

1979

Performance theorems for the resource scheduling functions of a multiprocessing system

George Arthur Nicol
Portland State University

Follow this and additional works at: https://pdxscholar.library.pdx.edu/open_access_etds



Part of the [Operations Research, Systems Engineering and Industrial Engineering Commons](#), and the [Systems and Communications Commons](#)

Let us know how access to this document benefits you.

Recommended Citation

Nicol, George Arthur, "Performance theorems for the resource scheduling functions of a multiprocessing system" (1979). *Dissertations and Theses*. Paper 2784.

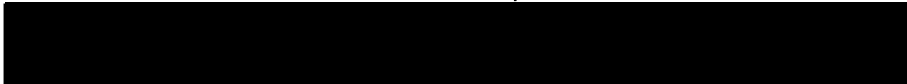
<https://doi.org/10.15760/etd.2779>

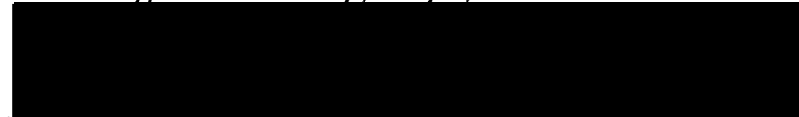
This Dissertation is brought to you for free and open access. It has been accepted for inclusion in Dissertations and Theses by an authorized administrator of PDXScholar. Please contact us if we can make this document more accessible: pdxscholar@pdx.edu.

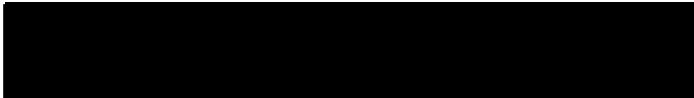
AN ABSTRACT OF THE DISSERTATION of George Arthur Nicol for the degree of Doctor of Philosophy in Systems Science presented May 15, 1979.

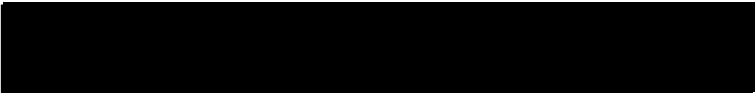
Title: Performance Theorems for the Resource Scheduling Functions of a Multiprocessing System

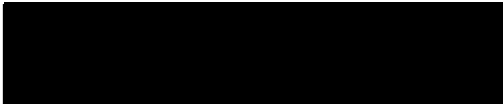
APPROVED BY MEMBERS OF THE DISSERTATION COMMITTEE:


Richard B. Crittenden, Chairman


George G. Lendaris, Systems Science


Robert L. Stanley, Mathematics


Robert W. Rempfer, Mathematics


James L. Hein, Mathematics

In this dissertation, a multiprocess scheduling model is developed and a set of performance theorems is constructed for the set of scheduling functions associated with the model. Each theorem describes the conditions under which a scheduling function exhibits a particular type of performance.

PERFORMANCE THEOREMS FOR THE RESOURCE SCHEDULING FUNCTIONS
OF A MULTIPROCESSING SYSTEM

by

GEORGE ARTHUR NICOL

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


DOCTOR OF PHILOSOPHY
in
SYSTEMS SCIENCE

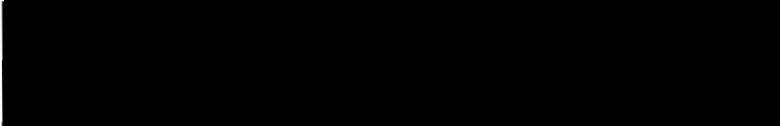
Portland State University
1979


George Arthur Nicol © 1979


TO THE OFFICE OF GRADUATE STUDIES AND RESEARCH:


The members of the Committee approve the dissertation of
GEORGE ARTHUR NICOL presented May 15, 1979.


Richard B. Crittenden, Chairman



George G. Lendaris, Systems Science



Robert L. Stanley, Mathematics


Robert W. Rempfer, Mathematics


James L. Hein, Mathematics

APPROVED:


Richard B. Halley, Acting Director of Systems Science


Stanley E. Rauch, Dean of Graduate Studies and Research

ACKNOWLEDGEMENTS

I would like to thank Professors Crittenden, Lendaris, Stanley, and Rempfer for reviewing my work on many different starts over the years in both private and group presentations. Also, Professor Crittenden oversaw the general direction of the work and he checked the validity of definitions and proofs. Professor Lendaris edited and provided helpful suggestions for developing chapters I and VI. Professor Erdman pointed out how to cleanup one of the proofs.

I would like to thank Inge Infante and George C. Nicol (son) for being so patient and understanding during the long hours of research.

TABLE OF CONTENTS

Acknowledgementiii

Chapter

I. Introduction and Overview 1

II. The Multiprocess Scheduling System 8

 d-Processes and Descriptions 9

 s-Processes and Schedules 11

 Scheduling Functions 12

III. Spectral Theory 22

 Test Sequences 23

 The Spectral Map and Spectrum 25

 Spectral Sequences 26

 The Sequential Spectrum 30

IV. Scheduling Function Performance 32

 The Performance Measure 32

 The Performance Theorems 34

V. Independent Task Scheduling Performance Theorems . . 50

 Descriptions, Schedules, and Scheduling Functions. . 50

VI. Some Speculation 53

Bibliography 61

CHAPTER I

INTRODUCTION AND OVERVIEW

Many man-made systems are characterized by three properties: they are goal oriented; they use resources in the process of achieving their goals; and there are different choices available as to how and what resources are used. In such systems, it is often the case that how the resources are used determines the productivity of the system, the extent to which it is successful in achieving its goals, and in the long run, whether or not the system survives.

How resources are used is often a question of scheduling. For example, consider a manufacturing company whose goal is to make a profit by the production and sale of certain products. In such a company, the resources include workers, machines, and materials. Poor scheduling of manpower, machines and/or material could produce delays in production. Such delays would occur, for example, if the right materials are not at the right machines at the right time for assembly. Extensive ongoing delays may result in large order backlogs, long order lead times, and high production costs. This in turn may put the company at a competitive disadvantage and decrease the company's survival chances.

Another example is the processing of computer programs in a multiprogramming environment. Here, programs residing

in the computer's main memory take turns using the system's processor to do calculations. At each processing time, the scheduling function decides, according to some predetermined algorithm, which program gets the processor. Because of the high cost of large computer systems, it is often desirable to have a scheduling function producing schedules that result in high levels of throughput, i.e. the number of processed programs per unit of time.

Both of the above systems, although different in nature, rely on good scheduling to operate efficiently. The schedules are determined by a scheduling function. But the procedure the scheduling function uses may not be well defined. In the case of the manufacturing company, the shop foreman might schedule all of the machines using intuitive methods based on years of experience. In the computer system a program based on a well defined algorithm might be used to schedule the processor. In either case, relative to some predetermined performance criterion (like shortest possible schedule), the schedules being produced might not be the best possible. On the other hand, how is one to know whether or not the schedules are optimal without actually constructing better ones or showing that better ones cannot be constructed? Are there conditions under which scheduling function performance is optimal no matter which scheduling function is used? Can knowing the performance of a scheduling function give information about the schedules it produces? Under what conditions can scheduling performance be easily calculated?

The types of situations and questions alluded to in the preceding paragraph have various practical consequences. Before answering such questions, however, certain basic properties of scheduling functions and the underlying algorithms for generating their schedules must be investigated.

Historically, the general emphasis in the study of scheduling algorithms has been on individual algorithms rather than on classes of algorithms, but more recently this state of affairs has begun to change, as Reingold, Nievergelt, and Deo point out in [3]:

. . . one of the trends responsible for the rapid progress in combinatorial computing is a stronger emphasis on the study of classes of algorithms as opposed to individual algorithms.

Another line of research has been that of developing scheduling algorithms which are efficient in their operation (in the sense of computer run time) and which produce "reasonably good" schedules. Garey, Graham, and Johnson [2] comment:

Unfortunately, although it is not difficult to design optimization algorithms (e.g., exhaustive search is usually applicable), the goal of designing efficient optimization algorithms has proved much more difficult to attain. . . . This pessimistic outlook has been bolstered by recent results . . .

The "goodness" of these schedules is determined by comparing the length of the schedules generated by the algorithm to the length of some a priori determined optimal schedule. Furthermore, in [2], the "goodness" of an algorithm is determined by constructing a performance guarantee theorem which gives a least upper bound to the algorithm's worst case performance. This approach is applied on an individual algorithm basis,

and as pointed out in [2], works well on many algorithms. However, since the structure of the algorithm being studied is used to direct the construction of the performance theorem, the more complex the algorithm, the less likely it is for this approach to work. In any case, when this approach is successful, valuable information about individual algorithms is gained.

The approach taken in this dissertation is a combination and extension of the above two approaches. Classes of algorithms are the primary focus, and performance theorems are utilized in their analysis. A novel approach to considering the classes of algorithm is developed. This includes studying the general properties of the scheduling function, developing certain methods based on these, and then, by making a natural assumption of correspondence, applying these methods directly to the class of algorithms which compute the scheduling functions.

The underlying questions of concern in this dissertation are: What can be said in general concerning the performance of algorithms associated with a particular scheduling system? If general performance theorems exist for a given system, then what techniques are involved in the construction and proof of the theorems, of what value are the theorems in analysis of individual algorithms, and to what extent can the theorems and techniques be applied to different scheduling systems?

The contributions of this dissertation may be divided

into roughly three groups: (1) the development of a new mathematical model for simulating resource scheduling in multiprocessing systems, (2) the creation of a set of performance theorems for the class of scheduling functions belonging to the model, and (3) the transfer of the mathematical techniques and theorems from the model to another scheduling model.

The model developed here can be used to study the scheduling of reusable resources in a manufacturing environment. It can also be used to study the scheduling of a processor in a multiprogramming computer system. In chapter 2 it is shown that the model has a submodel which is mathematically equivalent to the model given in [1] which is used to study computer processor scheduling. In general, M, the model developed here, can be used to simulate any system having the following properties: (1) the system accomplishes its tasks by using resources from a finite set of reusable resources, (2) the system has a finite set of processes each of which provides a description of what resources it needs through time to accomplish its task, and (3) for each process set, selection of a schedule can be made from a variety of possible schedules. Each schedule gives a listing of what resources are used by which processes so that there are no resource conflicts (no two processes are assigned the same resource at the same time).

Each of the performance theorems created in this dissertation for M describes the conditions under which a

scheduling function exhibits a particular type of performance. For example, one of the theorems states that for each scheduling function there exist a significant number of cases where the scheduling function exhibits best case performance. Here, performance of a scheduling function f refers to the comparison between the length of f 's schedules and the length of the optimal schedules. One of the key insights underlying the work reported in this dissertation is that the performance theorems are important not only because they describe how scheduling functions behave but also because they describe how any algorithm used to compute the schedules of a scheduling function behaves. That is, if there is an algorithm for computing the schedules of a given scheduling function f , then this algorithm has the same performance properties as f . Thus in the example theorem given above, an algorithm which computes f 's schedules exhibits best case performance for those cases where f exhibits best case performance.

The third group of contributions, mentioned above, involves the transfer of the spectral theory techniques developed in chapter 3. It is demonstrated that these techniques are transferable to the Independent Task Scheduling model, ITS, given in [2]. This is done by redefining the ITS model using methods similar to those used to define M . The redefined ITS model is mathematically equivalent to the definition of ITS given in [2], but now it has the interesting property that all of the performance theorems and associated

lemmas and corollaries developed for M hold for ITS as well. This leads one to speculate on the possible existence of a general scheduling model, similar to M, that encompasses many of the standard scheduling models. This may be achieved in part by dropping and/or modifying the defining axioms of the model M.

CHAPTER II

THE MULTIPROCESS SCHEDULING SYSTEM

In this chapter, the multiprocess scheduling system, M , is developed. $M = (R, F, D, S)$. That is, M is a 4-tuple of sets where R , F , D , and S are, respectively, the set of resources, the set of scheduling functions, the set of descriptions, and the set of schedules. The motivation for M is given as the definitions are developed.

The following notation is presented first since it is used throughout this work. Other special notation is presented as needed in the remainder of this paper. Also, in a definition or notation, a word or phrase being defined is underlined.

Notation. Let $N = \{1, 2, 3, \dots\}$. For each m in N , let $N_m = \{1, 2, 3, \dots, m\}$. If d is an n -tuple, then the width of d , denoted $w(d)$, is the number n where n is from N . If S is a set, then S^c denotes the complement of S , \bar{S} denotes the cardinality of S , and $P(S)$ denotes the finite power set of S , that is, $P(S)$ is the set of all finite subsets of S . If p is a sequence in $P(S)$, that is, p is a function from N into $P(S)$, then the kernel of p , denoted $\ker(p)$, is the set $\{t \text{ in } N: p(t) = \emptyset\}$.

Let S_k and Q_k be sequences in N . S_k is eventually greater than Q_k , denoted $S_k \overset{e}{>} Q_k$, if there is an m in N so

that $S_k > Q_k$ whenever $k > m$. $\overset{e}{=}$, $\overset{e}{\geq}$, $\overset{e}{\leq}$, and $\overset{e}{<}$ are defined similarly.

Let $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ denote the greatest and smallest integer function respectively. $[\cdot]$ means that either $\lceil \cdot \rceil$ or $\lfloor \cdot \rfloor$ may be used with the restriction that once one is selected for a result then it is used exclusively throughout that result.

d-Processes and Descriptions

A d-process may be thought of as a technological description of, say, a manufacturing process. It describes the resources needed by the process at each time increment to produce some item or accomplish some task. A description denotes a set of processes which will be processed concurrently in the system. A description may be also thought of as a set of programs to be processed in a multiprogramming computer system with the individual programs representing the d-processes. Here, each program requires the computer processor at various time increments to do calculations.

Definition 2.1. Let $k \in \mathbb{N}$. A k-resource set, R_k , is the nonempty finite set $\{r_1, r_2, r_3, \dots, r_k\}$. The resource set, R , is the set $\bigcup_{k=1}^{\infty} R_k$. Elements of R are called resources. A sequence $p(t)$ in $P(R)$ is called a d-process if there is an s in \mathbb{N} such that $p(t) \neq \emptyset$ if $t \leq s$ and $p(t) = \emptyset$ if $t > s$. Such an s is called the stop-time for p . The set of all d-processes, denoted P , is called the process set. An n-tuple of d-processes is called a description if $n \geq 2$. The set of all

descriptions, denoted D , is called the description set. That is, $D = \bigcup_{n=2}^{\infty} P^n$.

Definition 2.2. Let d be a description. Say that $d = (d_1, d_2, \dots, d_n)$. The description length of d , denoted $\|d\|$, is the number $\max\{s_i : i \text{ in } N_n\}$ where s_i denotes the stop-time for the d -process d_i . Note that $\|d\| = \max\{\ker^c(d_i) : i \text{ in } N_n\}$.

As the following example shows, a description d is thought of as a matrix with a total number of rows equal to $\|d\|$.

Example 2.3. Let $d = (d_1, d_2)$ be the description given in the following diagram. The stop-times for d_1 and d_2 are 4 and 3 respectively. $\|d\| = 4$.

<u>time</u>	<u>d₁</u>	<u>d₂</u>
1	r_1	r_1
2	$r_2 r_4$	r_3
3	$r_1 r_3$	$r_2 r_4$
4	r_2	\emptyset

In this example, the two d -processes which makeup d are d_1 and d_2 . They will be processed concurrently in the multiprocessing system. Each requires a subset of the system's resources from the resource set $R_4 = \{r_1, r_2, r_3, r_4\}$ at each time increment. At time 1, d_1 and d_2 each need resource r_1 . At time 2, d_1 needs r_2 and r_4 while d_2 requires r_3 . For simplicity, subsets of resources are written in abbreviated form; d_1 at time 2 requires the subset $\{r_2, r_4\}$ which in the above diagram is written as $r_2 r_4$.

S-PROCESSES AND SCHEDULES

An s-process is the schedule of resources for some corresponding d-process. A set of s-processes is the schedule of resources for some description. As will be apparent from the following definitions, there may be many different schedules for a given description. The scheduling problem becomes the problem of selecting the 'best' schedule for a given description. The explicit definitions for s-processes and schedules are developed next.

Definition 2.4. A sequence $p(t)$ in $P(R)$ is called an s-process if there is an s in N such that $p(s) \neq \emptyset$ and $p(t) = \emptyset$ if $t > s$. Such an s is called the stop-time for p . The set of all s-processes, denoted Q , is called the s-process set.

Definition 2.5. An n-tuple of s-processes, say, (s_1, s_2, \dots, s_n) is a schedule if the following conditions hold:

- (1) $n \geq 2$
- (2) For t in N , if $r \in s_j(t)$ for some j in N_n , then $r \notin s_i(t)$ for each i in $N_n - \{j\}$
- (3) If there is an x in N so that $s_j(x) = \emptyset$ for each i in N_n , then $s_j(t) = \emptyset$ for each $t > x$ and i in N_n

The set of all schedules, denoted S , is called the schedule set. Evidently, $S \subset \bigcup_{n=2}^{\infty} Q^n$.

Definition 2.6. Let s be in S , say, $s = (s_1, s_2, \dots, s_n)$. The schedule length of s , denoted $\|s\|$, is the number $\max\{t_i : i \text{ in } N_n\}$ where t_i denotes the stop-time for the s-process s_i .

Note that, in general, $\|s\| \neq \max\{\overline{\overline{\ker^c(s_i)}} : i \text{ in } N_n\}$. $\|s\|$ is the number of rows in the schedule matrix of s .

Example 2.7. The following diagram is a schedule matrix for the schedule $s = (s_1, s_2)$. s is one way in which the resources of description d in example 2.3 may be scheduled. Here, $\|s\| = 5$. Notice that there are no resource conflicts, i.e., two processes using the same resource at the same time. This is guaranteed by (2) of definition 2.5. Also, (3) guarantees that if a row in a schedule contains all empty sets, then the schedule is empty for all future times.

<u>Time</u>	<u>s₁</u>	<u>s₂</u>
1	r ₁	∅
2	r ₂ r ₄	r ₁
3	r ₁ r ₃	∅
4	r ₂	r ₃
5	∅	r ₂ r ₄

SCHEDULING FUNCTIONS

The concept of a scheduling function is developed next. This concept provides the mechanism by which schedules are assigned to descriptions. The class of all scheduling functions is central in this paper. It is this class for which the performance theorems are developed. As it turns out, if s is a schedule for d , then there is a scheduling function f so that $f(d) = s$.

Definition 2.8. Let d be in D , say, $d = (d_1, d_2, \dots, d_n)$. A function f from D into S is a scheduling function if the

following conditions hold:

(1) $w(d) = w(fd)$, i.e., f operating on d may be thought of as f operating on each individual coordinate of d ; thus,

$$f(d) = f(d_1, d_2, \dots, d_n) = (fd_1, fd_2, \dots, fd_n)$$

$$(2) \overline{\ker^c(d_j)} = \overline{\ker^c(fd_j)} \text{ for each } j \text{ in } N_n$$

(3) For each j in N_n , $d_j(t) = fd_j(t')$ for each t in $\ker^c(d_j)$ where $'$ is the natural order preserving bijection from $\ker^c(d_j)$ into $\ker^c(fd_j)$

Property (1) states that the number of processes in the description equals the number of processes in the schedule. Property (2) states that the number of nonvoid steps in a process equals the number of nonvoid steps in the schedule of that process. Property (3) states that resource splitting is not allowed as the following example demonstrates.

Example 2.9. Let $a, b, c,$ and x be in R . Let f be a scheduling function. Define $d, s,$ and s' as follows:

$$d = \begin{array}{cc} ab & a \\ c & b \\ x & c \end{array}, \quad s = \begin{array}{cc} ab & \emptyset \\ c & a \\ x & b \\ \emptyset & c \end{array}, \quad \text{and } s' = \begin{array}{cc} a & \emptyset \\ bc & a \\ x & b \\ \emptyset & c \end{array}; \text{ then, } s \text{ is an allow-}$$

able image for d under f . Also, s' is not an allowable image for d under f although s' is a schedule.

Definition 2.10. Let F denote the set of all scheduling functions. The multiprocess scheduling system, M , is the 4-tuple (R, F, D, S) .

Definition 2.11. Let d be in D . The max-length of d , denoted $\|d\|$, is the number $\max\{\|f(d)\| : f \text{ is in } F\}$. $\|d\|$ is the longest possible schedule length derivable from d . Also, $\|d\|$

$$= \sum_{i=1}^n \overline{\ker^c(d_i)} \text{ where } w(d) = n.$$

There are three subsets of F which partition F into three pairwise disjoint sets. The sets are the severe scheduling functions, the optimal scheduling functions, and the intermediate scheduling functions, denoted respectively, SF , OF , and IF . These sets are defined next.

Definition 2.12. f in F is a severe scheduling function if $\|f(d)\| = \|d\|$ for each d in D . SF is used to denote the set of all severe scheduling functions. Unless noted otherwise SEV is used to denote an arbitrary but fixed element of SF .

Example 2.13. Let f be in F . Let d be in D , say, $d = (d_1, d_2, \dots, d_n)$ where s_i is the stop-time of d_i . Let j_1, j_2, \dots, j_n be in N_n and let them be n distinct indicies. Then f is a string scheduling function if

$$\begin{aligned} fd_1(t) &= dj_1(t) \\ fd_2(t) &= \begin{cases} \emptyset & \text{if } 1 \leq t \leq sj_1 \\ dj_2(t - sj_1) & \text{otherwise} \end{cases} \\ fd_3(t) &= \begin{cases} \emptyset & \text{if } 1 \leq t \leq sj_1 + sj_2 \\ dj_3(t - (sj_1 + sj_2)) & \text{otherwise} \end{cases} \\ &\vdots \\ fd_n(t) &= \begin{cases} \emptyset & \text{if } 1 \leq t \leq sj_1 + \dots + sj_{n-1} \\ dj_n(t - (sj_1 + sj_2 + \dots + sj_{n-1})) & \text{otherwise} \end{cases} \end{aligned}$$

For example, let $a, b,$ and c be in R . Let f be in F and let

$$d = \begin{array}{ccc} a & a & c \\ b & c & c \\ c & \emptyset & ab \end{array}, \text{ then if } f \text{ is a string scheduling function, } f(d)$$

might be either of the following where $\|d\| = 3$ and $\|f(d)\| = 8 = \|d\|$:

$Q = (d^k)_{k=1}$ is given as:

<u>Time</u>	<u>d_1^k</u>	<u>d_2^k</u>
1	$r_{1 \ 1}$	$r_{1 \ 1}$
2	$r_{1 \ 2}$	$r_{1 \ 2}$
3	$r_{1 \ 3}$	$r_{1 \ 3}$
\vdots	\vdots	\vdots
v	$r_{1 \ v}$	$r_{1 \ v}$
$v+1$	$r_{2 \ 1}$	$r_{2 \ 1}$
$v+2$	$r_{2 \ 2}$	$r_{2 \ 2}$
$v+3$	$r_{2 \ 3}$	$r_{2 \ 3}$
\vdots	\vdots	\vdots
$2v$	$r_{2 \ v}$	$r_{2 \ v}$
\vdots	\vdots	\vdots
$(k-1)v$	$r_{k-1 \ v}$	$r_{k-1 \ v}$
$(k-1)v+1$	$r_{k \ 1}$	$r_{k \ 1}$
$(k-1)v+2$	$r_{k \ 2}$	$r_{k \ 2}$
$(k-1)v+3$	$r_{k \ 3}$	$r_{k \ 3}$
\vdots	\vdots	\vdots
kv	$r_{k \ v}$	$r_{k \ v}$
$kv+1$	$r_{k+1 \ 1}$	
$kv+2$	$r_{k+1 \ 2}$	
$kv+3$	$r_{k+1 \ 3}$	
\vdots	\vdots	
$(k+1)v$	$r_{k+1 \ v}$	

Throughout this example, v is some arbitrary but fixed element of N . For each choice of v , there is a different sequence

Q.

OPT(Q) maybe given as follows:

<u>Time</u>	<u>OPT(d₁^k)</u>	<u>OPT(d₂^k)</u>
1	r ₁ 1	
2	r ₁ 2	
3	r ₁ 3	
⋮	⋮	
v	r ₁ v	
v+1	r ₂ 1	r ₁ 1
v+2	r ₂ 2	r ₁ 2
v+3	r ₂ 3	r ₁ 3
⋮	⋮	⋮
2v	r ₂ v	r ₁ v
2v+1	r ₃ 1	r ₂ 1
2v+2	r ₃ 2	r ₂ 2
2v+3	r ₃ 3	r ₂ 3
⋮	⋮	⋮
3v	r ₃ v	r ₂ v
⋮	⋮	⋮
⋮	r _k 1	r _{k-1} 1
⋮	r _k 2	r _{k-1} 2
⋮	r _k 3	r _{k-1} 3
⋮	⋮	⋮
kv	r _k v	r _{k-1} v
⋮	r _{k+1} 1	r _k 1
⋮	r _{k+1} 2	r _k 2
⋮	r _{k+1} 3	r _k 3
⋮	⋮	⋮
(k+1)v	r _{k+1} v	r _k v

Finally, let f be in F so that $f(Q)$ is given as:

<u>Time</u>	<u>$f(d_1^k)$</u>	<u>$f(d_2^k)$</u>
1	$r_1 \ 1$	
2		$r_1 \ 1$
3	$r_1 \ 2$	
4	\vdots	$r_1 \ 2$
\vdots	$r_1 \ v$	\vdots
$2v$	$r_2 \ 1$	$r_1 \ v$
$2v+1$		$r_2 \ 1$
$2v+2$	$r_2 \ 2$	
\vdots		$r_2 \ 2$
\vdots	$r_2 \ 3$	
\vdots	\vdots	$r_2 \ 3$
$2v+(2v-1)$	$r_3 \ 1$	\vdots
\vdots		$r_2 \ v$
\vdots		$r_3 \ 1$
\vdots	$r_3 \ 2$	
\vdots	\vdots	$r_3 \ 2$
$2v+(k-2)(2v-1)$	$r_k \ 1$	\vdots
\vdots		$r_{k-1} \ v$
\vdots		$r_k \ 1$
\vdots	$r_k \ 2$	
\vdots	\vdots	$r_k \ 2$
$2v+(k-1)(2v-1)$	$r_{k+1} \ 1$	\vdots
\vdots		$r_k \ v$
\vdots	$r_{k+1} \ 2$	
\vdots	$r_{k+1} \ 3$	
\vdots	\vdots	
$2v+(k-1)(2v-1)+v-1$	$r_{k+1} \ v$	

In the previous example, the following inequalities hold: $\|d^k\| = \|\text{OPT}(d^k)\| = (k+1)v = kv+v \leq k(2v-1)+v = 2v+(k-1)(2v-1)+v-1 = \|f(d^k)\| = 2(k+1)v-(k+1)-v+1 = 2\|d^k\|-(k+1)-v+1 \leq \|d^k\| = \|\text{SEV}(d^k)\| = (k+1)v+kv = 2(k+1)v-v = 2\|d^k\|-v \leq 2\|d^k\| = w(d^k)\|d^k\|.$

In the next example, it is shown that M has a subsystem, M' , which is equivalent to the model given in [1] for studying the scheduling of a processor in a multiprogramming computer system.

Example 2.17. In [1], a program P_i is defined to be a finite sequence of integers $T_{i1}, t_{i1}, T_{i2}, t_{i2}, \dots, t_{in_i-1}, T_{in_i}$ where $t_{ij} > 0$ for $j \leq n_i-1$; $T_{ij} > 0$ for $1 < j < n_i$; and $T_{ij} \geq 0$ for $j = 1$ or $j = n_i$. The T_{ij} 's are called compute times and the t_{ij} 's are the wait times, the times when the program is in an I/O state or simply waiting for the processor. A program is a fixed sequence of compute and wait times.

A multiprogramming description consists of k programs being processed by one processor where the processor is assigned to one program at a time in increments of one unit of time. After program P_i has been assigned the processor for T_{i1} units of time P_i must wait t_{i1} units of time regardless of whether or not the processor is free. After t_{i1} units of time, P_i may again compete with the other programs for the processor. After being assigned T_{i2} units of processor time, it goes into waiting again. While P_i has the processor for a compute period no other program may use it, although, P_i 's compute period may be pre-empted. But when P_i again gets the processor, it picks

up where processing left off. For example, consider the following description: let P_1 be given by $T_{11} = 3$, $t_{11} = 1$, $T_{12} = 1$; and P_2 be given by $T_{21} = 1$, $t_{21} = 2$, and $T_{23} = 2$. The description $d = (P_1, P_2)$ may be represented graphically as follows:

$$P_1: \text{---} \cdot \text{---}$$

$$P_2: \text{---} \cdot \cdot \text{---}$$

where the dashes and dotes represent, respectively, compute and wait times.

Let M' be the subsystem of M defined as follows: first, associate $R' = \{p, w_1, w_2, w_3, \dots\}$ with R ; let D' be the restriction of D to all those descriptions d so that w_i in $d_j(t)$ implies that $i = j$, and $d_j(t) \neq \emptyset$ implies $d_j(t)$ is a singleton set; do the same to S to get S' ; and, finally, restrict each f in F to D' to get F' . Let $M' = (R', F', D', S')$, then M' is a multiprocess scheduling subsystem of M . By associating the p 's and w 's of M' with the dashes (---) and dotes (\cdot) of the other system, the equivalence of the two systems follows. As an example, d given above may be represented in M' as follows:

Time	P_1	P_2
1	p	p
2	p	w_2
3	p	w_2
4	w_1	p
5	p	p

Here, w_1 and w_2 are the wait symbols for P_1 and P_2 respectively.

and p represents the computer processor.

One possible schedule for d is as follows:

sP_1 : _ _ _ . _

sP_2 : _ . . _ _

The equivalent schedule in terms of M' is as follows:

Time	sP_1	sP_2
1	p	\emptyset
2	\emptyset	p
3	p	w_2
4	p	w_2
5	w_1	p
6	p	\emptyset
7	\emptyset	p

CHAPTER III

SPECTRAL THEORY

The theory developed here deals with special structures of the scheduling function domain set, the test sequences, and the range of possible lengths the schedules of the descriptions may have. The concepts of test sequences, spectral sequences, and the sequential spectrum of a scheduling function are fundamental to the construction of the performance theorems given in chapter 4.

The test sequences are developed first since the spectral theory is built from them. Test sequences are also central in constructing the performance theorems since they can bring out the best or worst performance of a scheduling function.

The spectrum of a test sequence is developed next. From this concept, the spectral sequences are constructed. The structure and properties of these sequences are then studied, for they also play an important role in the construction of the performance theorems.

Finally, the sequential spectrum of a scheduling function is defined and certain properties concerning it are given. This concept allows one to consider a scheduling function in terms of spectral sequences. This permits the use of certain spectral sequence results in the analysis of the scheduling functions.

TEST SEQUENCES

Throughout this section, let $Q = (d^k)_{k=1}^{\infty}$ be a sequence in D , the set of all descriptions.

Definition 3.1. Q is of constant width if, for some fixed i in N with $i > 1$, $w(d^k) = i$ for each k in N . In this case, the width of Q , denoted $w(Q)$, is i .

Definition 3.2. Q is a test sequence if the following conditions hold:

- (1) Q is of constant width
- (2) $\|d^{k+1}\| > \|d^k\|$ for each k in N
- (3) $\|d^k\| \div \|d^k\| \rightarrow w(Q)$ as $k \rightarrow \infty$

Notation. Let T denote the set of all test sequences in D .

In the next example, a sequence Q of D is given which is also a test sequence.

Example 3.3. Define Q as follows: for each k in N , let d^k be given as

$$\begin{array}{cc} a_1 & b_1 \\ a_2 & b_2 \\ a_3 & b_3 \\ \vdots & \vdots \\ a_k & b_k \\ \vdots & \\ a_{k+100} & \end{array}$$

Here, $w(Q) = 2$, $\|d^{k+1}\| = k+101 > k+100 = \|d^k\|$ for each k in N , and $\|d^k\| \div \|d^k\| = 2k+100 \div k+100 = 1 + (1/(1+(100 \div k))) \rightarrow 2$.

That is, Q is in T .

In the next example, a sequence Q in D is given which is not a test sequence.

Example 3.4. Define Q as follows: for each k in N , let d^k be given as

$$\begin{array}{ll} a_1 & b_1 \\ a_2 & b_2 \\ a_3 & b_3 \\ \vdots & \vdots \\ a_k & b_k \\ \vdots & \\ a_{2k} & \end{array}$$

Here, $w(Q) = 2$, $\|d^{k+1}\| = 2(k+1) > 2k = \|d^k\|$ for each k in N , but $\|d^k\| \div \|d^k\| = 3k \div 2k = 3/2 \neq 2$. Therefore, Q is not in T .

Example 3.5. The sequence Q in example 2.16 is in T . since $w(Q) = 2$, $\|d^{k+1}\| = (k+2)v > (k+1)v = \|d^k\|$ for each k in N , and $(\|d^k\| \div \|d^k\|) = (k+1)v + kv \div (k+1)v = 1 + (k - (k+1)) \rightarrow 2$. That is, Q is in T .

Notation. Let $|d|$ denote $\|d\| - \|d\|$.

The next result shows that if Q is a test sequence, then the difference between the lengths of the longest and the shortest possible description schedules gets arbitrarily large as the sequence of descriptions progresses.

Proposition 3.6. If Q is in T , then $|d^k| \rightarrow \infty$.

Proof. Since Q is in T , $w(Q) > 1$. Thus, $w(Q) - 1 > 0$ and $w(Q) - 1 - \epsilon > 0$ for some small $\epsilon > 0$. Since $\|d^k\| \div \|d^k\| \rightarrow w(Q)$, pick q in N so that if $k > q$, then $|(\|d^k\| \div \|d^k\|) - w(Q)| < \epsilon$. So, if $k > q$, then $-\epsilon < \|d^k\| \div \|d^k\| - w(Q)$, $(w(Q) - \epsilon) \|d^k\| < \|d^k\|$,

and $0 < (w(Q)-e-1)\|d^k\| < \|d^k\| - \|d^k\|$. Since $w(Q)-e-1$ is fixed and $\|d^k\| \rightarrow \infty$, then $|d^k| = \|d^k\| - \|d^k\| > \|d^k\|(w(Q)-e-1) \rightarrow \infty$.

Example 3.7. In example 3.3, $|d^k| = 2k+100-(k+100) = k \rightarrow \infty$. In example 3.4, $|d^k| = 3k-2k = k \rightarrow \infty$. In example 2.16, $|d^k| = (k+1)v+kv-(k+1)v = kv \rightarrow \infty$.

In the next result, properties regarding the limits of test sequence schedule lengths are given.

Proposition 3.8. Let Q be in T , then (1) $\|d^k\| \rightarrow \infty$, (2) $\|d^k\| \rightarrow \infty$, (3) $\|f(d^k)\| \rightarrow \infty$ for each f in F , and (4) $|d^k| \div \|d^k\| \rightarrow w(Q)-1$.

Proof. (1) Since $\|d^k\|$ is in N for each k in N and since $\|d^{k+1}\| > \|d^k\|$ for each k in N , then $\|d^k\| \rightarrow \infty$. (2) Since $\|d^k\| > \|d^k\|$, $\|d^k\| \rightarrow \infty$. (3) $\|f(d^k)\| \geq \|OPT(d^k)\| \geq \|d^k\| \rightarrow \infty$. (4) $|d^k| \div \|d^k\| = (\|d^k\| - \|d^k\|) \div \|d^k\| = (\|d^k\| \div \|d^k\|) - 1 \rightarrow w(Q) - 1$.

THE SPECTRAL MAP AND SPECTRUM

Definition 3.9. The spectral map is the function m from D into $\bigcup_{n=1}^{\infty} N^n$ such that

$$m(d) = (\|d\|, \|d\|+1, \|d\|+2, \dots, \|d\|-2, \|d\|-1, \|d\|)$$

$m(d)$ is the spectrum of d .

Definition 3.10. Let $Q = (d^k)_{k=1}^{\infty}$ be a sequence in D . The spectrum of Q , denoted $m(Q)$ or $m(d^k)$, is the sequence $(m(d^k))_{k=1}^{\infty}$.

Example 3.11. Consider the sequence Q of 2.16 where $v = 3$. The spectrum of Q , $m(Q)$ is given in the following diagram:

$$m(d^1) = (6, 7, 8, 9)$$

$$m(d^2) = (9, 10, 11, 12, 13, 14, 15)$$

$$m(d^3) = (12, 13, 14, 15, 16, 17, 18, 19, 20, 21)$$

$$m(d^4) = (15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27)$$

$$m(d^5) = (18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33)$$

$$\vdots \quad \quad \quad \vdots$$

$$m(d^k) = (3k+3, 3k+4, 3k+5, 3k+6, 3k+7, 3k+8, \dots, 6k, 6k+1, 6k+2, 6k+3)$$

The above array is called the spectral diagram of Q .

SPECTRAL SEQUENCES

Throughout this section, let $Q = (d^k)_{k=1}^{\infty}$ be a test sequence in D , n an element of N , and r an element of $(0, 1)$.

Definition 3.12. A sequence $s(k)$ is a spectral sequence of Q if $s(k)$ is in $m(d^k)$ for each k in N .

Definition 3.13. A spectral sequence $s(k)$ of Q is:

- (1) n^{th} -left if $s(k) \in \parallel d^k \parallel + n - 1$
- (2) n^{th} -right if $s(k) \in \lll d^k \lll - n + 1$
- (3) r^{th} -left intermediate if $s(k) \in \parallel d^k \parallel + [r \mid d^k \mid]$
- (4) r^{th} -right intermediate if $s(k) \in \lll d^k \lll - [r \mid d^k \mid]$

Notation. Let $S(Q), S_L(Q), S_R(Q), S_{LI}(Q), S_{RI}(Q)$, and $S_I(Q)$ denote, respectively, the set of all, left, right, left-intermediate, right-intermediate, and intermediate spectral sequences of Q . $S_I(Q)$ is defined to be $S_{LI}(Q) \cup S_{RI}(Q)$.

Example 3.14. In example 3.11, the sequence $s(k) = \parallel d^k \parallel + 2 - 1 = 3k + 4$ is a 2^{nd} -left spectral sequence. $s(k) = \lll d^k \lll -$

$2+1 = 6k+2$ is a 2^{nd} -right spectral sequence. The sequence $s(k) = \|d^k\| + \lceil \frac{1}{2} |d^k| \rceil = 3k+3 + \lceil \frac{1}{2} (6k+3 - (3k+3)) \rceil = 3k+3 + \lceil 3k/2 \rceil = 5k+3$ is a $\frac{1}{2}$ -left intermediate spectral sequence.

Definition 3.15. Let X and V be sets of sequences in N . X is eventually contained in V , denoted $X \overset{e}{\subset} V$, if for each x in S there is a y in V so that $x \overset{e}{=} y$. X is eventually equal to V , denoted $X \overset{e}{=} V$, if $X \overset{e}{\subset} V$ and $V \overset{e}{\supset} X$.

Example 3.16. For each k in N , let x_k be the sequence $1, 1, 1, \dots, 1, 1, \dots, 0, 0, 0, \dots$ where 1 appears k times and the 0's continue forever. Let y be the zero sequence. Let X be the set $\{x_k : k \text{ is in } N\}$. Let V be the singleton set $\{y\}$. Then, $X \overset{e}{=} V$.

The next proposition shows that $S_{LI}(Q)$ is eventually equal to $S_{RI}(Q)$. Thus, as far as the performance theorems are concerned, it will necessary in the future to consider only, say, the left-intermediate spectral sequences.

Proposition 3.17. $S_{LI}(Q) \overset{e}{=} S_{RI}(Q)$ for each Q in T .

Proof. Let s be in $S_{RI}(Q)$, say, $s(k) \overset{e}{=} \|d^k\| - \lceil r |d^k| \rceil$. choose $s = 1-r$. So $\|d^k\| + \lceil s |d^k| \rceil$ is in $S_{LI}(Q)$. Therefore, $s(k) \overset{e}{=} \|d^k\| - \lceil r |d^k| \rceil = \|d^k\| + \|d^k\| - \|d^k\| - \lceil r |d^k| \rceil = \|d^k\| + |d^k| - \lceil r |d^k| \rceil = \|d^k\| + \lceil |d^k| - r |d^k| \rceil = \|d^k\| + \lceil s |d^k| \rceil$ is in $S_{LI}(Q)$. That is, $S_{RI}(Q) \overset{e}{\subset} S_{LI}(Q)$. The proof that containment holds in reverse is just as straightforward.

One of the main features of the system $M = (R, F, D, S)$ is the set F of scheduling functions. The next two results give the size of F and the size of the spectral sequence sets.

Proposition 3.18. Every test sequence has denumerably many left spectral sequences, denumerably many right spectral sequences, and a continuum of intermediate spectral sequences.

Proof. By considering the spectral diagram for a given test sequence, it is clear that every test sequence left and right spectral sequences since $|d^k| \rightarrow \infty$.

Let s be in $(0,1)$ so that $r < s$. Since $|d^k| \rightarrow \infty$, $r \leq s-2/|d^k|$, $r|d^k| \leq s|d^k|-2$, and $[r|d^k|] \leq r|d^k|+1 \leq s|d^k|-1 \leq [s|d^k|]$. Let $x(k) = \|d^k\| + [r|d^k|]$ and $y(k) = \|d^k\| + [s|d^k|]$, then x and y are in $S_I(Q)$ and $x \leq y$, i.e., $x \not\geq y$. Therefore, for each r in $(0,1)$, there is an x_r in $S_I(Q)$ so that $x_r \leq y_s$ for each y_s in $S_I(Q)$ such that $r < s$. That is, there is a continuum of intermediate sequences.

Corollary 3.19. There is a continuum of scheduling functions, i.e., $\overline{F} \geq c$.

Proof. For each i in N_k where k is in N , let $a_i = a \neq b = b_i$. Let d^k be given as

$$\begin{array}{cc} a_1 & b_1 \\ a_2 & b_2 \\ a_3 & b_3 \\ a_4 & b_4 \\ \vdots & \vdots \\ a_k & b_k \end{array}$$

Then d^k is a test sequence. Clearly, $\|OPT(d^k)\| = \|d^k\|$. Let f be in F so that $\|f(d^k)\| = \|d^k\| + [r|d^k|]$. Since there is a continuum of choices for r , $\overline{F} \geq c$.

The next proposition shows that forming new spectral sequences by combining a left and right spectral sequence does not create a new spectral sequence type. The following notation details how the combining takes place. It is similar to the method used in constructing left and right spectral sequences.

Notation. Let $S_I^!(Q)$ be the subset of $S(Q)$ formed as follows: I is in $S_I^!(Q)$ if $I(k) \stackrel{e}{=} \|d^k\|_{+n-1} + [s(|d^k| - m - n + 2)]$ where, for some s in $(0, 1)$, L in $S_I(Q)$, and R in $S_R(Q)$, $L(k) \stackrel{e}{=} \|d^k\|_{+n-1}$ and $R(k) \stackrel{e}{=} \|d^k\|_{-m+1}$.

Proposition 3.20. For each I in $S_I^!(Q)$ there is an H and J in $S_I(Q)$ such that $H \stackrel{e}{\leq} I \stackrel{e}{\leq} J$.

Proof. Pick r in $(0, 1)$ so that $r < s$. Since $|d^k| \rightarrow \infty$, $r \stackrel{e}{\leq} s + (n - sm - sn + 2s - 3) / |d^k|$. Thus, $[r|d^k|] \stackrel{e}{\leq} r|d^k| + 1 \stackrel{e}{\leq} s|d^k| + n - sm - sn + 2s - 2 \stackrel{e}{\leq} n - 1 + [s|d^k| - sm - sn + 2s]$. Choose $H(k)$ to be $\|d^k\|_{+} [r|d^k|]$. Thus, H is in $S_I(Q)$. So $H(k) = \|d^k\|_{+} [r|d^k|] \stackrel{e}{\leq} \|d^k\|_{+} n - 1 + [s(|d^k| - m - n + 2)] = I(k)$, i.e., $H \stackrel{e}{\leq} I$.

Pick t in $(0, 1)$ so that $s < t$. As above, $s + (n - sn - sm + 2s + 1) / |d^k| \stackrel{e}{\leq} t$. So, $n - 1 + [s(|d^k| - m - n + 2)] = n + s(|d^k| - m - n + 2) \stackrel{e}{\leq} t|d^k| - 1 = [t|d^k|]$. Choose $J(k)$ to be $\|d^k\|_{+} [t|d^k|]$. So, J is in $S_I(Q)$. Thus, $J(k) = \|d^k\|_{+} [t|d^k|] \stackrel{e}{\geq} \|d^k\|_{+} n - 1 + [s(|d^k| - m - n + 2)] = I(k)$. Therefore, $H \stackrel{e}{\leq} I \stackrel{e}{\leq} J$.

In the next proposition, it is shown that if H and J are two intermediate spectral sequences such that H is eventually less than J , then $H+m$ is eventually between H and J for each natural number m .

Proposition 3.21. Let H and J be in $S_I(Q)$. If $H \stackrel{e}{\leq} J$, then $H \stackrel{e}{\leq} H+m \stackrel{e}{\leq} J$ for each m in N .

Proof. Suppose $H(k) \stackrel{e}{=} \|d^k\| + [r|d^k|]$ and $J(k) \stackrel{e}{=} \|d^k\| + [s|d^k|]$. Suppose $H \stackrel{e}{\leq} J$. Then $r < s$. So $r+(m+2)/|d^k| \stackrel{e}{\leq} s$ and $r|d^k|+m+2 \stackrel{e}{\leq} s|d^k|$. Therefore, $H(k) \stackrel{e}{=} \|d^k\| + [r|d^k|] \stackrel{e}{\leq} \|d^k\| + [r|d^k|+m] \stackrel{e}{=} H(k)+m \stackrel{e}{=} \|d^k\| + r|d^k| + m + 1 \stackrel{e}{\leq} \|d^k\| + s|d^k| - 1 \stackrel{e}{=} \|d^k\| + [s|d^k|] \stackrel{e}{=} J(k)$.

The next result shows that each intermediate spectral sequence is eventually greater than each left spectral sequence and eventually less than each right spectral sequence.

Proposition 3.22. If L is in $S_L(Q)$, R is in $S_R(Q)$, and I is in $S_I(Q)$, then $L \stackrel{e}{\leq} I \stackrel{e}{\leq} R$.

Proof. Suppose that $L(k) \stackrel{e}{=} \|d^k\| + n - 1$, $I(k) \stackrel{e}{=} \|d^k\| + [r|d^k|]$, and $R(k) \stackrel{e}{=} \|d^k\| - m + 1$. Since $|d^k| \rightarrow \infty$, $n/|d^k| \stackrel{e}{\leq} r$ and $m/|d^k| \stackrel{e}{\leq} 1 - r$. So, $n - 1 \stackrel{e}{\leq} r|d^k| - 1 = [r|d^k|]$ and $\|d^k\| + n - 1 \stackrel{e}{\leq} \|d^k\| + [r|d^k|]$. Also, $m \stackrel{e}{\leq} |d^k| - r|d^k| = \|d^k\| - \|d^k\| - r|d^k|$. Therefore, $\|d^k\| + [r|d^k|] \stackrel{e}{=} \|d^k\| + r|d^k| + 1 \stackrel{e}{\leq} \|d^k\| - m + 1$. Thus, $L \stackrel{e}{\leq} I \stackrel{e}{\leq} R$.

THE SEQUENTIAL SPECTRUM

Definition 3.23. Let f be in F . Let $Q = (d^k)_{k=1}^{\infty}$ be a sequence in D . The sequential spectrum of f over Q , denoted $\|f(Q)\|$, is the sequence $(\|f(d^k)\|)_{k=1}^{\infty}$. Evidently, $\|f(d^k)\|$ is in $m(d^k)$ for each k in N .

Definition 3.24. Let Q be in T . Q is a left (intermediate, right) test sequence if $\|OPT(Q)\|$ is in $S_L(Q)$ ($S_I(Q)$, $S_R(Q)$).

Notation. Let T_L , T_I , and T_R denote, respectively, the sets of left, intermediate, and right test sequences of D .

The next result states that the sequential spectrum of a scheduling function over a right test sequence cannot be a left or an intermediate spectral sequence.

Proposition 3.25. Let f be in F and $Q = (d^k)_{k=1}^{\infty}$ be in T_R . Then $\|f(Q)\|$ is not in $S_L(Q) \cup S_I(Q)$.

Proof. $\|OPT(d^k)\| \leq \|f(d^k)\|$ for each k in N . Therefore, the result follows from proposition 3.22.

The next result states that the sequential spectrum of a scheduling function over an intermediate test sequence cannot be a left spectral sequence.

Proposition 3.26. Let f be in F and $Q = (d^k)_{k=1}^{\infty}$ be in T_I . Then $\|f(Q)\|$ is not in $S_L(Q)$.

Proof. This result also follows from proposition 3.22.

Example 3.27. The test sequence Q in example 2.16 is a left test sequence since $\|OPT(d^k)\| = (k+1)v = \|d^k\|$.

CHAPTER IV

SCHEDULING FUNCTION PERFORMANCE

In this chapter, the notion of scheduling function performance is defined and a set of performance theorems is constructed for the set F of scheduling functions. Each of the performance theorems describe the conditions under which a scheduling function exhibits a particular type of performance. Also, results are given to demonstrate that under certain conditions scheduling function performance is easy to calculate.

THE PERFORMANCE MEASURE

In this section, let f be in F , d in D , $Q = (d^k)_{k=1}^{\infty}$ in T , and $W(Q) = i$.

Definition 4.1. The performance of f at d , denoted $pf(d)$, is the number $\|f(d)\|/w(d)\|OPT(d)\|$.

Proposition 4.2. $1/w(d) \leq pf(d) \leq 1$.

Proof. Since $1 \leq \|d\| \leq \|OPT(d)\| \leq \|f(d)\| \leq w(d)\|d\| \leq w(d)\|OPT(d)\|$, $pf(d) = \|f(d)\|/w(d)\|OPT(d)\| \leq 1$ and $1 \leq \|f(d)\| \div \|OPT(d)\|$. Since $1 < w(d)$, $0 < 1/w(d) < 1$. Therefore, $1/w(d) \leq \|f(d)\|/w(d)\|OPT(d)\| = pf(d)$.

Proposition 4.3. $pf(d) = 1/w(d)$ iff $\|f(d)\| = \|OPT(d)\|$.

Proof. $pf(d) = 1/w(d)$ iff $\|f(d)\|/w(d)\|OPT(d)\| = 1/w(d)$ iff $\|f(d)\| = \|OPT(d)\|$.

Proposition 4.4. $pf(d) = 1$ iff $\|d\| = \|OPT(d)\|$ and $\|f(d)\| = w(d)\|d\|$.

Proof. $pf(d) = 1$ implies $\|f(d)\| / w(d)\|OPT(d)\| = 1$. So, $\|f(d)\| = w(d)\|OPT(d)\|$ and $\|f(d)\| = w(d)\|d\| = w(d)\|OPT(d)\|$ since $\|f(d)\| \leq w(d)\|d\| \leq w(d)\|OPT(d)\|$. Thus, $\|d\| = \|OPT(d)\|$. The reverse direction is trivial.

Next the definition of performance over a test sequence is given and the basic types of performance are defined.

Definition 4.5. The performance of f over Q, denoted $pf(Q)$, is the limit, if it exists, $\lim pf(d^k)$ for k in N .

Definition 4.6. The three basic performance types are defined as follows:

(1) f exhibits best case performance (e.b.c.p.) over Q if $pf(Q) = 1/w(Q)$

(2) f exhibits intermediate case performance (e.i.c.p.) over Q if $1/w(Q) < pf(Q) < 1$

(3) f exhibits worst case performance (e.w.c.p.) over Q if $pf(Q) = 1$

Example 4.7. From example 2.16, $\|f(d^k)\| / w(Q)\|OPT(d^k)\| = (2v+(k-1)(2v-1)+v-1)/2(k+1)v = (1+1/2k)/(1+1/k) - 1/(2v+2v/k) \rightarrow 1 - 1/2v$ as $k \rightarrow \infty$. Thus, $pf(Q) = 1 - 1/2v$ is in $(\frac{1}{2}, 1)$ for each v in N such that $v > 1$. Therefore, f e.b.c.p. over Q for $v = 1$ and f e.i.c.p. over Q for $v = 2, 3, 4, \dots$

Relative to the three basic kinds of test sequences, namely, left, intermediate, and right, under what conditions will a scheduling function exhibit each of the basic types of performance? This is the fundamental performance problem.

THE PERFORMANCE THEOREMS

In this section, the fundamental performance problem is solved; that is, for each test sequence type, the conditions that a scheduling function must satisfy in order to exhibit each type of performance are stated in the six performance theorems given in this section.

Notation. In this section, let f be a scheduling function, $Q = (d^k)_{k=1}^{\infty}$ be a test sequence, and $w(Q) = i$. Let Q' denote the quotient $1/i\|OPT(d^k)\|$. Thus $Q'I(k)$ is a shorthand way to write $I(k)/i\|OPT(d^k)\|$. Let Q'' denote $\|OPT(d^k)\|$.

The following lemma gives the performance of f over a left test sequence if the sequential spectrum of f is an intermediate spectral sequence. Thus, in this case, if the sequential spectrum of f is known, then the performance of f is easy to calculate.

Lemma 4.8. Let Q be in T_L and I be in $S_I(Q)$, say, $I(k) \cong \|d^k\| + [r|d^k|]$ for some r in $(0,1)$, then $\lim_{k \leftarrow N} Q'I(k) = \lim_{k \leftarrow N} Q'(\|d^k\| + [r|d^k|]) = (i+r(i-1))/i$.

Proof. Since Q is in T_L , for some m in N , $Q'' \cong \|d^k\| + m - 1$. Clearly,

$$Q'(\|d^k\| + r|d^k| \pm 1) = \frac{1 + r \frac{|d^k|}{\|d^k\|} \pm \frac{1}{\|d^k\|}}{i + \frac{i(m-1)}{\|d^k\|}}$$

Therefore, since $|d^k|/\|d^k\| \rightarrow i-1$ and $\|d^k\| \rightarrow \infty$, it follows

that $\lim_{k \in \mathbb{N}} Q'(\|d^k\| + r \|d^{k-1}\|) = \lim_{k \in \mathbb{N}} Q'(\|d^k\| + r \|d^{k+1}\|) = (1+r(i-1))/i$. Therefore, since $\|d^k\| + r \|d^{k-1}\| \stackrel{e}{\leq} I(k) \stackrel{e}{=} \|d^k\| + [r \|d^{k-1}\|] \stackrel{e}{=} \|d^k\| + r \|d^{k+1}\|$, $\lim_{k \in \mathbb{N}} Q' I(k) = \lim_{k \in \mathbb{N}} Q'(\|d^k\| + [r \|d^{k-1}\|]) = (1+r(i-1))/i$.

The first performance theorem, theorem 1, states that a scheduling function f exhibits best case performance over a left test sequence Q if and only if the sequential spectrum of f over Q is eventually less than every intermediate spectral sequence of Q .

Theorem 1. Let f be in F and Q be in T_L . Then f e.b.c.p. over Q iff $\|f(Q)\| \stackrel{e}{\leq} I$ for each I in $S_I(Q)$.

Proof. Suppose that $Q \stackrel{e}{=} \|d^k\| + m - 1$ for some m in \mathbb{N} .

\Rightarrow Assume that f e.b.c.p. over Q ; that is, $pf(Q) = 1/i$.

The proof is by contradiction; suppose it is not the case that

$\|f(Q)\| \stackrel{e}{\leq} I$ for each I in $S_I(Q)$. Then there is an I in $S_I(Q)$

so that $\|f(Q)\| \not\stackrel{e}{\leq} I$; that is, there does not exist an n in \mathbb{N}

so that $\|f(d^k)\| < I(k)$ if $k > n$. Therefore, there is a denumerable subset of \mathbb{N} , say N' , so that $\|f(d^k)\| \geq I(k)$ for each

k in N' . Say that $I(k) \stackrel{e}{=} \|d^k\| + [r \|d^{k-1}\|]$ for some r in $(0,1)$.

So, from lemma 4.8, $1/i = pf(Q) = \lim_{k \in \mathbb{N}} Q' \|f(d^k)\| = \lim_{k \in N'} Q' \|f(d^k)\|$

$= \lim_{k \in N'} Q' I(k) = \lim_{k \in \mathbb{N}} Q' I(k) = (1+r(i-1))/i$. This is a contra-

dition since $r \neq 0$ and $i > 1$.

\Leftarrow Assume now that $\|f(Q)\| \stackrel{e}{\leq} I$ for each I in $S_I(Q)$. By definition, to show that f e.b.c.p. over Q , it must be shown that the following limit holds: $\lim_{k \in \mathbb{N}} Q' \|f(d^k)\| = 1/i$.

Given $\epsilon > 0$. Since $(1+r(i-1))/i \rightarrow 1/i$ as $r \rightarrow 0$, pick s in $(0,1)$ small enough so that if r is in $(0,s)$, then $(1+r(i-1))/i < 1/i + \epsilon/2$.

Let r be in $(0,s)$. Pick I in $S_I(Q)$ so that $I(k) \stackrel{\epsilon}{\leq} \|d^k\| + [r|d^k|]$. Also, from the above hypothesis, $\|f(Q)\| \stackrel{\epsilon}{\leq} I$. Thus, there is a q in N so that if $k > q$, then $\|f(d^k)\| < I(k) = \|d^k\| + [r|d^k|]$.

Since $1/\|d^k\| \rightarrow 0$ as $k \rightarrow \infty$, pick p in N so that if $k > p$, then $1/\|d^k\| < \epsilon/2$.

Let $k > \max p, q$, then $1/i = Q' \|f(d^k)\| < Q' I(k) =$

$$\frac{\|d^k\| + [r|d^k|]}{i(\|d^k\| + m - 1)} = \frac{\|d^k\| + r|d^k| + 1}{i(\|d^k\| + m - 1)} = \frac{1+r\frac{|d^k|}{\|d^k\|} + \frac{1}{\|d^k\|}}{i(1 + \frac{m-1}{\|d^k\|})} = \frac{1+r(i-1) + \frac{1}{\|d^k\|}}{i(1 + \frac{m-1}{\|d^k\|})} =$$

$$(1+r(i-1))/i + 1/\|d^k\| < 1/i + \epsilon.$$

The next result states that a scheduling function exhibits best case performance over a left test sequence whenever the sequential spectrum of the scheduling function is a left spectral sequence.

Corollary 4.9. Let f be in F and Q in T_L . If $\|f(Q)\|$ is in $S_L(Q)$, then f e.b.c.p. over Q .

Proof. From proposition 3.22, $\|f(Q)\| \stackrel{\epsilon}{\leq} I$ for each I in $S_I(Q)$. Therefore, from theorem 1, f e.b.c.p. over Q .

The following lemma gives the performance of a scheduling function f over an intermediate test sequence if the

sequential spectrum of f is an intermediate spectral sequence. As in lemma 4.8, if the sequential spectrum of f , in this case, is known, then the performance of f is easy to calculate.

Lemma 4.10. Let Q be in T_I and J in $S_I(Q)$, say, $Q \equiv \|d^k\| + [s|d^k|]$ and $J(k) \equiv \|d^k\| + [r|d^k|]$ for r and s in $(0,1)$. Then $\lim Q'J(k) = \lim (\|d^k\| + [r|d^k|]) / i(\|d^k\| + [s|d^k|]) = (1+r(i-1))/i(1+s(i-1))$.

Proof. Clearly,

$$\frac{\|d^k\| + r|d^k| \pm 1}{i(\|d^k\| + s|d^k| \mp 1)} = \frac{1+r \frac{|d^k|}{\|d^k\|} \pm \frac{1}{\|d^k\|}}{i(1+s \frac{|d^k|}{\|d^k\|} \mp \frac{1}{\|d^k\|})}$$

Since $|d^k|/\|d^k\| \rightarrow i-1$ and $\|d^k\| \rightarrow \infty$,

$$\lim_{k \in \mathbb{N}} \frac{\|d^k\| + r|d^k| + 1}{i(\|d^k\| + s|d^k| - 1)} = \lim_{k \in \mathbb{N}} \frac{\|d^k\| + r|d^k| - 1}{i(\|d^k\| + s|d^k| + 1)} = \frac{1+r(i-1)}{i(1+s(i-1))}.$$

$$\text{Since } \frac{\|d^k\| + r|d^k| + 1}{i(\|d^k\| + s|d^k| - 1)} \geq \frac{\|d^k\| + [r|d^k|]}{i(\|d^k\| + s|d^k|)} \equiv Q'J(k) \geq$$

$$\frac{\|d^k\| + r|d^k| - 1}{i(\|d^k\| + s|d^k| + 1)}, \quad \lim_{k \in \mathbb{N}} Q'J(k) = \lim_{k \in \mathbb{N}} \frac{\|d^k\| + [r|d^k|]}{i(\|d^k\| + [s|d^k|])} =$$

$$(1+r(i-1))/i(1+s(i-1)).$$

The next performance theorem, theorem 2, states that a scheduling function f exhibits best case performance over an intermediate test sequence Q if and only if the sequential

spectrum of f over Q is eventually less than each intermediate spectral sequence over Q that is eventually greater than the sequential spectrum of OPT over Q .

Theorem 2. Let f be in F and Q in T_I . Then f e.b.c.p. over Q iff $\|f(Q)\| \leq J$ for each J in $S_I(Q)$ such that $J > \|OPT(Q)\|$.

Proof. Suppose that $Q'' \stackrel{e}{=} \|d^k\| + [s|d^k|]$ for some s in $(0,1)$.

\Rightarrow Assume that f e.b.c.p. over Q , i.e., $pf(Q) = 1/i$.

The proof is by contradiction; suppose there is a J in $S_I(Q)$ such that $J > Q''$ and $\|f(Q)\| \not\leq J$. Therefore, there is a denumerable subset of N , say N' , such that $\|f(d^k)\| \geq J$ for each k in N' . Say that $J(k) \stackrel{e}{=} \|d^k\| + [r|d^k|]$ where r is in $(s,1)$. So, from the previous lemma, $1/i = pf(Q) = \lim_{k \in N} Q' \|f(d^k)\| = \lim_{k \in N'} Q' \|f(d^k)\|$.

$$Q' \|f(d^k)\| = \lim_{k \in N'} Q' J(k) = \lim_{k \in N'} Q' J(k) = (1+r(i-1))/i(1+s(i-1)).$$

This is a contradiction since $0 < s < r < 1$ and $i > 1$.

\Leftarrow Assume that $\|f(Q)\| \leq J$ for each J in $S_I(Q)$ such that $J > Q'$. By definition, to show that f e.b.c.p. over Q , it must be shown that $\lim_{k \in N} Q' \|f(d^k)\| = 1/i$.

Given $e > 0$. Since $\lim_{r \rightarrow s} (1+r(i-1))/i(1+s(i-1)) = 1/i$,

pick u in $(s,1)$ small enough so that if r is in (s,u) , then $(1+r(i-1))/i(1+s(i-1)) < 1/i + e/3$.

Let r be in (s,u) . Pick I in $S_I(Q)$ so that $I(k) \stackrel{e}{=} \|d^k\| + [r|d^k|]$. Since $\|f(Q)\| \leq I$, there is a q in N so that if $k > q$ then $\|f(d^k)\| < I(k) = \|d^k\| + [r|d^k|]$.

Since $|d^k| \rightarrow \infty$, pick m large enough so that if $k > m$, then $s|d^k| > 1$. So if $k > m$, $s|d^k|/\|d^k\| - 1/\|d^k\| > 0$; that is,

$$i\left(1 + s\frac{|d^k|}{\|d^k\|} - \frac{1}{\|d^k\|}\right) > i$$

Since $\|d^k\| \rightarrow \infty$, pick p large enough so that if $k > p$, then $1/\|d^k\| < e/3$.

Choose l large enough so that if $k > l$, then

$$\frac{1+r(i-1)}{i\left(1+s\frac{|d^k|}{\|d^k\|} - \frac{1}{\|d^k\|}\right)} - \frac{1+r(i-1)}{i(1+s(i-1))} < \frac{e}{3}$$

Let $k > \max\{l, m, p, q\}$, then $1/i \leq Q'\|f(d^k)\| < Q'I(k) =$

$$\begin{aligned} \frac{\|d^k\| + [r|d^k|]}{i(\|d^k\| + [s|d^k|])} &\leq \frac{\|d^k\| + r|d^k| + 1}{i(\|d^k\| + s|d^k| - 1)} = \frac{1+r\frac{|d^k|}{\|d^k\|} + \frac{1}{\|d^k\|}}{i\left(1+s\frac{|d^k|}{\|d^k\|} - \frac{1}{\|d^k\|}\right)} \leq \\ \frac{1+r(i-1) + \frac{1}{\|d^k\|}}{i\left(1+s\frac{|d^k|}{\|d^k\|} - \frac{1}{\|d^k\|}\right)} &\leq \frac{1+r(i-1)}{i\left(1+s\frac{|d^k|}{\|d^k\|} - \frac{1}{\|d^k\|}\right)} + \frac{1}{i\|d^k\|} < \frac{e}{3} + \frac{1+r(i-1)}{i(1+s(i-1))} \\ + 1/\|d^k\| &< e/3 + 1/i + e/3 + e/3 = 1/i + e. \end{aligned}$$

Corollary 4.11. Let f be in F and Q in T_I . If $\|f(Q)\| \stackrel{e}{\leq} Q'' + m$ for some m in N , then f e.b.c.p. over Q .

Proof. Suppose J is in $S_I(Q)$ and $Q'' \stackrel{e}{\leq} J$. Then, from proposition 3.21, $Q'' + m \stackrel{e}{\leq} J$. Therefore, $\|f(Q)\| \stackrel{e}{\leq} J$. Thus, from theorem 2, f e.b.c.p. over Q .

The next performance theorem, theorem 3, states that every scheduling function exhibits best case performance over any right test sequence.

Theorem 3. Let f be in F and Q in T_R . Then f e.b.c.p. over Q .

Proof. Suppose that $Q^n \stackrel{e}{=} \llbracket d^k \rrbracket_{-m+1}$ for some m in N . Let $\epsilon > 0$ be given. For each m in N , where $m' = (1-m)/\llbracket d^k \rrbracket$, $1/(1+m') = \llbracket d^k \rrbracket / (\llbracket d^k \rrbracket_{-m+1}) = 1$. Furthermore, $1/(1+m') \rightarrow 1$ since $\llbracket d^k \rrbracket \rightarrow \infty$.

Pick n large enough so that if $k > n$, then $1/(1+m') < 1+\epsilon$ and $Q^n = \llbracket d^k \rrbracket_{-m+1}$.

For each k in N , $\|f(d^k)\| \leq \llbracket d^k \rrbracket$. Let $k > n$, then $1/i \leq Q^i \|f(d^k)\| \leq \llbracket d^k \rrbracket / i (\llbracket d^k \rrbracket_{-m+1}) = 1/i(1+m') < (1+\epsilon)/i = 1/i + \epsilon$.

In the next three lemmas, i is in N and $i > 1$. These lemmas are technical results used in the proofs of the following performance theorems.

Lemma 4.12. $1/i < (1+r(i-1))/i$ iff $0 < r$ iff $1/(1+r(i-1)) < 1$.

Proof. $1/i < (1+r(i-1))/i$ iff $1 < 1+r(i-1)$ iff $0 < r(i-1)$ iff $0 < r$ iff $1 < 1+r(i-1)$ iff $1/(1+r(i-1)) < 1$.

Lemma 4.13. $1/i < 1/(1+r(i-1))$ iff $r < 1$ iff $(1+r(i-1))/i < 1$.

Proof. $1/i < 1/(1+r(i-1))$ iff $1+r(i-1) < i$ iff $r(i-1) < i-1$ iff $r < 1$ iff $1+r(i-1) < i$ iff $(1+r(i-1))/i < 1$.

Lemma 4.14. $t < r$ iff $1/i < (1+r(i-1))/i(1+t(i-1))$.

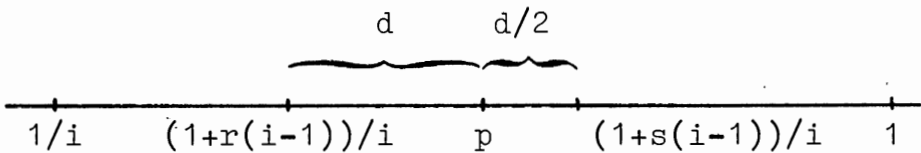
Proof. $t < r$ iff $1+t(i-1) < 1+r(i-1)$ iff $(1+t(i-1))/i < (1+r(i-1))/i$ iff $1/i < (1+r(i-1))/i(1+t(i-1))$.

Notation. Let Q be in T and r in $(0,1)$. Let I_r denote the r^{th} left intermediate spectral sequence of Q , i.e., $I_r(k) \stackrel{e}{=} \|d^k\| + [r \|d^k\|]$.

The next performance theorem, theorem 4, describes the conditions under which a scheduling function exhibits intermediate case performance over a left test sequenc.

Theorem 4. Let f be in F and Q in T_L . Then f e.i.c.p. over Q iff there is a p in $(1/i, 1)$ so that $I_r \stackrel{e}{\prec} \|f(Q)\| \stackrel{e}{\prec} J_s$ for each I_r and J_s in $S_I(Q)$ such that $* 1/i < (1+r(i-1))/i < p < (1+s(i-1))/i < 1$.

Proof. \Rightarrow Assume that f e.i.c.p. over Q , say that $pf(Q) = p$ which is in $(1/i, 1)$. Let I_r and J_s be in $S_I(Q)$ so that $*$ above holds. let $d = \min\{(1+s(i-1))/i - p, p - (1+r(i-1))/i\}$.



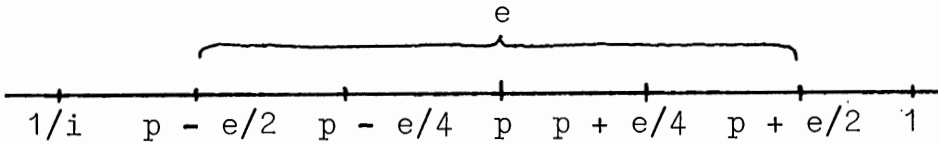
Since $\lim_{k \leftarrow \mathbb{N}} Q' \|f(d^k)\| = p$, $\lim_{k \leftarrow \mathbb{N}} Q' I_r(k) = (1+r(i-1))/i$,

and $\lim_{k \leftarrow \mathbb{N}} Q' J_s(k) = (1+s(i-1))/i$, pick u large enough so that

if $k > u$, then $|Q' \|f(d^k)\| - p| < d/2$, $|Q' I_r(k) - (1+r(i-1))/i| < d/2$, and $|Q' J_s(k) - (1+s(i-1))/i| < d/2$. Therefore, if $k > u$, then $Q' I_r(k) < Q' \|f(d^k)\| < Q' J_s(k)$. Therefore, if $k > u$, then $I_r(k) < \|f(d^k)\| < J_s(k)$. That is, $I_r \stackrel{e}{\prec} \|f(Q)\| \stackrel{e}{\prec} J_s$.

← Assume now that there is a p in $(1/i, 1)$ so that $I_r \stackrel{e}{\leq} \|f(Q)\| \stackrel{e}{\leq} J_s$ for each I_r and J_s in $S_I(Q)$ so that * above holds. By definition, it must be shown that $\text{pf}(Q) = \lim Q'(\|f(d^k)\|)$ is in $(1/i, 1)$.

Given $e > 0$. Without loss of generality, say that $1/i < p - e/2$ and $p + e/2 < 1$. Pick $r = (i(p-e/4)-1)/(i-1)$ and $s = (i(p+e/4)-1)/(i-1)$. Thus $(1+r(i-1))/i = p - e/4$ and $(1+s(i-1))/i = p + e/4$. So r and s are in $(0, 1)$ from lemmas 4.12 and 4.13 since $p - e/4$ and $p + e/4$ are in $(1/i, 1)$.



By lemma 4.8, $Q'I_r(k) \rightarrow (1+r(i-1))/i$ and $Q'J_s(k) \rightarrow (1+s(i-1))/i$. Pick u in N large enough so that if $k > u$, $|Q'I_r(k) - (1+r(i-1))/i| < e/4$, $|Q'J_s(k) - (1+s(i-1))/i| < e/4$, and $I_r(k) < \|f(d^k)\| < J_s(k)$. Let $k > u$. Then $Q'I_r(k) < p < Q'J_s(k)$. So, $|Q'\|f(d^k)\| - p| < |Q'I_r(k) - Q'J_s(k)| \leq |Q'I_r(k) - (p - e/4)| + |(p + e/4) - Q'J_s(k)| + |(p - e/4) - (p + e/4)| < e/4 + e/4 + 2e/4 = e$.

The next lemma states that a scheduling function exhibits intermediate case performance over an intermediate test sequence whenever the sequential spectrum of the scheduling function is a right spectral sequence.

Lemma 4.15. Let Q be in T_I and $\|f(Q)\|$ in $S_R(Q)$, say $Q' \stackrel{e}{\leq} \|d^k\| + [t|d^k|]$ where t is in $(0, 1)$. Then $\text{pf}(Q) = \lim Q'\|f(d^k)\| = \lim Q'(\|d^k\|^{-m+1}) = 1/(1+t(i-1))$.

Proof. Consider the limit of the following inequalities as k gets large:

$$\frac{\frac{\|d^k\|}{\|d^k\|} + \frac{1-m}{\|d^k\|}}{i(1+t\frac{|d^k|}{\|d^k\|} + \frac{1}{\|d^k\|})} = \frac{\|d^k\| - m + 1}{i(\|d^k\| + t|d^k| + 1)} \leq Q'(\|d^k\| - m + 1) = \frac{\|d^k\| - m + 1}{i(\|d^k\| + t|d^k|)}$$

$$\leq \frac{\|d^k\| - m + 1}{i(\|d^k\| + t|d^k| - 1)} = \frac{\frac{\|d^k\|}{\|d^k\|} + \frac{1-m}{\|d^k\|}}{i(1+t\frac{|d^k|}{\|d^k\|} - \frac{1}{\|d^k\|})}$$

The limit of both the first and last term as k gets large, in the above string of equations, is equal to $1/(1+t(i-1))$. So, the result is proved.

The next lemma gives a bound on the performance of a scheduling function over an intermediate test sequence.

Lemma 4.16. Let Q be in T_I , say, $Q'' \equiv \|d^k\| + [t|d^k|]$. Then $\text{pf}(Q)$, if it exist, is in $[1/i, 1/(1+t(i-1))]$.

Proof. For each k in N , $Q'' \leq \|f(d^k)\| \leq \|d^k\|$. Therefore, for each k in N , $1/i = Q''Q' \leq Q'\|f(d^k)\| \leq Q'\|d^k\|$. By lemma 4.15, $Q'\|d^k\| \rightarrow 1/(1+t(i-1))$. Thus, if $\text{pf}(Q)$ exists, then $1/i \leq \text{pf}(Q) = \lim_{k \in N} Q'\|f(d^k)\| = \lim_{k \in N} Q'\|d^k\| = 1/(1+t(i-1))$.

Notation. In what follows, t' is used to denote the expression $1+t(i-1)$. The same notation is used for the letters r and s .

The following definition and five lemmas are used in the statement and proof of the next performance theorem, theorem 5.

Definition 4.17. Let Q be in T_I , say, $Q \stackrel{e}{=} I_t$. Let $1/i \leq p \leq 1/t'$ and let $S_I^1(Q)$ denote the set J in $S_I(Q): I_t \stackrel{e}{\leq} J$. Then the lower set for Q at p , denoted $L_p(Q)$, is the set $\{I_r \text{ in } S_I^1(Q): r'/t' < p\}$. The upper set for Q at p , denoted $U_p(Q)$, is the set $\{J_s \text{ in } S_I^1(Q): s'/t' > p\}$.

Lemma 4.18. $1/i < p < 1/t'$ iff $L_p(Q) \neq \emptyset \neq U_p(Q)$.

Proof. \Rightarrow Suppose that $1/i < p < 1/t'$. Let $2d = \min\{1/t' - p, p - 1/i\}$, $r = (it'(p-d)-1)/(i-1)$, and $s = (it'(p+d)-1)/(i-1)$. Then $r'/it' = p-d$, $s'/it' = p+d$, and $1/i < p-d < p < p+d < 1/t'$. From lemmas 4.13 and 4.14, r and s are in $(0,1)$. Therefore, I_r is in $L_p(Q)$ and I_s is in $U_p(Q)$.

\Leftarrow Suppose $L_p(Q) \neq \emptyset \neq U_p(Q)$. Then there is an I_r in $L_p(Q)$ and an I_s in $U_p(Q)$ so that $r'/it' < p < s'/it'$. By the definition of p , $1/i \leq p \leq 1/t'$. If $p = 1/i$, then $r'/it' < 1/i$ and $r' < t'$, a contradiction since $0 < t < r < 1$. If $p = 1/t'$, then $1/t' < s'/it'$ and $i < s'$, a contradiction since $0 < s < 1$. Therefore, $1/i < p < 1/t'$.

Lemma 4.19. If $p = 1/t'$, then $L_p(Q) \neq \emptyset$.

Proof. Given $0 < t < 1$. From lemmas 4.12 and 4.13, $1/i < 1/t' < 1$. Let $p = 1/t'$, $2d = \min\{1-p, p - 1/i\}$, and $r = (it'(p-d)-1)/(i-1)$. Then $1/i < r'/it' = p-d < p = 1/t' < 1$. Thus, $r'/i < 1$. From lemma 4.13, $r < 1$. From lemma 4.14, $t < r$. Since $0 < t < r < 1$, I_r is in $L_p(Q)$.

Lemma 4.20. $1/i < p \leq 1/t'$ iff $L_p(Q) \neq \emptyset$.

Proof. This result follows directly from lemmas 4.18 and 4.19.

Lemma 4.21. If $p = 1/i$, then $U_p(Q) \neq \emptyset$.

Proof. Let $p = 1/i$, $2d = 1/t' - p$, and $s = (it'(p+d) - 1)/(i-1)$. Then $1/i = p < (1+s(i-1))/it' = p+d < 1/t' < 1$. Therefore, $s'/i < 1$. From lemma 4.13, $s < 1$. From lemma 4.14, $t < s$. Since $0 < t < s < 1$, I_s is in $U_p(Q)$.

Lemma 4.22. $1/i \leq p < 1/t'$ iff $U_p(Q) \neq \emptyset$.

Proof. This result follows directly from lemmas 4.18 and 4.21.

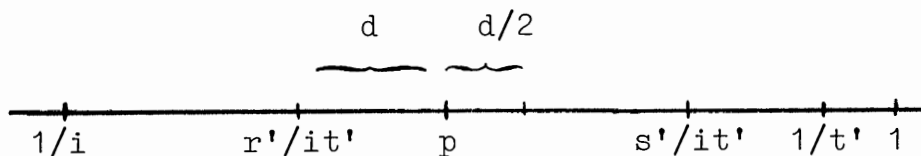
The next result describes the conditions under which a scheduling function exhibits intermediate case performance over an intermediate test sequence.

Theorem 5. Let f be in F and Q in T_I , say, $Q'' \stackrel{e}{=} I_t$. Then f e.i.c.p. over Q iff there is a p in $(1/i, 1/t']$ so that $I_r \stackrel{e}{\leq} \|f(Q)\|$ for each I_r in $L_p(Q)$ and $\|f(Q)\| \stackrel{e}{\leq} J_s$ for each J_s in $U_p(Q)$.

Proof. \Rightarrow Assume that f e.i.c.p. over Q , say, $pf(Q) = p$. By lemma 4.16, $1/i < p \leq 1/t'$.

Case 1. Suppose $1/i < p < 1/t'$. From lemma 4.18, $L_p(Q) \neq \emptyset \neq U_p(Q)$. Let I_r be in $L_p(Q)$ and J_s in $U_p(Q)$. Then $r'/it' < p < s'/it'$ and $0 < t < r < s < 1$. From lemma 4.12, $1/t' < 1$. From lemma 4.13, $s'/it' < 1/t'$. From lemma 4.14, $1/i < r'/it'$. Thus, $1/i < r'/it' < p < s'/it' < 1/t' < 1$. Let $d = \min\{p - (1 +$

$r(i-1))/it'$, $(1+s(i-1))/it' - p$. For example, say,

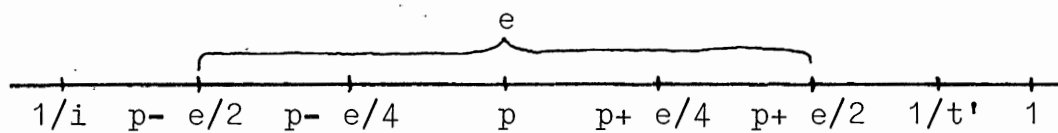


Since $\lim Q' \|f(d^k)\| = p$, $\lim Q' I_r(k) = r'/it'$, and $\lim Q' J_s(k) = s'/it'$, pick u large enough so that if $k > u$, then $|Q' \|f(d^k)\| - p| < d/2$, $|Q' I_r(k) - r'/it'| < d/2$, and $|Q' J_s(k) - s'/it'| < d/2$. Thus, if $k > u$, then $Q' I_r(k) < Q' \|f(d^k)\| < Q' J_s(k)$. Therefore, if $k > u$, then $I_r(k) < \|f(d^k)\| < J_s(k)$. That is, $I_r \stackrel{e}{<} \|f(Q)\| \stackrel{e}{<} J_s$.

Case 2. Suppose that $p = 1/t'$. Then $U_p(Q) = \emptyset$. Let I_r be in $L_p(Q)$. Therefore, $1/i < r'/it' < p = 1/t' < 1$. Let $d = p - r'/it'$. As above, pick u large enough so that if $k > u$, then $|Q' \|f(d^k)\| - p| < d/2$ and $|Q' I_r(k) - r'/it'| < d/2$. Thus if $k > u$, then $Q' I_r(k) < \|f(d^k)\| Q'$. Therefore, $I_r \stackrel{e}{<} \|f(Q)\|$.

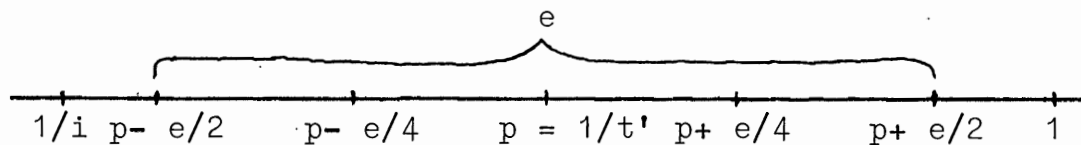
\Leftarrow Suppose there is a p in $(1/i, 1/t']$ so that $I_r \stackrel{e}{<} \|f(Q)\|$ for each I_r in $L_p(Q)$ and $\|f(Q)\| \stackrel{e}{<} J_s$ for each J_s in $U_p(Q)$. By definition, it must be shown that $pf(Q) = \lim Q' (\|f(d^k)\|)$ is in $(1/i, 1)$.

Case 1. Suppose that p is in $(1/i, 1/t')$. Given $e > 0$. Without loss of generality, assume that $1/i < p - e/2$ and $p + e/2 < 1/t'$. Pick $r = (it'(p - e/4) - 1)/(i - 1)$ and $s = (it'(p + e/4) - 1)/(i - 1)$. Then $p - e/4 = r'/it'$ and $p + e/4 = s'/it'$. By lemma 4.12, $0 < r < s < 1$ since $p - e/4$ and $p + e/4$ are in $(1/i, 1)$. For example,



Therefore, I_r is in $L_p(Q)$ and J_s is in $U_p(Q)$. From lemma 4.10, $\lim Q'I_r(k) = r'/it'$ and $\lim Q'J_s(k) = s'/it'$. Since $I_r(k)$ is in $L_p(Q)$ and $J_s(k)$ is in $U_p(Q)$, $I_r(k) \stackrel{e}{\leq} \|f(Q)\| \stackrel{e}{\leq} J_s(k)$. Pick u in N large enough so that if $k > u$, then $|Q'I_r(k) - r'/it'| < e/4$, $|Q'J_s(k) - s'/it'| < e/4$, and $I_r(k) < \|f(d^k)\| < J_s(k)$. Let $k > u$, then $Q'I_r(k) < p < Q'J_s(k)$ and $Q'I_r(k) < Q'\|f(d^k)\| < Q'J_s(k)$. So, if $k > u$, then $|Q'\|f(d^k)\| - p| < |Q'I_r(k) - Q'J_s(k)| = |Q'I_r(k) - (p- e/4)| + |(p+ e/4) - Q'J_s(k)| + |(p- e/4) - (p+ e/4)| < e/4 + e/4 + 2e/4 = e$.

Case 2. Given $e > 0$. Without loss of generality, assume that $1/i < p- e/2$ and $p+ e/2 < 1$. Suppose that $p = 1/t'$. Pick $r = (it'(p- e/4)-1)/(i-1)$. So $r'/it' = p- e/4$. By lemmas 4.13 and 4.14, $0 < r < 1$ since $p- e/4$ is in $(1/i, 1/t')$. for example,



Therefore, $I_r(k) \stackrel{e}{\leq} \|d^k\| + [r|d^k|]$ is in $L_p(Q)$. Since $\lim Q'I_r(k) = r'/it'$, $\lim Q'\|d^k\| = 1/t' = p$, and $I_r(k)$ is in $L_p(Q)$, pick u in N large enough so that $|Q'I_r(k) - r'/it'| < e/4$, $|Q'\|d^k\| - p| < e/4$, and $I_r(k) < \|f(d^k)\|$. Since $\|f(d^k)\| \leq \|d^k\|$ for each k in N , then $Q'\|f(d^k)\| \leq Q'\|d^k\|$ for each k in N . Thus, if $k > u$, $I_r(k)Q' < Q'\|f(d^k)\| \leq Q'\|d^k\|$. Let $k > u$, then $|Q'\|f(d^k)\| - p| \leq |Q'\|f(d^k)\| - Q'\|d^k\| + |Q'\|d^k\| - p| \leq |Q'I_r(k) - Q'\|d^k\|| + e/4 \leq |Q'I_r(k) - (p- e/4)| + |(p- e/4) - Q'\|d^k\|| + e/4 \leq e/4$

$$+ |p - Q' \|d^k\| | + |-e/4| + e/4 < e/4 + e/4 + e/4 + e/4 = e.$$

The next corollary states that information concerning the sequential spectrum of a scheduling function over an intermediate test sequence can be obtained by knowing that the function exhibits intermediate case performance.

Corollary 4.23. Let f be in F and Q in T_I . If f e.i.c.p. over Q , then $Q' + m \stackrel{e}{\leq} \|f(Q)\|$ for each m in N .

Proof. Suppose f e.i.c.p. over Q . By theorem 5, there is a p in $(1/i, 1/t']$ so that $I_r \stackrel{e}{\leq} \|f(Q)\|$ for each I_r in $L_p(Q)$. By lemma 4.20, $L_p(Q) \neq \emptyset$. So, there is an I_r in $L_p(Q)$ so that $Q' \stackrel{e}{\leq} I_r \stackrel{e}{\leq} \|f(Q)\|$. By proposition 3.21, $Q' + m \stackrel{e}{\leq} I_r$ for each m in N .

The last performance theorem describes the conditions under which a scheduling function exhibits worst case performance over a left test sequence.

Theorem 6. Let f be in F and Q in T_L . Then f e.w.c.p. over Q iff $\|f(Q)\| \stackrel{e}{\geq} I$ for each I in $S_I(Q)$.

Proof. \Rightarrow Assume that f e.w.c.p. over Q , i.e., $pf(Q) = 1$. The proof is by contradiction; suppose that $\|f(Q)\| \not\stackrel{e}{\geq} I_r$ is in $S_I(Q)$ for some r in $(0, 1)$. Then there is a denumerable subset of N , say, N' such that $\|f(d^k)\| \leq I_r(k)$ for each k in N' . From lemmas 4.8 and 4.13, $Q' I_r(k) \rightarrow r'/i < 1$. Therefore,

$$1 = pf(Q) = \lim_{k \in N} Q' \|f(d^k)\| = \lim_{k \in N'} Q' \|f(d^k)\| \leq \lim_{k \in N'} Q' I_r(k) =$$

$\lim_{k \in N} Q' \cdot I_r(k) = r'/i < 1$, a contradiction.

\Leftarrow Assume that $\|f(Q)\| \stackrel{e}{>} I$ for each I in $S_I(Q)$. By definition, it must be shown that $\text{pf}(Q) = \lim Q' \|f(d^k)\| = 1$.

Given $e > 0$. Without loss of generality, assume that $e < 2(1 - 1/i)$. So $1/i < 1 - e/2 < 1$. Pick $r = (i(1 - e/2) - 1)/(i - 1)$. Then $r' = 1 - e/2$. From lemmas 4.12 and 4.13, $1/i < r < 1$. So, I_r is in $S_I(Q)$ and $I_r \stackrel{e}{<} \|f(Q)\|$. From lemma 4.8, $Q'I_r(k) \rightarrow r'/i$. Pick u in N large enough so that if $k > u$, then $|Q'I_r(k) - r'/i| < e/2$ and $I_r(k) < \|f(d^k)\|$. Let $k > u$, then $Q'I_r(k) < Q'\|f(d^k)\| \leq 1$ and $|Q'\|f(d^k)\| - 1| \leq |Q'I_r(k) - 1| \leq |Q'I_r(k) - r'/i| + |r'/i - 1| < e/2 + |1 - e/2 - 1| = e/2 + e/2 = e$.

Corollary 4.24. Let f be in F and Q in T_L . If $\|f(Q)\|$ is in $S_R(Q)$, then f e.w.c.p. over Q .

Proof. From proposition 3.22, $\|f(Q)\| \stackrel{e}{>} I$ for each I in $S_I(Q)$. Therefore, from theorem 6, f e.w.c.p. over Q .

CHAPTER V

INDEPENDENT TASK SCHEDULING PERFORMANCE THEOREMS

In this chapter, the m -process independent task scheduling model ITS, as given in [2], is redefined so that the performance theorems as given in chapter four can be applied to the redefined but equivalent model.

DESCRIPTIONS, SCHEDULES, AND SCHEDULING FUNCTIONS

In the following definitions, let T denote the task set $\{T_1, T_2, T_3, \dots\}$. T is essentially the resource set R , given earlier, relabeled. Although the relabeling is a minor point, it is mentioned since the notion of task, not resource, is central to the independent task scheduling model.

Definition 5.1. Let D denote the set $\bigcup_{k=2}^{\infty} N^k$. D is the description set and elements of D are descriptions. Let d be in D , say, $d = (d_1, d_2, \dots, d_n)$. Then $\|d\| = \max\{d_i : i \text{ in } N_n\}$ and $\|d\|_1 = d_1 + d_2 + \dots + d_n$. Think of d_i as the time needed by any processor to complete or process task T_i . So if $d = (3, 1, 345, 1, 4)$, then d is process time description of the task set $\{T_1, T_2, T_3, T_4, T_5\}$ where 1 is the process time for task T_2 and T_4 , 345 is the process time for task T_3 , and 3 and 4 are respectively the process times for T_1 and T_5 . Thus when referring to a task set, refer to its description.

Definition 5.2. A sequence $p(t)$ in $P(T)$ is called an s-process if the following conditions hold:

- (1) $p(t) = \emptyset$ or $p(t)$ is a singleton for each t in N
- (2) There is an s in N so that $p(s) \neq \emptyset$ and $p(t) = \emptyset$ if $t > s$
- (3) If $p(t) = p(t+k) \neq \emptyset$ for some k in N , then $p(t) = p(i)$ for each i in $\{t+1, t+2, \dots, t+k\}$

Definition 5.3. An m -tuple of s -processes, say, $(s_1, s_2, s_3, \dots, s_m)$ is a schedule if the following conditions hold:

- (1) $m \geq 2$
- (2) For $(j, t) \in N_m \times N$, if $s_j(t) \neq \emptyset$, then $s_j(t) \neq s_i(u)$ for each (i, u) in $(N_m - \{j\}) \times N$
- (3) If there is a v in N so that $s_i(v) = \emptyset$ for each i in N_m , then $s_i(t) = \emptyset$ for each $t > v$ and i in N_m

Let S denote the set of all schedules. Let S_m denote the set $\{s \text{ in } S; w(s) = m\}$.

Definition 5.4. Let F_m denote the set of all functions from D into S_m such that if f is in F_m , d is in D , s is in S_m , $f(d) = s$, $d = (d_1, d_2, \dots, d_n)$, and $s = (s_1, s_2, \dots, s_m)$, then the following conditions hold:

- (1) i is in N_n iff there is a (j, t) in $N_m \times N$ such that $T_i = s_j(t)$
- (2) If for some (j, t) in $N_m \times N$, $T_i = s_j(t)$, then $\sum_{t=1}^{\infty} \overline{\{T_i\}} \cap s_j(t) = d_i$.

(1) guarantees that only the tasks to be scheduled are scheduled. (2) guarantees that a task to be scheduled is scheduled

just long enough to be processes.

Definition 5.5. The m-process, independent task scheduling system T_m is the 4-tuple (T, F_m, D, S_m) where T is the set of tasks, F_m is the set of scheduling functions, D is the set of descriptions, and S_m is the set of schedules.

Example 5.6. Let $\{T_i : i \text{ in } N_5\}$ be a set of tasks to be scheduled on 3 identical processors where $d = (1, 3, 4, 1, 2)$ is the process time description. Then the system T_3 is used. Let f be in F_3 . Then $f(d)$ may have any of the following schedules:

<u>Time</u>	<u>P₁</u>	<u>P₂</u>	<u>P₃</u>	<u>P₁</u>	<u>P₂</u>	<u>P₃</u>	<u>P₁</u>	<u>P₂</u>	<u>P₃</u>
1	T ₁	T ₂	T ₃	T ₁	T ₂	T ₅	T ₂	T ₃	T ₄
2	T ₅	T ₂	T ₃	T ₃	T ₂	T ₅	T ₂	T ₃	
3	T ₅	T ₂	T ₃	T ₃	T ₂		T ₂	T ₃	
4		T ₄	T ₃	T ₃	T ₄		T ₅	T ₃	
5				T ₃			T ₅	T ₁	

The first schedule above has length 4. The second and third schedules both have schedule lengths of 5. The first is an example of an optimal schedule.

Notice that the definition of ITS given in 5.5 above is equivalent to the definition of ITS given in [2]. Also at this point the definitions in chapter 3 can now be applied to ITS. Thus the performance theorems given in chapter 4 hold for ITS.

CHAPTER VI

SOME SPECULATION

Though the basic work reported in this dissertation is theoretical in nature, it was felt appropriate to offer some speculative extensions based on the insights gained during the development of the theory.

The extension being suggested is toward a general resource flow network model (RFN). The motivation is to use the RFN model to analyze large complex social systems, such as industrial, corporate, governmental and combinations of these systems, in search of new insights into their structure and behavior.

To begin, two heuristic equations derivable from the mathematics of this dissertation are developed;

I. One involves the relation between resource availability and the degree of resource conflict experienced in constructing schedules. Roughly speaking, the level of resource conflict is inversely proportional to the availability of resources. This may be represented by the expression $C = kR^{-1}$ where C represents the resource conflict level, R represents the availability of resources, and k is a constant of proportion dependent on the particular system under investigation.

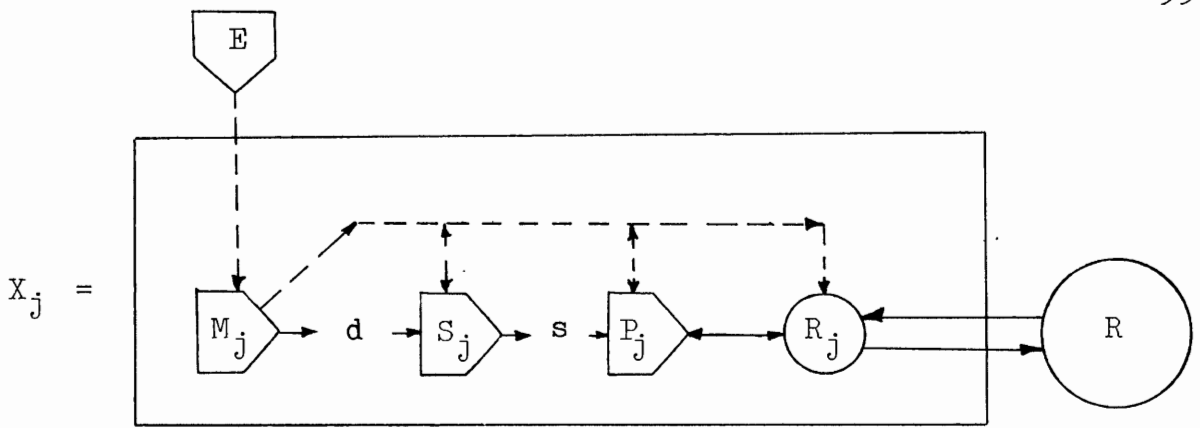
This expression says that for a specified description, as the quantity of "needed" resources is increased, the level of scheduling conflict, in general, decreases. Likewise, as the resource availability level decreases, the conflict level increases.

II. The second equation states that the length of the schedules produced is proportional to the level of resource conflict. This is represented by the expression $L = hC + m$ where L represents the length of the schedule, h the conflict, and m represents the length of the longest process description. This expression says that as conflict increases, so does the schedule length. But the schedule length can never get smaller than the longest process description in the mix of processes being scheduled.

From these two equations, the equation $L = kR^{-1} + m$ is derivable. It says that schedule length is inversely proportional to the availability of resources.

The next step in this speculative exercise is to develop the RFN model in outline form. The explicit mathematical development is left as a future research project.

A system X_j is an s-type system if it can be conceptualized as given in the following diagram:



S-system Description. The s-system X_j is made of the constructs inside the large rectangle given above. M_j is the management or control mechanism for X_j . It decides what the system does by providing the scheduling device S_j with instructions or descriptions, denoted d , of the goals it wants to attain. Using the project descriptions and knowledge about the resources in the resource or inventory set R_j , the scheduling device S_j produces resource schedules, denoted s , which are processed by the production or processing device P_j which in turn produces new resources that are sent to the resource set along with unused resources for possible distribution to a central resource set R . The resource set R_j sends, trades or otherwise dispenses with X_j 's resources through R and under control of M_j . X_j also receives new resources via R .

The dotted lines indicate communications flow. For example, the dotted line from M_j to R_j indicates that M_j can communicate orders or instructions to R_j but since the arrow is one way, R_j does not communicate orders to M_j . Notice that M_j also can give orders to S_j or P_j . The dotted line from E

to M_j indicates that there may be an outside entity that gives orders or sets policy for M_j to follow. The solid lines indicate resource flow.

Intuitively, M_j does high level planning and has overall control over X_j (although E , if it exists for X_j , has control over M_j), S_j sees to the detail of carrying out plans and scheduling resources, P_j uses the resources and does the processing, and R_j takes care of the inventory.

A process here, intuitively, is a finite step-by-step procedure for doing something. Resources are used up in the execution of each step. The things that an s-system does are accomplished via the processes. A set of processes to be executed by a system during the same processing period is called a description. For instance, if a company decides to build a new product, there are specific steps that must be done that require using company resources, i.e., designing, testing, producing, and marketing the product. Each of these steps describes a process that must be completed and many steps or processes can be done simultaneously since they do not in general require the same resources.

Properties of S-systems. Each s-system has a set of critical processes, things that must be done on a regular basis for the survival of the system. For example, in most companies management and accounting functions must be done regularly.

Changes in the environment may require the system to initiate a new process to respond to these changes. A company

may have to change a product in order to meet new air pollution standards. So a new product design and development process would have to be started. These types of changes require an additional and sometimes unplanned for expenditure of resources.

Thus, if not periodically restored, the system's resource set is eventually depleted by critical and unplanned processes, and the system eventually terminates.

S-systems usually replenish their resource sets by producing resources for sale or trade. The sale or trade of a resource is represented by the exchange of resources with the critical resource set R . For instance, if x is sold or traded for y , then x goes to R and y goes to the systems resource set from R .

Thus for each s-system there may be a continuous flow of resources to and from its resource set and R . If a system is to survive for very long, the resource flow must be such that there are always enough resources for the operation of its critical processes, i.e., survivability is a function of resource availability.

New Systems From Old. It is permissible for an s-system to be a resource of another s-system. This happens when the management of the first system is controlled by the management of the second. For example, company A may own companies B and C both of which build components for A's use. That is, A uses B and C as resources to build the components it needs.

Let W denote the set of all s-systems, so for some finite

indexing set I , $W = \{X_j: j \text{ is in } I\}$. W is called the world system. Let K denote a resource sink. That is, when a resource is destroyed or in some other way deemed totally useless or valueless, it goes into K . The set $R = \{K\} \cup \{R_j: i \text{ is in } I\}$ is called the world resource set.

Let $J = \{i, j, k\} \subseteq I$. The coupling product of X_i and X_j , denoted X_i/X_j , is the set $\{X_i, X_j\}$ together with the property that at least one of the systems has some control of the behavior of the other s-system. This is referred to as the coupling control property. So the coupling product is commutative and associative. $X_i/X_j/X_k$ may also be denoted as $\prod_{i \in I} X_i$. So $W = \prod_{i \in I} X_i$.

Example of an RFN Model. For example, $USA = \prod_{i \in I} X_i$ where X_i is in USA if X_i is an s-system and the coupling properties consist of the laws by which the US government governs. One of the X_i 's here is the US government since the government can be conceptualized as an s-system. In this manner, the world system W can be partitioned into a set of governments (a set of coupling products) that correspond to the governments of the real world. Using this technique, a network of resource and communication flows can be considered. Such a concept, that is, a set of coupling products together with the resource and communications network is an example of a resource flow network (RFN) model.

Conflict Analysis Using RFN Models. Using this model and the heuristic equations given earlier, an investigation into the properties and behavior of and conflict between large

complex social systems such as world governments and their interactions can be made.

For example the analysis of world wars can be simulated using this model. In such a model, the allies of WWII could be represented as one coupling product; the enemy, Germany and Japan, as another and the neutral countries as a third.

The RFN model focuses on resources; it provides a resource-based explanation as to why the Germans lost the war. According to this model, it was basically a question of reflow and scheduling. From the work developed in chapter three, Spectral Theory, it can be seen that for a fixed test sequence and a decreasing resource set, the spectral sequence for each scheduling function shifts to the right in the spectral diagram of the test sequence. This in turn means that the schedule lengths get longer; that is, for a fixed set of processes, as the resource set decreases, the schedule lengths increase.

In light of the spectral shift, consider the efforts the Germans made to hold fronts, expand the war, and build weapons as examples of critical processes being carried out in parallel. As the Germans began losing resources at a rate faster than the replacement rate, the schedule lengths or processing times of the critical processes began getting longer and longer because of the increasing resource conflict generated by the continual destruction of their resource set. The difference of these rates was such that these critical processes could not be maintained. As a result the system did not survive.

One area where research and development of this model is clearly needed is in the mathematical development of the resource and communication flow networks. Once this is developed, a mathematical explanation of the above may be explicitly stated.

Clearly the above speculative extensions of the theoretical results of this dissertation are at an early stage of conception. However, it is hoped that they suggest some possible extension of the scheduling system notions both in theory and in application.

BIBLIOGRAPHY

- [1] Bass, L.J., "On Optimal Processor Scheduling for Multiprogramming", SIAM Journal of Computing, December 1973, pp. 273-280.
- [2] Garey, M.R., Graham, R.L., Johnson, D.S., "Performance Guarantees for Scheduling Algorithms", Operations Research, January-February 1978, pp. 3-21.
- [3] Reingold, E.M., Nievergelt, J., Deo, N., Combinatorial Algorithms, Prentice-Hall, 1977.