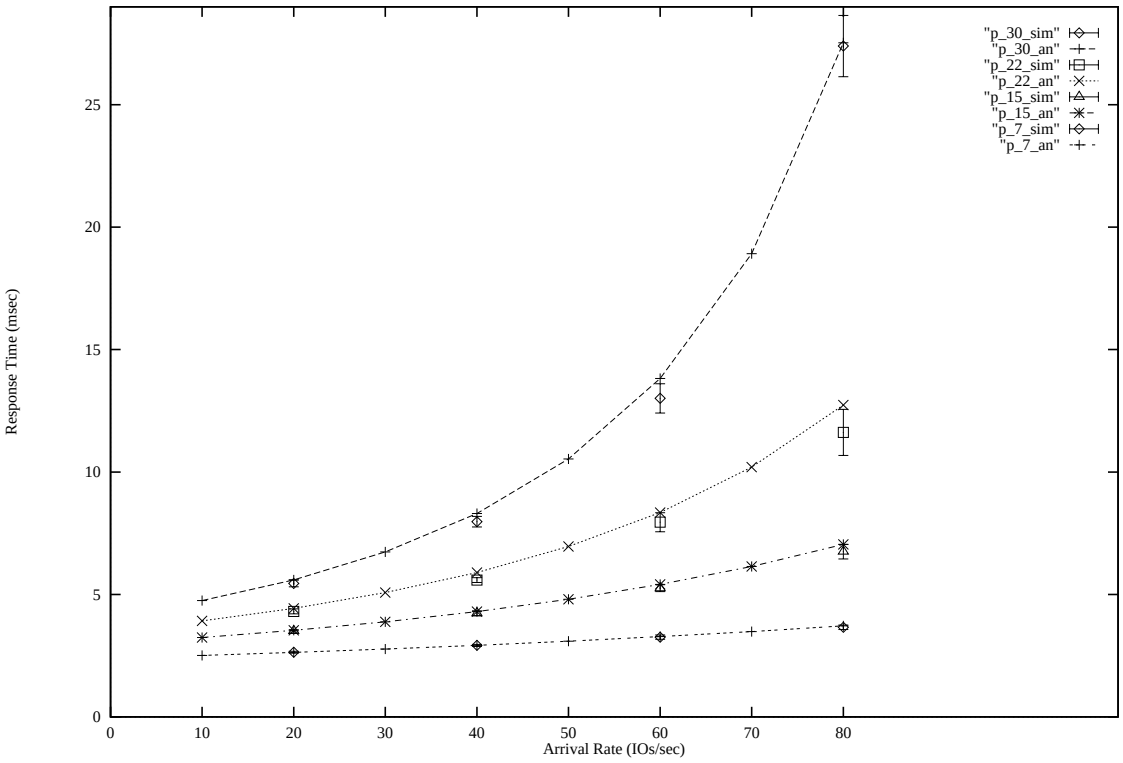

Figure 3.10: 12K block requests of RAID-5 (6 ldevs, 6 physical disks)

4. Multimedia Kiosks and Preliminary Sequence Caching

Audio-video information-dispensing computers, known as *kiosks*, are becoming a popular technology for businesses and government agencies. Kiosks offer individuals a convenient means of information retrieval while reducing the demands placed on service organizations by automating the answering of routine questions. Current computer kiosks have a broad domain of utility, from generating a jobs listing in a California state office to pavilion previews to one stop shopping. The success of multimedia will be driven by cost-effective, widespread applications, and kiosks provide users with a sample of the information technology which will eventually become commonplace at work and in the home.

Currently, kiosks are autonomous stations typically using analog signals for the presentation of audio and video information. This format is effective, however, it does not harness the potential of a digital format. In this chapter, we consider multimedia kiosk objects as digital data, managed by an electronic library. The system architecture is distributed, where multimedia stations are networked in an application group and each group is connected to the electronic library, see Figure 4.1. Thus, the management of the entire set of kiosks and multimedia objects are supported by the electronic library. For example, to update an application group of kiosks with a modified video sequence, a 'librarian' could complete the modification from a single electronic library client (workstation). In the autonomous, analog kiosks, a technician would have to visit the site of every station in order to physically update its media.

Although the distributed architecture facilitates kiosk management, it suffers from the common multimedia bottlenecks associated with storage performance and network bandwidth requirements. These problems are especially prevalent since each kiosk user expects

real-time response at the stations. The current economics associated with large storage requirements and real-time delivery constraints of continuous media objects prohibits storing a local copy of all data at each multimedia kiosk. Therefore, some effort must be expended on developing storage techniques [65].

Rangan and Vin [57] have developed an effective technique for designing file systems which support continuous media storage and retrieval. They also present a method for multiple media stream maintenance which is complimentary to the development of kiosk groups, as described in the next section. Little and Ghafoor discuss spatial and temporal composition of multimedia objects in an distributed framework [45]. The approach presents a general taxonomy of distributed multimedia information systems which subsumes our architecture by virtue of its generality. However, our goal is to provide some insight into the performance expected in our real-time application. The kiosk application groups have some specific simplifying constraints, discussed in more detail in the body of this chapter, which help isolate the analysis of the retrieval performance and clarify the effects of specific parameters. Otherwise, only a qualitative treatment, *e.g.* [63, 25], can be applied as the system dynamics become more complex. Other related research efforts have concentrated on limited media forms, such as audio, *e.g.* the Etherphone project [70] which deals with the considerable task of processing voice samples. The Etherphone concept has been extended to video [75] in a multimedia conferencing application, however, the video component is supported as an analog signal.

In this chapter, we investigate the performance of maintaining a set of multimedia stubs in a cache which is local to each kiosk. The multimedia stubs are the preliminary parts of the audio-video sequences available to a particular kiosk station. In the next section, we describe the multimedia kiosk system architecture and the use of preliminary sequence

caching (PSC) in more detail. In addition to storing the preliminary sequences at each kiosk in an application group, the complete set of multimedia sequences of the application group are distributed among the kiosks. The use of distributed storage has the advantage of reducing contention at a single storage device. The distributed I/O architecture can also provide high data rates when objects are striped across the storage units, see *e.g.* [38, 13, 4, 73].

The goal of this chapter is to present an approach to multimedia storage for the kiosk application group architecture and evaluate its performance. We study an efficient means of servicing each kiosk group from an electronic library and then focus on the start-up latency experienced by users in a particular kiosk group. The start-up latency is the time between when the user requests a particular multimedia object and when "viewing" begins. This time is critical for user satisfaction and acceptance.

The idea is to manage a storage hierarchy, similar to the "classical" cache architecture discussed in the previous chapters. We utilize a cyclic servicing strategy to manage requests to the electronic library. We also expand the techniques developed for the kernel restricted overlap model to guide our analysis of a kiosk application group. In the following sections, we outline the kiosk system architecture, an application description, a model of an electronic library utilizing an optical disk jukebox with cyclic service of storage platters, an extension of the restricted overlap model which represents the kiosk system, and finally some results.

4.1 System Architecture

The architecture of a single kiosk, see Figure 4.2, is composed of five main components: a graphical user interface (GUI), local storage, a multimedia presentation block (MPB), a network interface (NI), and a control block (CB). The GUI includes both a menu overlay

for information navigation and the input mechanism, *e.g.* a touch screen. The local storage is schematically represented in Figure 4.2 as a disk. However, this "device" represents the logical view of the storage hierarchy, encompassing the local cache and the multimedia objects available across the kiosk group. The MPB includes the necessary techniques for bringing the stored multimedia sequences to the user, *e.g.* decoding compressed video. The NI is the implementation of the protocol which provides data exchange between the kiosk itself and the electronic library or other kiosks in the application group. Finally, the CB manages the block interplay.

The distributed multimedia system architecture can be divided into two main components, see Figure 4.1:

- the electronic library
- a kiosk application group

The electronic library is an information system which manages both collections and catalogs. The development of electronic libraries stemmed from the desire to replace paper in large research and administrative applications. The requirements for an electronic library were outlined in [26] and IBM offers the *Image and Records Management (IRM) System* which was designed to meet those requirements [36, 26].

The electronic library, as viewed by the kiosk group, can be thought of as a tertiary or archival store which maintains a kiosk group with a particular set of multimedia sequences constituting an application. Since the electronic library has massive collections, the cost must be controlled [48] by accepting reduced performance in a storage device such as an optical disk jukebox. This excludes its use as a direct server of requests generated by users at the kiosk stations. The electronic library's function is to configure the local storage of each kiosk in the application group and then let the group operate independently except

for any multimedia object updates or a complete "overhaul" with a new set of application sequences. The idea is to provide interactive response to relatively small data sets, which can be staged to the kiosk group, and provide scheduled service to the full set of objects which are archived in the electronic library.

In this chapter, we make use of the following kiosk characteristics which are derived naturally from the application:

- *primarily read-only requests* - the typical user only retrieves multimedia sequences from the kiosk (loading the kiosk storage with data is reserved as an electronic library task or 'superuser' task)
- *real-time service needs* - when a request is made at a kiosk, the user expects immediate response
- *asymmetric compression* - the primary function of the kiosk is playback of multimedia sequences, therefore off-line compression techniques may be applied to the sequences in an effort to optimize compression (decompression must be supported in real-time)

In the next section, we present a model of an electronic library utilizing an optical disk jukebox with cyclic service of storage platters.

4.2 Electronic Library

4.2.1 Modeling Cyclic Service of Optical Jukebox Platters

The optical disk jukebox with cyclic platter service was modeled as a set of N queues where N is the number of platters in the jukebox, see Figure 4.3. Service of the active, or currently loaded, platter continues while its queue is not empty. Upon emptying the active platter's queue (*i.e. exhaustive service*), the next non-empty queue is made active

and a switch-over time is incurred. The additional switch-over time only contributes to the response time when a request is present for the next platter in the cycle of service.

This represents the optimization of not loading a platter into the player if no requests are present for that platter when its turn in the cycle occurs. By introducing this optimization, we no longer have a simple expression for the switch-over statistics needed to directly apply the models of [5, 41, 66, 76]. Thus, we introduce a simple model assumption in an effort to approximate the jukebox model. We start by introducing some notation.

The following notation is used to represent the parameters in the system and will be used consistently throughout this paper unless otherwise noted:

- λ_i - the Poisson process arrival rate at platter i
- β and $\beta^{(2)}$ - the first and second moments of a general distribution representing the time to service each request excluding the platter switching time
- s_i and $s_i^{(2)}$ - the first and second moments of a general distribution representing the time to switch from platter i to platter $i+1$
- $\bar{X}_i = \bar{X} \quad \forall\ i$ - the average number of requests queued for platter i (the expected number is the same for all i's due to the symmetry of our problem definition)
- $\bar{W}_i = \bar{W} \quad \forall\ i$ - the average waiting time for each request queued for platter i (again due to symmetry, the expected waiting time is equivalent for all i's)

We assume the λ_i's are the generated from a single Poisson source, λ, uniformly distributed to each platter, *i.e.*

$$\begin{aligned} \lambda &= \sum_{i=1}^{N} \lambda_i \\ &= \sum_{i=1}^{N} \frac{\lambda}{N} \ . \end{aligned}$$

The traffic intensity at each individual platter and the overall intensity are defined as

$$\begin{aligned} \rho_i &= \lambda_i \cdot \beta \quad \text{and} \\ \rho &= \sum_{i=1}^{N} \rho_i = \lambda \cdot \beta, \quad \text{respectively.} \end{aligned}$$

Also, denote the expected total switch-over time in a cycle through all the platters as

$$s = \sum_{i=1}^{N} s_i \ .$$

with corresponding second moment, $s^{(2)}$.

If the switch-over times, in the given system, are zero (*i.e.* the time to swap disks is negligible), the *conservation laws* apply. This means the total amount of work is independent of the choice of service order and service is nonpreemptive. Therefore, from Kleinrock [40], the following result holds,

$$\sum_{i=1}^{N} \rho_i \cdot W_i = \frac{\rho \cdot W_0}{1-\rho} \tag{4.1}$$

$$\Rightarrow \quad W = \frac{W_0}{1-\rho} \quad \square \tag{4.2}$$

where

$$\begin{aligned} W_0 &= \sum_{i=1}^{N} \rho_i \cdot \frac{\beta^{(2)}}{2\beta} \\ &= \frac{\lambda \cdot \beta^{(2)}}{2} \end{aligned}$$

However, assuming zero switch-over time for the jukebox model is overly optimistic since the major time penalty incurred in the system is the time spent changing platters. Therefore, we must account for the switch-over times in order to realistically represent the optical disk jukebox with cyclic platter service. Since no work is accomplished by the server

during the switching times and we can change the number of switches to serve some set of requests based on our choice of scheduling, we can no longer service requests out of order and expect the total work to be independent of that order. Thus, the conservation law no longer applies when one accounts for the time to swap platters.

Since systems with non-zero switch-over times fail to satisfy the conservation law constraints, the results must be generalized to include such systems. The work of Boxma and Groenendijk [5] provides an elegant proof of the so-called pseudo-conservation laws applied to cyclic service systems. The conservation law is generalized to include switch-over times through the stochastic decomposition result,

$$V_c \overset{distr}{=} V + Y \tag{4.3}$$

where the *left-hand side (l.h.s.)* and *right-hand side (r.h.s.)* are equal in distribution and

V_c = amount of work in a cyclic service system at an arbitrary epoch

V = amount of work in a cyclic service system with zero switch-over times at an arbitrary epoch

Y = amount of work in a cyclic service system at an arbitrary epoch in the switching period

The proof of (4.3) is found in [5] which follows from the results developed in [24]. Equation (4.3) leads directly to

$$\Rightarrow \quad \bar{V_c} = \bar{V} + \bar{Y} \tag{4.4}$$

$$= \frac{\lambda\beta^{(2)}}{2\cdot(1-\rho)} + \bar{Y} \tag{4.5}$$

where the term, $\lambda\beta^{(2)}/(2\cdot(1-\rho))$ comes from the zero switch-over time calculation in (4.2). Also note, with the same line of reasoning used to derive (4.2), [5] shows,

$$\bar{V_c} = \sum_{i=1}^{N} \beta \bar{X_i} + \sum_{i=1}^{N} \rho_i \frac{\beta^{(2)}}{2\beta} \tag{4.6}$$

$$= \rho \bar{W} + \frac{\lambda \beta^{(2)}}{2} \tag{4.7}$$

Here, $\bar{V_c}$ is written as the sum of serving all outstanding platter requests at an arbitrary time epoch plus the residual of completing service on a request found at an arbitrary time epoch. Equations (4.5) and (4.7) generate the following expression for the average waiting time of a platter request,

$$\bar{W} = \frac{\lambda \beta^{(2)}}{2 \cdot (1 - \rho)} + \frac{1}{\rho} \bar{Y} \tag{4.8}$$

Now, we are left with deriving an expression for $\bar{Y}$, the average amount of work in the cyclic service system at an arbitrary epoch in the switching interval. This is accomplished by first considering $\bar{Y_i}$, the average amount of work in the system at a single, arbitrary switch-over period between platter i and platter $i+1$. Then,

$$\bar{Y} = \sum_{i=1}^{N} \Pr\{\text{switching from } i \text{ to } i+1 \mid \text{switch occurring}\} \cdot \bar{Y_i}$$

$$= \sum_{i=1}^{N} \frac{s_i}{s} \bar{Y_i} \tag{4.9}$$

Define the following variable for $k = 1$ to $N - 1$:

$$T_{i-k} = s_{i-k} + \frac{\rho_{i-k+1} s}{1 - \rho} + T_{i-k+1} \tag{4.10}$$

$$\text{where} \quad T_i = 0$$

T_{i-k} is an expression for the time it takes to complete service at queue i starting at the service completion epoch of the k^{th} previous queue's completion. In the equation, s_{i-k} is the time to switch from platter $i-k$ to $i-k+1$ (as defined previously) and $\frac{\rho_{i-k+1} s}{1-\rho}$ is the average visit time at queue $i-k+1$. With T_{i-k}, we can express $\bar{Y_i}$ as,

$$\bar{Y}_i = \rho\frac{s_i^{(2)}}{2s_i} + \sum_{k=1}^{N-1} \rho_{i-k} \cdot T_{i-k} \tag{4.11}$$

The first term on the *r.h.s.* represents the average amount of work arriving in the system during the time left to switch-over from i to $i+1$. The second term describes the average amount of work at the platter queues other than i at the epoch of completing i's service and the start of the switch-over period. There is no need to consider the average work arriving at platter i since our service strategy is exhaustive. Therefore, the average work at queue i will be zero.

From (4.9), (4.10), (4.11), and some algebra,

$$\bar{Y} = \rho\frac{s^{(2)}}{2s} + \frac{s}{2\cdot(1-\rho)}(\rho^2 - \sum_{i=1}^{N}\rho_i^2) \tag{4.12}$$

Finally, from (4.8) and (4.12),

$$\bar{W} = \frac{\lambda\beta^{(2)} + \rho(1-\frac{1}{N})}{2\cdot(1-\rho)} + \frac{s^{(2)}}{2s} \quad \square \tag{4.13}$$

This result from [5] could be applied directly to our system if the s_i's and $s_i^{(2)}$'s are known. However, given the dynamics of the system, these parameters are state dependent, where *state* is the current number of requests queued at each platter. So, we introduce a simplying assumption which approximates the cycle time (defined in the next section) which leads us to an approximation of the optical jukebox performance.

In the optical jukebox dynamics, recall that only when an actual swap occurs (*i.e.* the next platter queue is nonempty) is a time penalty charged. Therefore, empty platters are skipped since there is no need to load them into the player. In order to account for this optimization in the analytic model, we assume that the average cycle time (defined below) is exponentially distributed. This assumption allows us to generate the probability that a

request is pending in the next platter's queue and thus a switch-over time penalty will be accessed.

First, we derive the average cycle time, *i.e.* the average time to cycle through all platters, as follows:

$$\bar{C} = \sum_{i=1}^{N}(n_i\beta + s_i)$$

where n_i is the average number of requests served at platter i in a cycle. Therefore, applying Little's Law we get,

$$\begin{aligned} \bar{C} &= \sum_{i=1}^{N}(\bar{C}\lambda_i\beta + s_i) \\ &= \bar{C}\rho + s \\ \Rightarrow \quad \bar{C} &= \frac{s}{1-\rho} \end{aligned} \tag{4.14}$$

Now, we apply the assumption that the average time to cycle through the group of platters is exponentially distributed. This implies that the arrivals at each queue follow a Poisson process, *i.e.* Prob$\{k$ arrivals at queue i in a cycle$\}$ $= (\lambda_i\bar{C}^k/k!)\cdot e^{-\lambda_i\bar{C}}$ and the probability of some request on the next platter is $1 - e^{-\lambda_i\bar{C}}$.

Therefore, the switch-over statistics, s and $s^{(2)}$ may be generated as follows:

1. Select an initial value for s.
2. Calculate $\bar{C} = \frac{s}{1-\rho}$.
3. Calculate the new value of s as

$$\begin{aligned} s &= \hat{s}\sum_{i=1}^{N}(1 - e^{-\lambda_i\bar{C}}) \\ &= \hat{s}\cdot N\cdot(1 - e^{-\frac{\lambda}{N}\bar{C}}) \end{aligned}$$

4. Iterate back to 2 until a desired precision is reached.

5. Calculate $s^{(2)}$ as

$$\begin{aligned} s^{(2)} &= \hat{s}^{(2)} \sum_{j=1}^{N} \sum_{i=1}^{N} (1 - e^{-\lambda_j \bar{C}}) \cdot (1 - e^{-\lambda_i \bar{C}}) \\ &= \hat{s}^{(2)} \cdot N^2 \cdot (1 - e^{-\frac{\lambda}{N} \bar{C}})^2 \end{aligned}$$

6. Finally, use the values of s and $s^{(2)}$ to calculate the average waiting time given by (4.13).

where we use the notation $\hat{s}$ and $\hat{s}^{(2)}$ to represent the average switch-over time and the second moment of the switch-over time, respectively, given a change of platters occurs.

4.2.2 Results

Some representative results of the work described in the previous section are shown in this section. Figure 4.4 demonstrates the advantage of the cyclic service strategy over first-come first-serve (FCFS) strategy and the accuracy of the proposed approximation scheme for optical disk jukeboxes of different sizes (N).

The following parameter values, chosen to realistically represent current optical disk jukebox technology, were assumed for all the simulation and approximation runs:

- *Switch-over Time* $(\hat{s}, \hat{s}^{(2)})$ - the time it takes to unload a platter when its requests are exhausted and load the next platter which has a request awaiting service $\Rightarrow$ *10-15 seconds uniformly distributed.*
- *File Service Time* $(\beta, \beta^{(2)})$ - the rotational latency, seek, and transfer time of a file on the currently loaded platter $\Rightarrow$ *exponentially distributed with a mean of* $\frac{2}{3}$ *seconds.*
- *Arrival Rate* (λ) - the rate at which new requests enter the system $\Rightarrow$ *varied from 1.2 to 12.0 per minute* (λ is the horizontal axis of the graph).

- *Platter Count* (N) - the number of optical disk platters in the jukebox $\Rightarrow$ *chosen at 100*

A discrete event simulation was developed to model the optical disk jukebox with the parameters described above. We simulated two service disciplines, *FCFS* and *cyclic service*. The FCFS simulation was a single queue model with service time equal to the file service time if the previous request was for the same platter, or the file service time plus the switch-over time if the previous request was not for the same platter. The cyclic service simulation was modeled as a set of N queues. The requests were uniformly distributed to each queue. Therefore, each queue sees a Poisson arrival rate of λ/N. Service of a single queue (corresponding to the loaded platter) continues while the queue is full. Upon emptying that queue, the next nonempty queue begins service and a switch-over time is incurred.

Figure 4.4 dramatically shows the abrupt degradation in average response time using FCFS when arrivals reach a critical rate. High throughput levels cannot be sustained due to platter switches dominating the jukebox activities when accessed sequentially. In contrast, the cyclic method of service degrades gradually as the request arrival rates increase. In the runs we examined, the approximation scheme appears to accurately represent the cyclic service behavior. The interested reader is referred to [44] for more data and further discussion regarding the approximation's effectiveness.

We have shown a substantial performance gain, in terms of average waiting time, of exhaustive cyclic service over a first-come first-serve strategy for servicing requests to an optical disk jukebox in an archival application environment.

In the next section, we outline an application characterization used to derive a synthetic workload for the kiosk model. In the absence of some "real" traces, we use the application

characterization as a reasonable approximation of a workload.

4.3 Kiosk Application Group

4.3.1 Application Characterization

We consider the application presented to the kiosk group with the hierarchy shown in Figure 4.5. The root represents the application running on the kiosk group. Each topic represents a major subject heading. The "leaf" level represents the set of video sequences (multimedia objects) which comprise the topic. We utilize this general organization to form a workload model whereby any access to a multimedia sequence falling under a particular topic results in the preliminary sequence caching of the remaining sequences under the same heading. Thus, the preliminary stubs of each sequence are staged into local kiosk memory in expectation of some reference locality with respect to the topic. With the preliminary sequence stub local, the start-up latency can be reduced.

The idea is best demonstrated through an example. Consider the environment of an institute of education: a university, high school, grade school, etc. As a supplement to the courses, the library or learning center provides a kiosk application group from which students can retrieve multimedia briefs on various topics related to the classes.

For example, the root of the application hierarchy could represent a general discipline, *e.g. Electrical Engineering*. The topics could be comprised of various courses, *e.g. Circuits, Signal Processing, Computer Architecture, etc.* with the sequences corresponding to some clips pertaining to those course. Alternatively, the application group could correspond to a single course, *Circuits, Devices, and Systems*, allowing more detailed coverage of the topics *Ohm's Law, Op-Amps, Transistors, etc.* Obviously, the depth of the tree could be

expanded to cover both cases. However, we consider a workload of the form in Figure 4.5 to characterize the input to our model.

4.3.2 Kiosk Model

In this section, we develop another extension of the restricted overlap model to represent a kiosk group using preliminary sequence caching. A closed queueing model, shown in Figure 4.6, is considered in order to represent a typical user's behavior which is characterized by

- a period of think time,
- a request for a particular multimedia sequence, and
- viewing the sequence.

The kiosk group is viewed as a logical device, similar to the disk array model of Chapter 3. Accordingly, the server state descriptions, listed below, resemble those of the RAID-5 model. The kiosk storage architecture maintains a PSC at each kiosk. Therefore, stage II is expanded to a multiserver station similar to stage III of both this model and the disk array model. The resulting server state descriptions are as follows:

$$
\begin{aligned}
I &\triangleq \text{server station I is busy exclusively} \\
II_i &\triangleq \text{server station(s) IIi is (are) busy exclusively} \\
III_i &\triangleq \text{server station(s) IIIi is (are) busy exclusively} \\
\theta_{ij} &\triangleq \text{server stations IIi and IIIi are both busy (overlap).} \\
\zeta_i &\triangleq \text{server stations I and IIIi are both busy (new overlap)}
\end{aligned}
$$

where $i, j \in \{1, \cdots, K\}$, and K is the number of kiosks that form the application group. As K grows, the size of the state space becomes quite large. Therefore, we approximate the solution by constraining $i, j \in \{1, 2\}$. The accuracy of this approximation is assessed in the following results section.

Given our state description, the balance equations for the model can be written as:

$$
\begin{aligned}
p(n,I)((K-n)\lambda+\alpha) &= p(n-1,I)(K-n+1)\lambda + p(n,III_1)\gamma \\
&\quad + p(n+1,I)\alpha rp + p(n+1,II_1)\beta + p(n,\zeta_1)\gamma \\
p(n,II_1)((K-n)\lambda+\beta) &= p(n-1,II_1)(K-n+1)\lambda + p(n,\theta_{11})\gamma \\
&\quad + p(n+1,I)\alpha rq + p(n+1,II_2)2\beta p \\
p(n,II_2)((K-n)\lambda+2\beta) &= p(n-1,II_2)(K-n+1)\lambda + p(n,\theta_{21})\gamma \\
&\quad + p(n+1,II_2)2\beta q \\
p(n,III_1)((K-n)\lambda+\gamma) &= p(n-1,III_1)(K-n+1)\lambda + p(n+1,I)\alpha s\frac{p}{K} \\
&\quad + p(n+1,\theta_{11})\beta\frac{p}{K} \\
&\quad + p(n+1,\zeta_1)\alpha r\frac{p}{K} + p(n,III_2)2\gamma \\
p(n,\theta_{11})((K-n)\lambda+\beta+\gamma) &= p(n-1,\theta_{11})(K-n+1)\lambda + p(n+1,\zeta_1)\alpha rq \\
&\quad + p(n+1,I)\alpha sq \\
&\quad + p(n+1,\theta_{21})2\beta p + p(n,\theta_{12})2\gamma \\
p(n,\theta_{21})((K-n)\lambda+2\beta+\gamma) &= p(n-1,\theta_{21})(K-n+1)\lambda \\
&\quad + p(n+1,\theta_{21})2\beta q + p(n,\theta_{22})2\gamma \\
p(n,\zeta_1)((K-n)\lambda+\alpha+\gamma) &= p(n-1,\zeta_1)(K-n+1)\lambda + p(n+1,I)\alpha sp\frac{K-1}{K} \\
&\quad + p(n+1,\theta_{11})\beta\frac{K-1}{K} \\
&\quad + p(n+1,\zeta_1)\alpha rp\frac{K-1}{K} + p(n,\zeta_2)2\gamma
\end{aligned}
$$

$$
\begin{aligned}
p(n,III_2)((K-n)\lambda+2\gamma) &= p(n-1,III_2)(K-n+1)\lambda + p(n+1,\theta_{12})\beta\frac{2}{K} \\
&\quad + p(n+1,\zeta_1)\alpha s\frac{2p}{K} + p(n+1,\zeta_2)\alpha\frac{2p}{K} \\
p(n,\theta_{12})((K-n)\lambda+\beta+2\gamma) &= p(n-1,\theta_{12})(K-n+1)\lambda + p(n+1,\theta_{22})2\beta p \\
&\quad + p(n+1,\zeta_1)\alpha sq + p(n+1,\zeta_2)\alpha q \\
p(n,\theta_{22})((K-n)\lambda+\beta+2\gamma) &= p(n-1,\theta_{22})(K-n+1)\lambda + p(n+1,\theta_{22})2\beta q \\
p(n,\zeta_2)((K-n)\lambda+\alpha+2\gamma) &= p(n-1,\zeta_2)(K-n+1)\lambda + p(n+1,\theta_{12})\beta\frac{K-2}{K} \\
&\quad + p(n+1,\zeta_1)\alpha sp\frac{K-2}{K} + p(n+1,\zeta_2)\alpha p\frac{K-2}{K}
\end{aligned}
$$

where $K > n > 2$. The boundary condition, $n = K$, has the same set of equations as above without the "n+1" terms on the right-hand sides. Likewise, the boundary condition, $n = 2$, follows the same equations, with the following exceptions:

$$
\begin{aligned}
p(2,II_1)((K-2)\lambda+\beta) &= p(1,II_1)(K-1)\lambda p + p(2,\theta_{11})\gamma + p(3,I)\alpha rq \\
&\quad + p(3,II_2)2\beta p \\
p(2,II_2)((K-2)\lambda+2\beta) &= p(1,II_1)(K-1)\lambda q + p(2,\theta_{21})\gamma + p(3,II_2)2\beta q \\
p(2,\theta_{11})((K-2)\lambda+\beta+\gamma) &= p(1,\theta_{11})(K-1)\lambda p + p(3,\zeta_1)\alpha rq + p(3,I)\alpha sq \\
&\quad + p(3,\theta_{21})2\beta p + p(2,\theta_{12})2\gamma \\
p(2,\theta_{21})((K-2)\lambda+2\beta+\gamma) &= p(1,\theta_{11})(K-1)\lambda q \\
&\quad + p(3,\theta_{21})2\beta q + p(2,\theta_{22})2\gamma \\
p(2,\theta_{12})((K-2)\lambda+\beta+2\gamma) &= p(1,\theta_{12})(K-1)\lambda p + p(3,\theta_{22})2\beta p + p(3,\zeta_1)\alpha sq \\
&\quad + p(3,\zeta_2)\alpha q \\
p(2,\theta_{22})((K-2)\lambda+\beta+2\gamma) &= p(1,\theta_{12})(K-1)\lambda q + p(3,\theta_{22})2\beta q
\end{aligned}
$$

Finally, the boundary conditions for $n = 0$ and $n = 1$ are described as follows:

$$
\begin{aligned}
p(0,I\bar{I}I)K\lambda &= p(1,I)\alpha r + p(1,II_1)\beta + p(0,III_1)\gamma \\
p(0,III_1)(K\lambda+\gamma) &= p(1,I)\alpha s + p(1,\theta_{11})\beta + p(1,\zeta_1)\alpha r + p(0,III_2)2\gamma \\
p(0,III_2)(K\lambda+2\gamma) &= p(1,\zeta_1)\alpha s + p(1,\zeta_2)\alpha + p(1,\theta_2)\beta
\end{aligned}
$$

$$
\begin{aligned}
p(1,I)((K-1)\lambda+\alpha) &= p(0,I\bar{I}I)K\lambda p + p(1,III_1)\gamma + p(2,I)\alpha r p \\
&\quad + p(2,II_1)\beta + p(1,\zeta_1)\gamma \\
p(1,II_1)((K-1)\lambda+\beta) &= p(0,I\bar{I}I)K\lambda q + p(1,\theta_{11})\gamma + p(2,I)\alpha r q + p(2,II_2)2\beta \\
p(1,III_1)((K-1)\lambda+\gamma) &= p(0,III_1)\lambda p + p(2,I)\alpha s\frac{p}{K} + p(2,\theta_{11})\beta\frac{1}{K} \\
&\quad + p(2,\zeta_1)\alpha r\frac{p}{K} + p(1,III_2)2\gamma \\
p(1,\theta_{11})((K-1)\lambda+\beta+\gamma) &= p(0,III_1)K\lambda q + p(2,\zeta_1)\alpha r q + p(2,I)\alpha s q \\
&\quad + p(2,\theta_{21})2\beta + p(1,\theta_{12})2\gamma \\
p(1,\zeta_1)((K-1)\lambda+\alpha+\gamma) &= p(0,III_1)\lambda p(K-1) + p(2,I)\alpha s p\frac{K-1}{K} \\
&\quad + p(2,\theta_{11})\beta\frac{K-1}{K} + p(2,\zeta_1)\alpha r p\frac{K-1}{K} + p(1,\zeta_2)2\gamma \\
p(1,III_2)((K-1)\lambda+2\gamma) &= p(0,III_2)\lambda 2p + p(2,\theta_{12})\beta\frac{2}{K} + p(2,\zeta_1)\alpha s\frac{2p}{K} \\
&\quad + p(2,\zeta_2)\alpha\frac{2p}{K} \\
p(1,\theta_{12})((K-1)\lambda+\beta+2\gamma) &= p(0,III_2)K\lambda q + p(2,\theta_{22})2\beta + p(2,\zeta_1)\alpha s q + p(2,\zeta_2)\alpha q \\
p(1,\zeta_2)((K-1)\lambda+\alpha+2\gamma) &= p(0,III_2)\lambda p(K-2) + p(2,\theta_{12})\beta\frac{K-2}{K} \\
&\quad + p(2,\zeta_1)\alpha s p\frac{K-2}{K} + p(2,\zeta_2)\alpha p\frac{K-2}{K}
\end{aligned}
$$

These equation are then solved using the semi-numerical techniques presented in Chapter 2. The following section outlines some of our results.

4.3.3 Results

In this section, we evaluate the restricted overlap model as applied to the multimedia kiosk application group. We also evaluate the effectiveness of preliminary sequence caching on reducing start-up latencies for serving multimedia streams. Figure 4.7 and Figure 4.8 demonstrate the kiosk start-up response time for varying arrival rates and varying number of kiosks, respectively. The arrival rates, $0.05, 0.10, 0.15$, and 0.20 IOs/min, of Figure 4.7 are derived by considering multimedia streams of $20, 10, 6\frac{2}{3}$, and 5 minute lengths. The graph with a varying number of kiosks had a fixed per kiosk arrival rate of 0.10 IOs/min. We assume a 1 sec start-up latency for non-local sequences (stage I), a $\frac{1}{2}$ sec latency for sequences in the PSC (stage II_i), and a 12 sec busy period for misses resulting in a PSC staging period (stage III_i). Five curves are generated for each graph, corresponding to a different level of PSC locality ($p = 0.1, 0.2, 0.3, 0.4$, and 1.0). We include the $p = 1.0$ plot in order to compare the average start-up response when no preliminary sequence caching is used. In the examples shown, the use of preliminary sequence caching can lower the average start-up latency to delays which are less perceivable by the user.

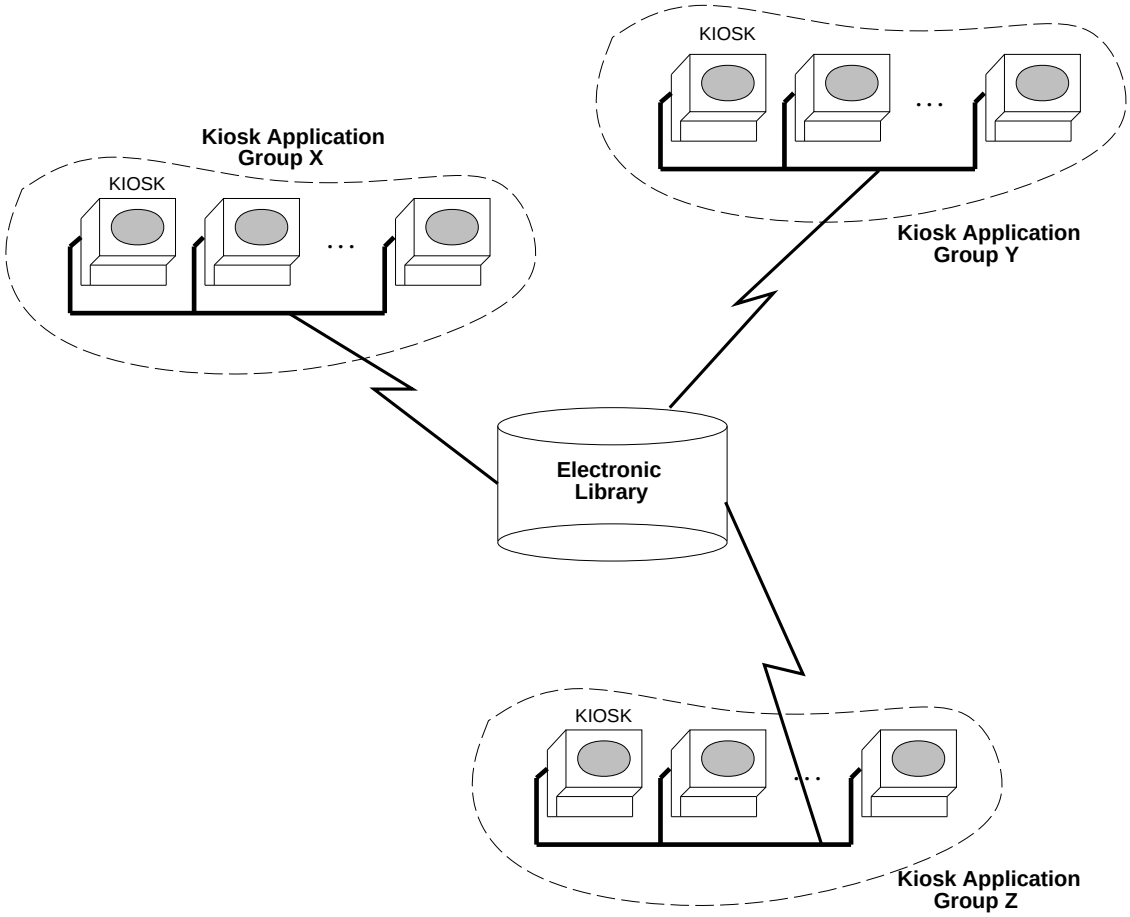

Figure 4.1: Distributed kiosk groups and electronic library

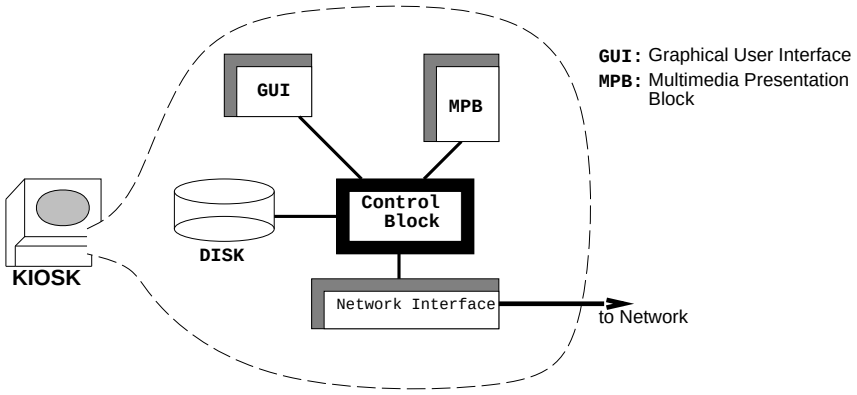

Figure 4.2: Kiosk Architecture

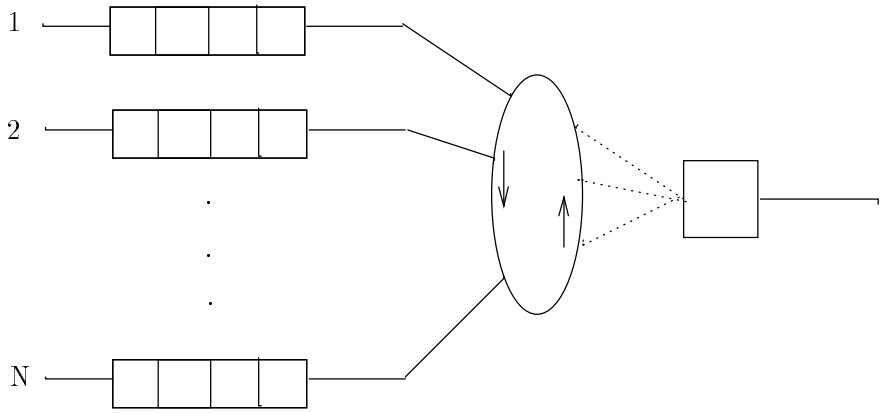

Figure 4.3: Cyclic Queueing Model

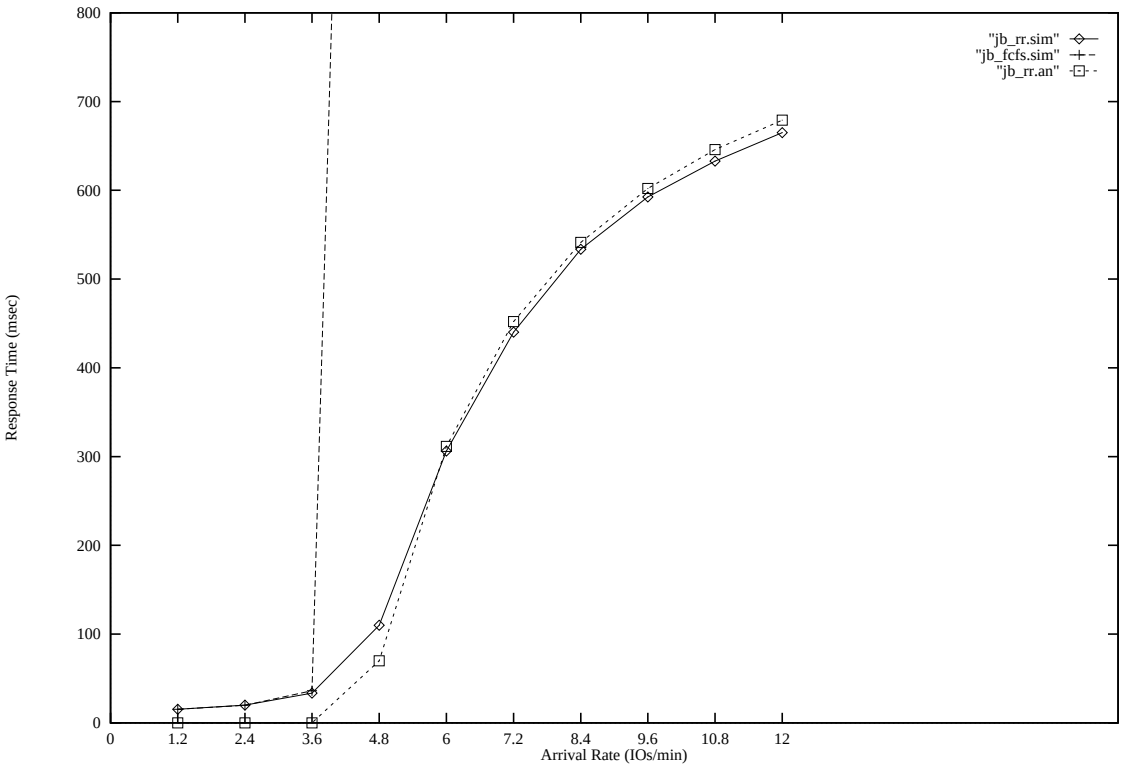

Figure 4.4: Performance of optical jukebox with 100 platters

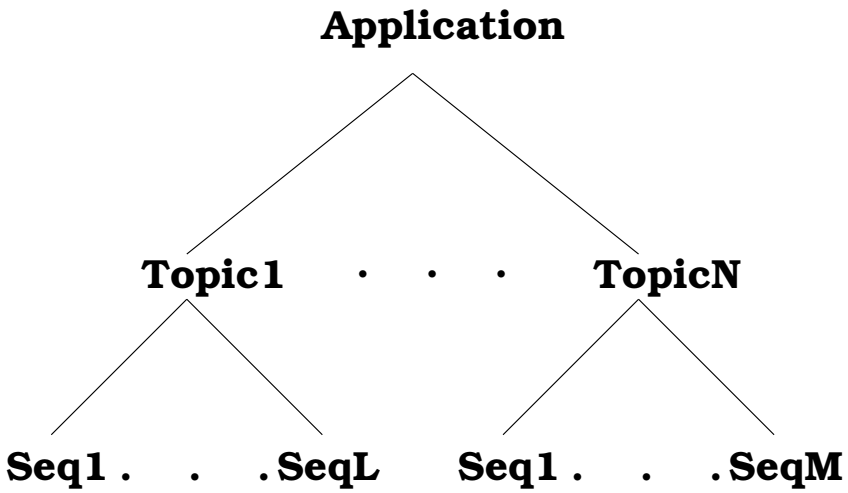

Figure 4.5: Kiosk application hierarchy

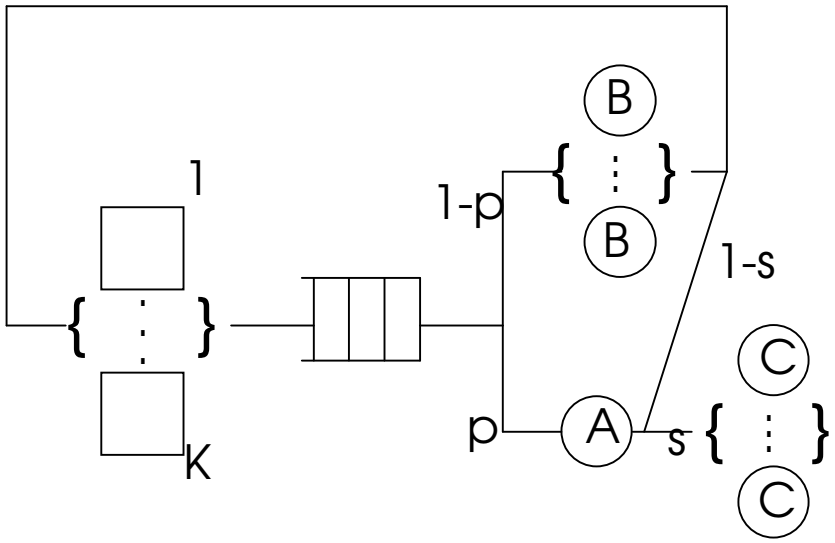

Figure 4.6: Kiosk queueing model

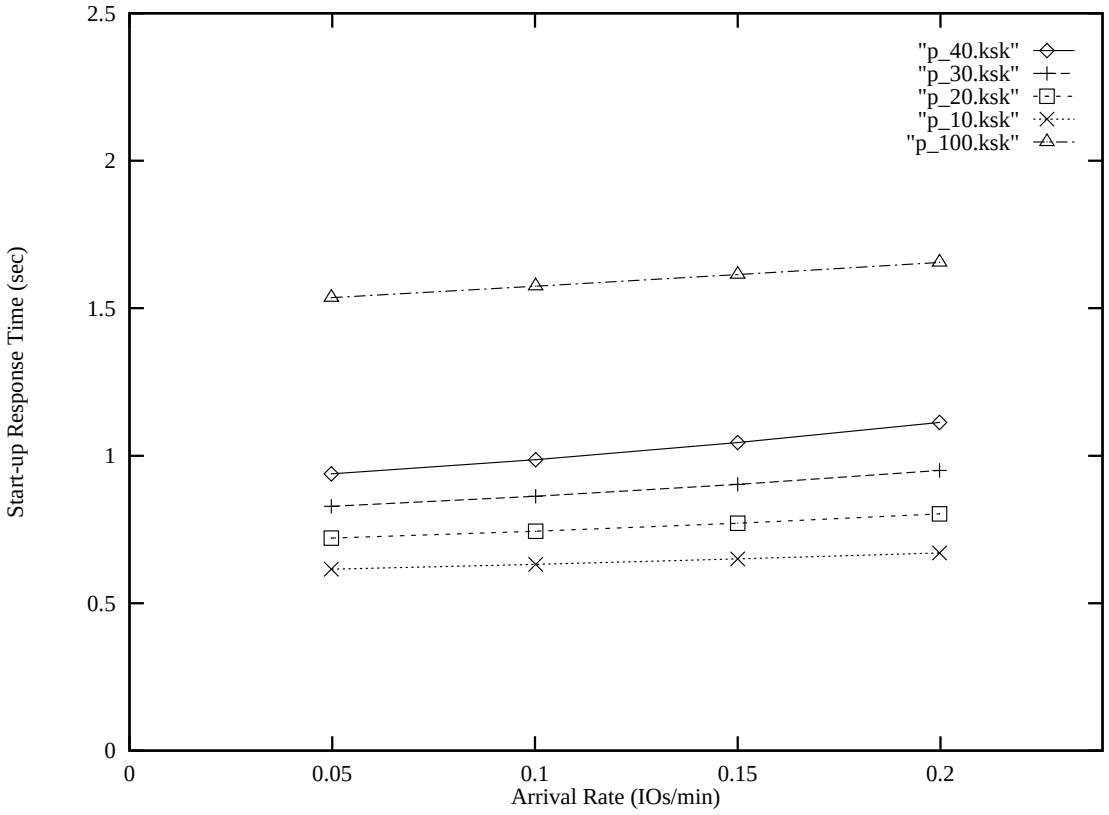

Figure 4.7: Performance of kiosk group with respect to changes in the arrival rate

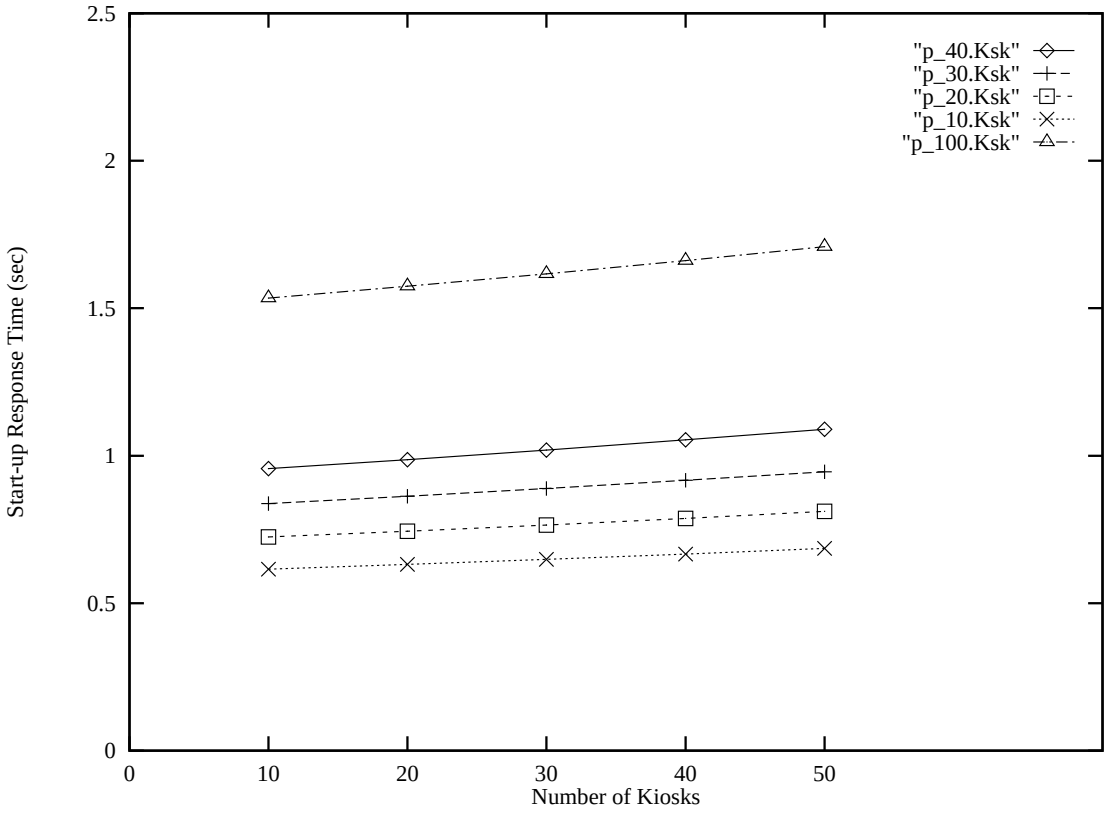

Figure 4.8: Performance of kiosk group with respect to changes in the number of kiosks

5. Conclusions

In this dissertation, we have presented an analytical model which represents the restricted overlap of resources. The model was developed to analyze the constrained parallel activity of service in modern cached I/O storage subsystems. A vacation model was utilized to represent the restricted overlap behavior. We started with a kernel model developed under exponential assumptions, in which a logical device corresponded to a single physical device and the effects of caching.

The efficiency of three solution techniques, two based on equivalence methods and one based on the matrix-geometric method, was investigated. The first solution was generated using a simple iterative approach. Next, by calculating each iteration with more recently derived information (requiring some added some complexity), the "modified iteration" demonstrated an effective speed-up over the simple iteration. We also studied a matrix-geometric solution to the restricted overlap model. Its efficiency was shown to be similar to the modified approach. We note our approach to the matrix-geometric solution considered a straightforward matrix manipulation. The further investigation of more efficient linear algebra methods could be an interesting direction for future studies.

Next, the restricted overlap model was extended to account for general distributions in service. We applied two methods: a coxian equivalence and a distributional adjustment approximation. For the points analyzed, the trade-off appears to be in the level of variation represented versus the level of computational complexity. The adjustment method is applied for lower variability since the coxian approach is not applicable. However, the added complexity of solving the coxian method may be acceptable at higher variability in order to generate a more accurate solution.

Chapter 3 extended the restricted overlap model to represent a cached RAID-5 storage subsystem. The model was applied to a disk array treated as a single logical device and multiple logical devices. The restricted overlap model was shown to be an effective approach as validated versus simulation. Since our goal was to focus on the overlap behavior, we did not represent path contention or any buffering which may occur along a path. It would be interesting to include their effects in future extensions of the model.

Other possible future research regarding the RAID-5 model could include characterizing performance in the presence of "hot spots", degraded mode operation, and cache write-back policies. Hot spots occur when the access patterns to the array of disks become skewed. As mentioned in Chapter 3, the model branching probabilities could be augmented to account for the skewed behavior. One of the characteristic features of the RAID-5 architecture is its ability to tolerate a device failure. However, the fault tolerance is derived from a data reconstructive procedure which leads to added device utilization. A future enhancement of the model could account for this added activity. Also, a more detailed analysis could be investigated for various write-back policies, as proposed by [74] for RAID cache destaging.

Chapter 4 presents a multimedia kiosk architecture. We presented a model of an electronic library with an optical disk jukebox utilizing cyclic service of the storage platters. The method is shown to be an effective service technique compared to a first-come first-serve strategy in which platter swap times dominate the jukebox activity resulting in poor performance. A closed model extension of the restricted overlap model was applied to a multimedia kiosk application. The goal was to evaluate the effectiveness of preliminary sequence caching in reducing the start-up latency in multimedia object presentation. It would be interesting to extend the multimedia kiosk model to account for the overall sequence delivery including any further enhancements, *e.g.* platter caching at the electronic

library level in the storage hierarchy.

Finally, since the restricted overlap model is a general abstraction of the overlap behavior, it would also be interesting to study how the model might be applied to other problems in computer engineering and, more generally, in operations research.

References

[1] Allen, A.O. *Probability, Statistics, and Queueing Theory.* Academic Press, 1990.

[2] Amdahl Corporation. *Amdahl 6100 Storage Processor.* Announcement, 1988.

[3] Baccelli, F., A.M. Makowski, and A. Schwartz. "The Fork-Join Queue and Related Systems with Synchronization Contraints: Stochastic Ordering and Computable Bounds". *Advances in Applied Probability*, (21), 1989.

[4] Berson, S., S. Ghandeharizadeh, R. Muntz, and X. Ju. "Staggered Striping in Multimedia Information Systems". *UCLA Computer Science Department Technical Report*, (CSD-930042), December 1993.

[5] Boxma, O.J. and W.P. Groenendijk. "Pseudo-Conservation Laws in Cyclic-Service Systems". *Journal of Applied Probabilty*, (24), 1987.

[6] Brandwajn, A. "An Iterative Solution of Two-Dimensional Birth and Death Processes". *Operations Research*, 27(3), 1979.

[7] Brandwajn, A. "Equivalence and Decomposition in Queueing Systems - A Unified Approach". *Performance Evaluation*, (5), 1985.

[8] Brandwajn, A. "Performance Benefits of Parallelism in Cached DASD Controllers". In *CMG 89 International Conference Proceedings.* CMG, December 1989.

[9] Brandwajn, A. *Private communication.* 1991.

[10] Bratley, P., B.L. Fox, and L.E. Schrage. *A Guide to Simulation.* Springer-Verlag, 1983.

[11] Buzen, J.P. and A.W. Shum. "Considerations for Modeling Shared DASD". In *CMG 89 International Conference Proceedings.* CMG, December 1989.

[12] Buzen, J.P. and A.W. Shum. "RAID, CAID, and Virtual Disks: I/O Performance at the Crossroads". In *CMG 93 International Conference Proceedings.* CMG, December 1993.

[13] Cabrera, L.F. and D.D.E. Long. "Using Disk Striping to Provide Multiple High I/O Data Rates". *Computing Systems*, 4(4), 1991.

[14] Cao, P., S.B. Lim, S. Venkataraman, and J. Wilkes. "The TickerTAIP Parallel RAID Architecture". *Hewlett-Packard Laboratory Technical Report*, (HPL-OSR-93-25), April 1993.

[15] Chen, S. and D. Towsley. "The Design and Evaluation of RAID 5 and Parity Striping Disk Array Architectures". *Journal of Parallel and Distributed Computing*, 17(1/2), January/February 1993.

[16] Chesnais, A., E. Gelenbe and I. Mitrani. "On the Modeling of Parallel Access to Shared Data". *Communications of the ACM*, 26(3), 1983.

[17] Cox, D.R. "A Use of Complex Probabilities in the Theory of Stochastic Processes". *Proc. Cambridge Phil. Soc.*, (51), 1955.

[18] Denning, P.J. "The Working Set Model for Program Behavior". *Communications of the ACM*, 11(5), May 1968.

[19] Denning, P.J. "Working Sets Past and Present". *IEEE Tranactions on Software Engineering*, SE-6(1), January 1980.

[20] Doshi, B.T. "Queueing Systems with Vacations". *Queueing Systems*, 1(1), June 1986.

[21] Erlang, A.K. "Solution of Some Problems in the Theory of Probabilities of Significance in Automatic Telephone Exchanges". *P.O. Elec. Engrs. Journal*, 10, 1918.

[22] Evans, R.V. "Geometric Distribution in Some Two-Dimensional Queueing Systems". *Operations Research*, (15), 1967.

[23] Fishman, G.S. *Principles of Discrete Event Simulation.* Wiley, 1978.

[24] Fuhrmann, S.W. and R.B. Cooper. "Stochastic Decompostion in the M/G/1 Queue with Generalized Vacations". *Operations Research*, (33), 1985.

[25] Ginige, A., A. Seneviratne, R. Gonzalez and S. Chandra. "An Experimental Multimedia System". In *Multimedia '92, 4th IEEE ComSoc International Workshop on Multimedia Communications*, April 1992.

[26] Gladney, H. M.and P. E. Mantey. "Essential Issues in the Design of Shared Document/Image Libraries". In *SPIE/SPSE Symposium on Electronic Imaging Science and Technology*, February 1990.

[27] Gray, J., B. Host, and M. Walker. "Parity Striping of Disk Arrays: Low-cost Reliable Storage with Acceptable Throughput". In *Proceedings 16th VLDB Conference*, August 1990.

[28] Hewlett-Packard. HP C2244/45/46/47 SCSI-2 Disk Drive. *Technical Reference Manual*, 1992.

[29] Hitz, D., J. Lau, and M. Malcolm. "File System Design for an NFS File Server Appliance". In *Proceedings of the 1994 Winter USENIX*, January 1994.

[30] Huang, Y. and Y. Chin. "Performance Evaluation of Three Locking Protocols". In Courtois, P.-J. and G. Latouche, editor, *Performance '87.* North Holland, 1987.

[31] IBM. *IBM 3380 Direct Access Storage Description and User's Guide.* Manual GA26-1664, 1981.

[32] IBM. *IBM System/370 XA Principles of Operation.* Manual SA22-7085, 1983.

[33] IBM. *IBM 3990 Storage Control Introduction.* Manual GA32-0098, 1987.

[34] IBM. *IBM 3390 Direct Access Storage Introduction.* Manual GC26-4573-2, 1990.

[35] IBM. *IBM Enterprise Systems Architecture/390 Principles of Operation.* Manual SA22-7200, 1990.

[36] IBM. Image and Records Management (IRM) General Information Guide (GC22-0027). 1991.

[37] Johnson, T. "Approximate Analysis of Reader and Writer Access to a Shared Resource". In *Proc. ACM Sigmetrics Conference on Measurement and Modelling of Computer Systems.* ACM Press, 1990.

[38] Katz, R.H., G.A. Gibson, and D.A. Patterson. "Disk System Architectures for High Performance Computing". *Proceedings of the IEEE*, 77(12), December 1989.

[39] Kim, M. "Synchronized Disk Interleaving". *IEEE Transactions on Computers*, C-35(11), November 1986.

[40] Kleinrock, L. *Queueing Systems: Theory*, volume 1. Wiley, 1975.

[41] Kuehn, P.J. "Multiqueue Systems with Nonexhaustive Cyclic Service". *The Bell System Technical Journal*, 58(3), March 1979.

[42] Law, A.M. and W.D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, 1982.

[43] Lee, E.K. and R.H. Katz. "An Analytic Performance Model of Disk Arrays". In *Proceedings ACM SIGMETRICS Conference*, May 1993.

[44] Levy, D.E. and P.E. Mantey. "Optical Jukebox Retrieval Performance in Electronic Libraries Utilizing Cyclic Scheduling". *UCSC Computer Research Laboratory Technical Report*, (UCSC-CRL-92-08), March 1992.

[45] Little, T.D.C. and A. Ghafoor. "Spatio-Temporal Composition of Distributed Multimedia Objects for Value-Added Networks". *Computer*, 24(10), October 1991.

[46] Livny, M., S. Khoshafian, and H. Boral. "Multi-disk Management Algorithm". In *Proceedings ACM SIGMETRICS Conference*, May 1987.

[47] MacDougall, M.H. *Simulating Computer Systems: Tools and Techniques*. The MIT Press, 1987.

[48] Mantey, P.E. and D.E. Levy. "Electronic Libraries and Optical Jukebox Server Scheduling". In *SPIE/SPSE Symposium on Electronic Imaging Science and Technology*, February 1992.

[49] Menon, J. and D. Mattson. "Performance of Disk Arrays in Transaction Processing Environments". In *Proceedings of the 12th International Conference on Distributed Computing Systems*, June 1992.

[50] Morris, R.J.T. and W.S. Wong. "Performance of Concurrency Control Algorithms with Nonexclusive Access". In Gelenbe, E., editor, *Performance '84*. North Holland, 1984.

[51] Nelson, R., D. Towsley and A.N. Tantawi. "Performance Analysis of Parallel Processing Systems". *IEEE Transactions on Software Engineering*, 14(4), 1988.

[52] Neuts, M.F. "Markov Chains with Applications in Queueing Theory, Which Have a Matrix-Geometric Invariant Vector". *Advances in Applied Probability*, (10), 1978.

[53] Neuts, M.F. *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*. The John Hopkins University Press, 1981.

[54] Parker, M. "Small Disks Come to Mainframes". *CMG Transactions*, (74), Fall 1991.

[55] Patterson, D.A., R.H. Katz, and G.A. Gibson. "A Case for Redundant Arrays of Inexpensive Disks (RAID)". In *Proceedings ACM SIGMOD Conference*, June 1988.

[56] Patterson, D.A., R.H. Katz, and G.A. Gibson. "A Case for Redundant Arrays of Inexpensive Disks (RAID)". In *Proceedings IEEE Spring CompCon*, June 1988.

[57] Rangan, P. Venkat and Harrick M. Vin. "Designing File Systems for Digital Video and Audio". In *Proceedings of the 13th ACM Symposium on Operating System Principles*, October 1991.

[58] Reiman, M.I. and P.E. Wright. "Performance Analysis of Concurrent-Read Exclusive-Write". In *Proc. ACM Sigmetrics Conference on Measurement and Modelling of Computer Systems*. ACM Press, 1991.

[59] Salem, K. and H. Garcia-Molina. "Disk Striping". In *International Conference on Data Engineering*, February 1986.

[60] Scherr, A.L. *An Analysis of Time-Shared Computer Systems*. The MIT Press, 1967.

[61] Schwetman, H.D. "CSIM Reference Manual (Revision 16)". *Microelectronics and Computer Technology Corporation Technical Report*, (ACT-ST-252-87Rev.16), May 1992.

[62] SCSI. Small Computer Systems Interface II. *X3T9 - I/O Interface*, 1990.

[63] Sincoskie, W.D. "System Architecture for a Large Scale Video on Demand Service". *Computer Networks and ISDN Systems*, September 1991.

[64] Strange, G. *Linear Algebra and Its Applications*. Harcourt Brace Jovanovich, Publishers, 1988.

[65] Swinehart, D. C. "Keynote Address: Multimedia Communication Systems: Are They Finally Real?". In *Multimedia '92, 4th IEEE ComSoc International Workshop on Multimedia Communications*, April 1992.

[66] Takagi, H. *Analysis of Polling Systems*. MIT Press, 1986.

[67] Takagi, H. *Queueing Analysis, A Foundation of Performance Evaluation: Vacation and Prioirty Systems*, volume 1. North-Holland, 1991.

[68] Tay, Y.C., N. Goodman and R. Suri. "Locking Performance in Centralized Databases". *ACM Transactions on Database Systems*, 10(4), 1985.

[69] Tay, Y.C., R. Suri and N. Goodman. "A Mean Value Performance Model for Locking in Databases: The No-Waiting Case". *Journal of the ACM*, 32(3), 1985.

[70] Terry, D.B. and D.C. Swinehart. "Managing Stored Voice in the Etherphone System". *ACM Transactions on Computer Systems*, February 1988.

[71] Thomasian, A. and I.K. Ryu. "A Decomposition Solution to the Queueing Network Model of the Centralized DBMS with Static Locking". In *Proc. ACM Sigmetrics Conference on Measurement and Modelling of Computer Systems*. ACM Press, 1983.

[72] Thomasian, A. and J. Menon. "Performance Analysis of RAID5 Disk Arrays with a Vacationing Server Model for Rebuild Mode Operation". In *Proceedings IEEE Data Engineering Conference*, February 1994.

[73] Tobagi, F.A., J. Pang, R. Baird, and M. Gang. "Streaming RAID - A Disk Array Management System for Video Files". In *Proceedings of ACM Multimedia 93*, 1993.

[74] Varma, A. *Private communication*. 1993.

[75] Vin, H.M., P.T. Zellwger, D.C. Swinehart, and P. Venkat Rangan. "Multimedia Conferencing in the Etherphone Environment". *Computer*, 24(10), October 1991.

[76] Watson, K.S. "Performance Evaluation of Cyclic Service Strategies". In Gelenbe, E., editor, *Performance '84*. North-Holland, 1984.