STOCHASTIC FLOW SHOP SCHEDULING

by

S. Suresh

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Industrial Engineering and Operations Research

APPROVED:

R. L. Disney, Co-chairman

R. D. Foley, Co-chairman

I. M. Besieris

Y. D. Mittal

H. D. Sherali

January, 1987

Blacksburg, Virginia

STOCHASTIC FLOW SHOP SCHEDULING

by

S. Suresh

Committee Co-Chairman: Ralph L. Disney
Department of Industrial Engineering and Operations Research

Committee Co-Chairman: Robert D. Foley
Industrial Systems Engineering Department

(ABSTRACT)

In this dissertation we present new results for minimizing the makespan in a flow shop with zero storage between machines. Makespan is the total length of time required to process a set of jobs. We consider m machines and n jobs with random processing times. Since there is no intermediate storage between machines, a job that has finished processing at one machine may have to stay on that machine until the next machine is free. Our goal is to select a schedule that will minimize the makespan.

Our results require various stochastic orderings of the processing time distributions. Some orderings minimize the expected makespan, and some stronger results stochastically minimize the meakspan. The optimum sequence of these latter cases not only minimizes the expected makespan but also maximizes the probability of completing a set of jobs by time t for any t.

The last result is concerned with scheduling jobs on two identical parallel machines. The jobs are subjected to some intree precedence constraints. We resolve a conjecture that appeared in Pinedo and Weiss (1985) and give conditions under which the conjecture is true and give

examples to prove that the conjecture is false in general.

ACKNOWLEDGEMENTS

I wish to express my gratitude and appreciation to Professors Yashaswini D. Mittal and Ioannis M. Besieris for serving on my dissertation committee. They have always given me useful and constructive criticism during my graduate career. I would like to thank Professor Hanif D. Sherali for serving on my committee and for also being an excellent teacher of many optimization courses. I would like to thank Professor Ralph L. Disney for agreeing to be the Co-chairman of my committee. Professor Disney was my first teacher in probability. His classes were always stimulating and contributed to my pursuing a career in this field.

I would like to thank my colleagues Professors Peter Nagaroka and Robert Storer for many useful discussions during my one year stay at Georgia Institute of Technology. I would like to thank my supervisor Dr. David Y. Burman and AT&T for permitting me work time to put the finishing touches on my dissertation.

I thank for typing this dissertation. Even though I was not physically present at Blacksburg when this document was typed, made the job easy with her professionalism.

Finally, I would like to express my sincere thanks to my advisor Professor Robert D. Foley. I doubt whether I could have ever finished my Ph.D. degree without his constant guidance and encouragement. He was always there when I needed him. I always felt that I was working 'with' him rather than working 'for' him. The education he provided included not only technical knowledge but also a sense of

professionalism and an appreciation of scholarly work. He was not only my academic advisor but my friend too.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction.

An extensive number of papers have been written on scheduling jobs efficiently. The results in the literature can be divided roughly into two groups: heuristic solutions and exact solutions. One of the first exact solutions is the classic result of Johnson (1954), who determines the optimal sequence to process jobs in a two-machine flow shop assuming that the processing times of each job on each machine are known. This assumption leads to a second way of dividing the literature: deterministic scheduling problems and stochastic scheduling problems. An example of a stochastic scheduling problem would be Johnson's two-machine flow shop in which the processing times are random variables with known distributions. Bagga (1970) determines the optimal schedule in a two-machine flow shop when the processing times are independent exponentially distributed random variables.

A large literature exists in the area of deterministic scheduling, i.e., where the processing times of each job on each machine are known. Baker (1974) and Conway, Maxwell and Miller (1969) provide a background on deterministic scheduling. On the contrary, the literature on stochastic scheduling is small. The work in this thesis falls into the category of exact solutions for stochastic scheduling problems. In particular, most of our work will involve flow shops that consist of m machines. We also have some results for scheduling jobs on two identical parallel machines.

A flow shop consists of n jobs that must be processed on m machines. The m machines are in tandem as shown in Figure 1.1. We assume that all n jobs are available in front of the first machine at time zero. A job is not allowed to pass other jobs; hence, the jobs are processed in the same order on each machine.

An important feature of the flow shop is the amount of storage between machines. In many of our results, we will be concerned with the two extreme cases: flow shops with infinite intermediate storage between machines and flow shops with zero intermediate storage between machines. We define a vector $(k_1,k_2,\ldots,k_{m-1})$ where k_i represents the storage space between machines i and i+1.

When a job finishes processing on machine i in a flow shop with infinite intermediate storage, the job is sent to machine i+1 if machine i+1 is empty; otherwise, the job is sent to the storage area between machine i and machine i+1. When a job finishes processing on machine i in a flow shop with zero intermediate storage between machines, the job is sent to machine i+1 if machine i+1 is empty; otherwise, the job must wait on machine i until the job on machine i+1 leaves. Thus, if there is a job on machine i+1, when the job on machine i finishes, machine i will be unable to process another job until the job on machine i+1 leaves for the next machine. This phenomenon is known as blocking. Hence, in a flow shop with zero intermediate storage, machines are in one of three states: busy, idle, or blocked. In a flow shop with infinite intermediate storage, machines are in one of the two states: busy or idle. Clearly, these two systems behave quite differently.

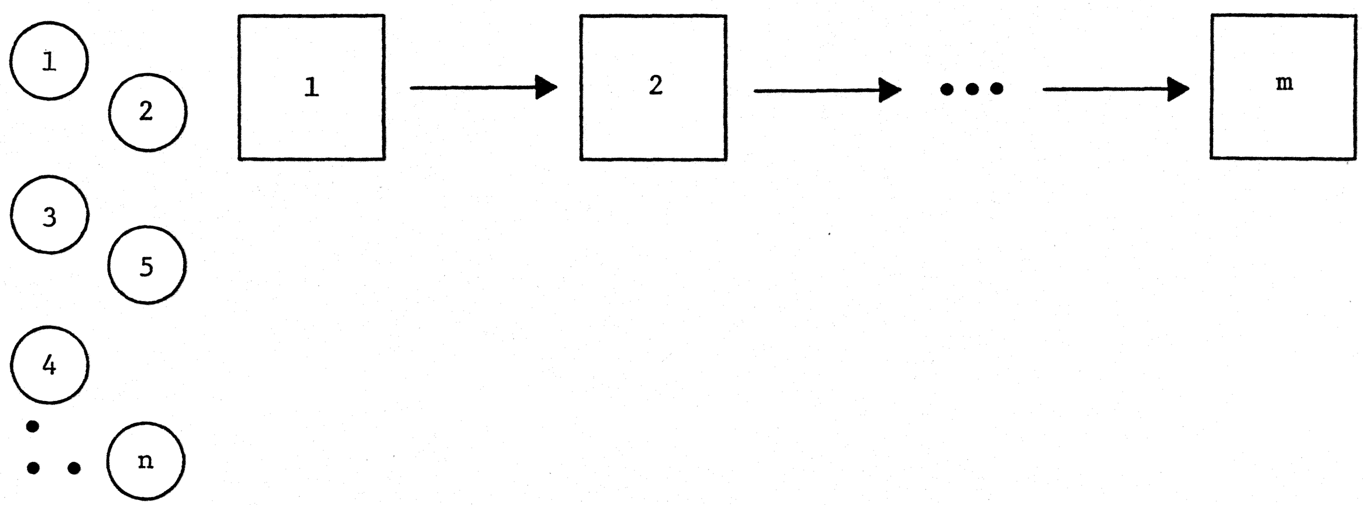

Figure 1.1 An m machine flow shop

In deterministic scheduling, there are different performance objectives. One common objective is minimizing the makespan. The makespan is the total length of time needed to process all the jobs. Since we assume that the first job starts processing at time zero, the makespan is the departure time of the last job from the last machine. Thus if $C_1, C_2, \ldots, C_n$ represents the completion times of the n jobs the makespan, C_{max}, is

$$C_{max} = \max(C_1, C_2, \ldots, C_n).$$

However, we are generally unable to minimize the makespan in a stochastic system, since the makespan is a random variable. Alternatives will be discussed after the following example.

Let us consider a 2 machine flow shop with 3 jobs to be processed. The processing times are given in Table 1.1. Thus, we have a deterministic scheduling problem. First, let us assume we have infinite storage between the machines. In this case, we can use Johnson's (1954) rule to determine an optimal schedule minimizing the makespan. Johnson's rule states that job i should precede job j if

$$\min(s_{i,2}, s_{j,1}) < \min(s_{i,1}, s_{j,2}),$$

where $s_{i,k}$ is the processing time of job i on machine k. For our example, the optimal sequence is 1,2,3. Now let $D_{i,j}$ represent the departure time of job i from machine j. For the sequence 1,2,3 we can calculate the departure times as

$$D_{1,1} = 2,\ D_{1,2} = 2+4 = 6,$$

$$D_{2,1} = 2+3 = 5,\ D_{2,2} = 6+2 = 8,$$

Table 1.1

		Machines	
		1	2
	1	2	4
Jobs	2	3	2
	3	3	1

$$D_{3,1} = 5+3 = 8, \quad D_{3,2} = 8+1 = 9.$$

Thus, the makespan is 9. Now, let us assume that there is no storage space between machines. The departure times change because of the blocking phenomenon. For the sequence 1,2,3, the departure times are

$$D_{1,1} = 2, \; D_{1,2} = 2+4 = 6,$$

$$D_{2,1} = \max(2+3,6) = 6, \; D_{2,2} = 6+2 = 8,$$

$$D_{3,1} = \max(6+3,8) = 9, \; D_{3,2} = 9+1 = 10.$$

In this case the makespan is 10. Notice that job 2 is blocked on machine 1 by job 1. We have no reason to believe that the sequence 1,2,3 is optimal.

For a stochastic scheduling example, let us assume that the processing times are independent, exponentially distributed random variables with means given by Table 1.1. In the case of infinite storage space between the machines, we can use Bagga's (1970) rule to determine the optimum sequence minimizing the <u>expected</u> makespan. Similar to Johnson's rule, Bagga's rule states that job i should precede job j if

$$E[\min(S_{i,2},S_{j,1})] \leq E[\min(S_{i,1},S_{j,2})],$$

where $S_{i,k}$ is the processing time of job i on machine k, and $E[\cdot]$ represents the expected value. For our example, the optimal sequence is 1,2,3. Ku and Niu (1986) show that the Bagga's sequence not only minimizes the expected makespan, but also <u>stochastically</u> minimizes the makespan. This is a stronger result and in order to understand this, we need to discuss the partial orderings of random variables. Before discussing these partial orderings, we briefly describe the system with two identical parallel machines with intree precedence constraints.

We are interested in scheduling n jobs on two identical parallel machines. The jobs are subjected to precedence constraints, described by a graph that has the form of an intree as shown in Figure 1.2. This intree precedence constraint implies that each job has at most one successor. The job at the root of the tree is said to be at level 0. A job is said to be at level ℓ if its successor is at level ℓ-1. When the processing times are deterministic, one common performance objective is minimizing the makespan.

As an example, let us consider the processing of 6 jobs on two parallel machines with the intree precedence constraint as shown in Figure 1.3. Let the jobs on each level have the same processing time. Let the jobs on level 2 have processing times of 1, and jobs at level 0 and level 1 have processing times of 2. One common scheduling policy is the Highest Level First, HLF. In this policy whenever a machine is freed, the decision-maker chooses one of the jobs that are at the highest level and ready for processing. Using that policy the makespan in our example is 7. Another policy which yields a makespan of 6, is to process one job at level 1 and another job at level 2, and then to use the HLF policy. In the deterministic case, there is no efficient algorithm to determine the optimal scheduling policy.

As an example of stochastic scheduling, let us assume that the processing times are independent, exponentially distributed random variables. Let all jobs on any level be identically distributed; let jobs on level 1 and 0 have a mean of 2 and the jobs at level 2 have a mean of 1. We encounter the same problem here as in the flow shops:

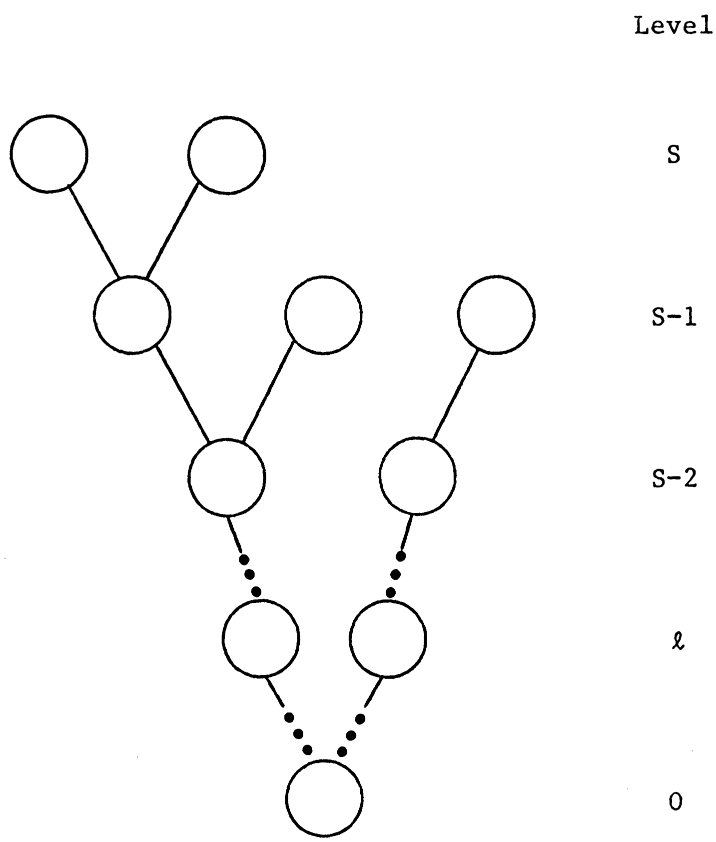

Figure 1.2

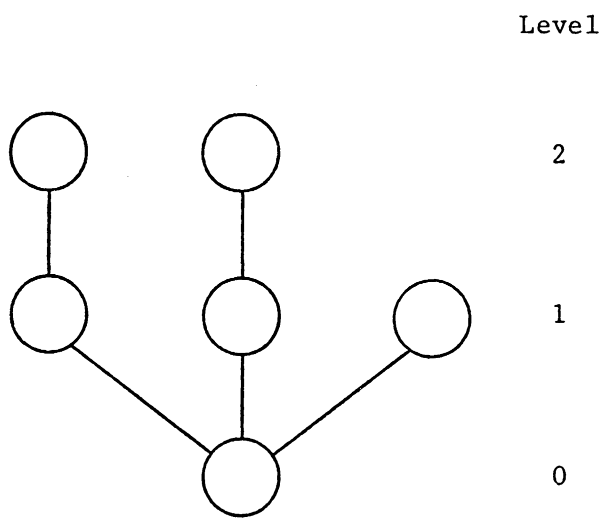

Figure 1.3

we are generally unable to minimize the makespan in a stochastic system, since the makespan is a random variable. Pinedo and Weiss (1985) show that the HLF policy minimizes the expected makespan when all jobs are exponentially distributed and all jobs at any level are identically distributed. Thus, in our example, the HLF policy minimizes one performance objective, the expected makespan. Pinedo and Weiss (1985) conjecture that in this case of HLF policy stochastically minimizes the makespan.

1.2 Partial Orderings of Random Variables.

We wish to say that one random variable is has a greater value than another random variable. There are many possible ways of expressing this idea: the most common ordering is simply to say X_i is bigger than X_j if

$$E[X_i] \geq E[X_j]$$

where $E[\cdot]$ represents the expected value. This ordering will be referred to as the expected value ordering. Even if $E[X_i] \geq E[X_j]$, and X_i and X_j are independent the $Pr\{X_i \geq X_j\}$ can be made arbitrarily close to zero. The expected value ordering is relatively weak. In many cases we need a stronger ordering. One stronger ordering is called stochastic ordering. We say X_i is stochastically greater than X_j if

$$Pr[X_i > t] \geq Pr[X_j > t]$$

for every value of t. (See Figure 1.4.) This stochastic ordering is represented by $X_i \underset{d}{\geq} X_j$ and will be referred to as d-ordering or as stochastic dominance. Here X_i is stochastically dominated by X_j. It

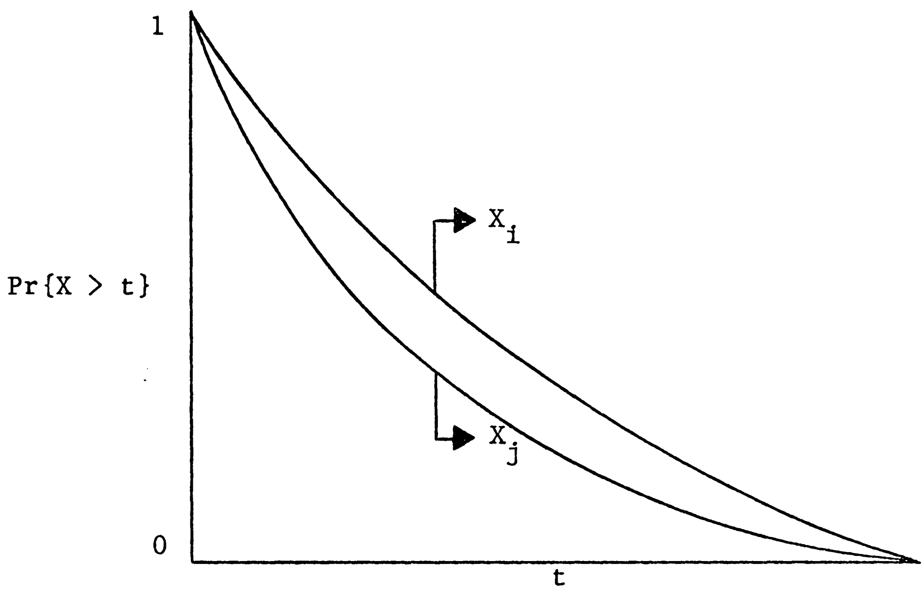

Figure 1.4

is in this sense that Ku and Niu (1986) have minimized the makespan for the two-machine flow shop with infinite storage between the machines. Note, if X_i is stochastically greater than X_j, then the expected value of X_i is greater than the expected value of X_j, since the area under the complimentary distribution gives the expected value. One of the strongest orderings is called the nonoverlapping ordering. X_i is said to be nonoverlapping and greater than X_j (represented by $X_i \gg X_j$) if

$$\Pr[X_i \gg X_j] = 1.$$

Furthermore, if X_i and X_j are independent then there exists a real number 'a' where

$$\Pr\{X_i > a\} = 1 \text{ and } \Pr\{X_j > a\} = 0.$$

In all these orderings only the expected value ordering is a complete ordering in the sense that two random variables may be ordered in the expected value but need not be d-ordered. There are many other orderings such as $\underset{c}{\geq}$, and $\underset{cc}{\geq}$, all of which are described in the Appendix.

The makespan is a random variable depending on the schedule. A sequence is said to stochastically minimize the makespan if the makespan of any other sequence is greater with respect to d-ordering. This is a strong ordering, the importance of which is explained in the Appendix.

In this dissertation, we assume certain orderings on the processing times of the jobs. Our objective is to find a schedule that will minimize the makespan in some sense.

Throughout this dissertation, we use $S_{i,j}$ to represent the processing time of job i on machine j. A makespan is said to be <u>stochastically minimized</u> if the makespan of this schedule is stochastically smaller than the makespan of any other sequence.

Random variables $X_1, X_2, \ldots, X_n$ are said to be exchangeable if $X_1, X_2, \ldots, X_n$ and $X_{\pi_1}, X_{\pi_2}, \ldots, X_{\pi_n}$ have the same distribution where $(\pi_1, \pi_2, \ldots, \pi_n)$ is any permutation of $(1,2,\ldots,n)$.

Certain sequences (e.g., SEPT, LEPT, SEPT-LEPT) often turn out to be optimal in flow shops. In <u>SEPT</u> sequence (Shortest Expected Processing Time first) the n jobs are arranged according to their expected processing times, with the shortest first, i.e.,

$$E[S_{1,j}] \leq E[S_{2,j}] \leq \cdots \leq E[S_{n,j}] \text{ for all j.}$$

Analogously, a <u>LEPT</u> sequence, <u>Longest Expected Processing Time first</u> is one in which

$$E[S_{1,j}] \geq E[S_{2,j}] \geq \cdots \geq E[S_{n,j}] \text{ for all j.}$$

A <u>SEPT-LEPT</u> sequence is one in which the first k jobs form a SEPT sequence and the last n-k jobs form a LEPT sequence, i.e.,

$$E[S_{1,j}] \leq E[S_{2,j}] \leq \cdots \leq E[S_{k,j}]$$

$$E[S_{k,j}] \geq E[S_{k+1,j}] \geq \cdots \geq E[S_{n,j}] \text{ for all j.}$$

Note that SEPT and LEPT sequences are special cases of SEPT-LEPT sequences. We use the term sequence in the case of flow shops since in this case the sequence completely specifies the schedule completely; that is the order of processing remains the same at every machine.

<u>Organization</u>.

This dissertation is divided into seven chapters and one appendix. Chapter 2 reviews the literature on stochastic flow shops and the results on scheduling stochastic jobs on parallel machines with intree precedence constraints.

Chapter 3 considers the problem of scheduling n nonoverlapping jobs in a m machine flow shop with any finite storage space. In the area of deterministic scheduling, Reddi and Ramamorthy (1972) and Wismer (1974) show that as the problem size increases, the amount of work required to find an optimal solution increases rapidly. More formally, they show that these problems are NP-Complete. For details on NP-Complete see Garey and Johnson (1979). Therefore, when we allow random processing times, we cannot hope to obtain optimal schedules unless we make additional assumptions. In Chapter 3, we assume neither that the processing times are independent nor that they are identically distributed. However, we do assume that the processing times are nonoverlapping and obtain an optimum sequence. We then show that the results obtained by Pinedo (1982a) and by Foley and Suresh (1984a) are speical cases of this result.

Chapter 4 considers the problem of scheduling machines in an m flow shop with zero intermediate storage between machines. Here we assume that all jobs are identical, i.e., the processing time of all jobs on a machine are identically distributed. Assembly lines can be modeled by this flow shop. We prove that the sequence that will minimize the expected departure time of nth job from the system, for every n, with m-2 deterministic machines and two stochastic machines, is to either a) arrange one of the stochastic machines first and the other one last or b) arrange the two stochastic machines together at one end. Suresh, Foley and Dickey (1985) obtained a similar result for a similar problem, assuming the machines to be identical and the jobs to be nonidentical. Pinedo (1982b) proved that when the two stochastic machines have symmetric density functions, arranging one of the two stochastic machines first and the other one last maximizes the throughput. Maximizing throughput does not necessarily minimize the expected departure time of the nth job, for all n. However, minimizing the expected departure time maximizes the throughput.

Chapter 5 considers the problem of scheduling jobs in an m machine flow shop with zero intermediate storage between machines. We assume that the processing times of a job on all the machines are identically distributed. We extend an important result by Muth (1979), which states that the makespan for any sequence of jobs 1,2,...,n is stochastically the same as the makespan for the reversed sequence n,...,2,1. The author required that processing time of the job i on all the machines be independent and identically distributed. We prove that the same result is true if the processing times of job i on different machines are exchangeable. Clearly independent and identically distributed random variables are exchangeable, but exchangeable random variables need not be independent. Consequently, we extend earlier results obtained by Pinedo (1982a, 1982b) and by Foley and Suresh (1984a and 1984b).

Next we consider scheduling (n+2) stochastic jobs; n of them are nonoverlapping, and the other two need not have any stochastic ordering. Suresh, Foley and Dickey (1985) solved this problem when the n jobs are deterministic. From past results, one might expect that the optimum sequence to have the n nonoverlapping jobs to be in a SEPT-LEPT sequence. However, we will show that this is not true in general. Sometimes it pays to break up the SEPT-LEPT sequence.

The last result in Chapter 5 schedules a group of jobs to minimize the expected makespan. Jobs are divided into p different types, and we have n_t jobs of type t, $t = 1,2,\ldots,p$. We assume that the processing times of one set of jobs are nonoverlapping with any other set. The optimum sequence minimizing the expected makespan is a specific SEPT-LEPT sequence where the SEPT portion consists of $\frac{n_t}{2}$ jobs of type t, $t = 1,2,\ldots,p$, and the LEPT portion consists of the remaining jobs. This result generalizes an earlier result by Foley and Suresh (1984a).

Chapter 6 considers the scheduling of n jobs on two parallel machines. The jobs are subject to some intree precedence constraints. We resolve a conjecture that appeared in Pinedo and Weiss (1985) by giving conditions under which the conjecture is true and examples to prove that the conjecture is false in general.

Chapter 7 draws conclusions from our results. We also give some discussions for further research.

The Appendix gives a detailed discussion of the results on stochastic orderings that are used in this dissertation. We also discuss the importance of the orderings and their relations to each other. Readers unfamiliar with stochastic orderings are encouraged to read the Appendix.

CHAPTER 2

LITERATURE REVIEW

This chapter will first review some of the literature on stochastic flow shops and will then review some of the results in scheduling jobs on parallel machines.

2.1 Stochastic Flow Shops.

In this section we use the terms sequence and schedule interchangeably, since in our case the sequence specifies the schedule completely; that is, the order of processing remains the same at every machine. Pinedo and Schrage (1981) provide a nice survey on stochastic shop scheduling. Since the problem is so complex even in a deterministic case, it will be impossible to deal with the stochastic case without suitable assumptions. The results in the literature can be roughly divided in two groups: those that assume all machines are independent and identical (Assumption IIM) and those that assume all jobs are independent and identical (Assumption IIJ).

Assumption IIM. Here we assume that all the machines are identical and that the processing times are independent (IIM). The processing time of job i on machine j depends on job i and not on machine j. Thus, if $S_{i,j}$ represents the processing time of job i on machine j then $Pr[S_{i,j} \leq t] = F_i(t)$ for all machines j. That is $S_{i,j}$ for $j = 1,2,...,m$ are identically distributed. In addition to this we assume that all random variables are mutually independent.

Assumption IIJ. Here we assume that all the jobs are identical and that the processing times are independent (IIJ). Specifically, here $Pr[S_{i,j} \leq t] = F_j(t)$ for all jobs i where $S_{i,j}$ represents the processing time of job i on machine j. Here $S_{i,j}$ for i = 1,2,...,n are identically distributed. In addition to this we assume that all random variables are mutually independent.

With Assumption IIM we are concerned with the sequencing of jobs. With Assumption IIJ we are concerned with the sequencing of the machines.

Duality.

Pinedo (1982a), in the case of a flow shop with infinite intermediate storage, establishes a flow shop that he calls the dual flow shop. Let $S_{i,j}$ denote the processing time of job i on machine j where i = 1,2,...,m and j = 1,2,....n for a flow shop called a 'primal flow shop'. Now consider a second flow shop with m machines and n jobs. Let $S_{i,j}$ denote the processing time of job j on machine i for a flow shop called 'dual flow shop' or a 'dual'. Pinedo shows the following deterministic result:

$$D^p_{i,j} = D^d_{j,i}$$

where $D^p_{i,j}$ is the departure time of job i from machine j in the primal flow shop and $D^d_{j,i}$ is the departure time of job j from machine i in the dual flow shop. With this result it is sufficient if we study stochastic flow shops using either Assumption IIJ or Assumption IIM, since the results obtained under either assumption can be applied to the other.

2.2 Stochastic Flow Shops with Infinite Intermediate Storage.

Weber (1979) shows that with Assumption IIJ, and with all the processing times being exponentially distributed random variables the departure epoch of every job is stochastically the same for any sequence. Tembe and Wolff (1974) show that with Assumption IIJ and two machines, one deterministic and the other stochastic, the departure epoch of every job is stochastically minimized by placing the deterministic machine first. Pinedo (1982a) shows that with Assumption IIM and two jobs, one being deterministic and the other stochastic, the makespan is stochastically minimized by sequencing the deterministic job first. Tembe and Wolff's result shows that if we have n jobs, the departure epoch of the n^{th} job is stochastically minimized thus stochastically minimizing the makespan. Tembe and Wolff have two machines, n jobs, and used Assumption IIJ. Pinedo has two jobs, n machines, and used Assumption IIM. Thus, the system used by Pinedo is a dual of the system used by Tembe and Wolff. The final results consequently follow from Pinedo's (1982a) proof on duality.

Tembe and Wolff (1974) also show that with Assumption IIJ and nonoverlapping processing time on m machines the LEPT sequence stochastically minimizes the departure epoch of every job. Pinedo (1982b) extends this result, showing that any SEPT-LEPT sequence stochastically minimizes the departure epoch of every job. Pinedo (1982a) shows that with Assumption IIM and nonoverlapping processing time of m jobs, any SEPT-LEPT sequence minimizes the expected makespan. Again, note that the system of Pinedo (1982b) is in the same sense the dual of his previous system. Though Pinedo proves that any SEPT-LEPT sequence minimizes the expected makespan, the result can be extended and it can be shown from his own results on duality that any SEPT-LEPT stochastically minimizes the makespan.

Pinedo (1982b) also shows that with Assumption IIJ, m-2 deterministic machines, and two stochastic machines, the departure epoch of every job is stochastically minimized if one stochastic machine is placed first and the other last. Pinedo (1982a) shows that with Assumption IIM, n-2 deterministic jobs, and the other two stochastic jobs, the makespan is stochastically minimized if one stochastic job is placed first and the other last. Not surprisingly, the system of Pinedo (1982a) and Pinedo (1982b) are again dual of one another. The results can be obtained using the duality.

Muth (1979) shows that with Assumption IIJ the makespan for any sequence of machines is stochastically the same as the makespan if the sequence of machines is reversed. This is an extremely important result, since it holds regardless of the storage capacity. As a consequence of duality, Pinedo (1982a) gives an important result based on Muth's result. In a flow shop without blocking with Assumption IIM the makespan for any sequence of jobs is stochastically the same as the makespan of the reversed sequence of jobs.

2.3 Stochastic Flow Shop with Zero Intermediate Storage.

The duality result obtained by Pinedo (1982) is not true in general to stochastic flow shops with zero intermediate storage. We therefore study this system under Assumption IIJ and Assumption IIM.

2.3.1 With Assumption IIJ.

Dattatreya (1978) shows that with Assumption IIJ and two machines, one deterministic and the other stochastic, placing the deterministic job first stochastically minimizes the departure epoch of every job.

He also shows that with the same assumption and nonoverlapping processing time on m machines the LEPT sequence stochastically minimizes the departure epoch of every job. The reader may recall that these are the same results that Tembe and Wolff (1974) obtained in a flow shop without blocking.

As mentioned in Section 2.2, Muth's result on makespan is true even in the case of stochastic flow shop with blocking.

Pinedo (1982b), working with Assumption IIJ, investigates the output rate in steady state, i.e., the reciprocal of the mean interdeparture time. This rate is called the capacity, i.e., the long run output rate with an infinite number of jobs awaiting processing at the first machine. Though capacity and departure epoch are not the same quantities, it is clear that if a sequence stochastically minimizes the departure epoch of every job, then the capacity is maximized. However, if the capacity is maximized by a sequence of machines, the departure epoch of every job is not necessarily minimized. Pinedo shows that if m-1 stations are identical and have deterministic service times, and if one station has an arbitrary stochastic service time distribution, the capacity does not depend on the sequence. He also shows that when the processing time are nonoverlapping on the machines, the capacity does not depend on the sequence. The capacity in this system is the reciprocal of the expected service time of the slowest machine.

In a system with m-2 deterministic machines having processing times of one unit each, and with two stochastic machines not necessarily identical, both having a mean processing time of one unit and symmetrical density functions, the capacity is maximized when either one of the stochastic machines is set up at the beginning and the other at the end of the sequence. Finally, Pinedo shows that in a system with m-2 identical machines having identical processing time distribution and faster than the other two nonidentical machines, the capacity is maximized when one of the slow stations is set up first and the other one last.

Notice that in the first two results the capacity does not depend on the sequence of machines. In the last two results the throughput does depend on the sequence of machines.

2.3.2 With Assumption IIM.

Pinedo (1982a) working with Assumption IIM was interested in minimizing the expected makespan. He shows that if the processing time of the jobs are nonoverlapping then the expected makespan is minimized if and only if the sequence is SEPT-LEPT. Foley and Suresh (1984a) extend this result to show that makespan is stochastically minimized by any SEPT-LEPT sequence. Pinedo (1982a) also shows that when the processing time of the jobs are d-ordered for a sequence 1,2,...,n with the slowest job first, the expected makespan on two machines is minimized by the sequence n, n-2,...,4,2,1,3,...,n-1 (or by Muth's result its reverse). Note that this sequence is a SEPT-LEPT sequence but not any SEPT-LEPT sequence. Comparing the two results the hypothesis is weakened from nonoverlapping to d-ordered, and the final result is weakened from any SEPT-LEPT sequence on m machines to a particular SEPT-LEPT sequence on two machines.

2.4 Lack of Duality.

Pinedo (1982a) points out that the same kind of duality relationship described in Section 2.1 for a stochastic flow shop without blocking does not exist in a flow shop with blocking. For counterexamples see Suresh (1984). In systems without blocking, due to the duality proved by Pinedo (1982a), results with Assumption IIJ are dual to results with Assumption IIM in dual systems. Unfortunately this is not the case in stochastic flow shops with blocking. Thus, a separate study is necessary with Assumption IIM. This dissertation considers the relatively new area of stochastic flow shop and obtains some new results.

2.5 Scheduling Jobs on Parallel Machines.

considerable work has been done in the area of scheduling deterministic jobs on parallel machines when the jobs are subject to intree precedence constraints. For a survey see Lawler, Lenstra and Rinooy Kan (1982) and Garey, Johnson, Tarjan and Yannakakis (1982). Hu (1981) showed that the HLF policy is optimal when there are m machines and n identical deterministic jobs which are subject to intree precedence constraints.

Not much work has been done in the area of stochastic systems. Chandy and Reynolds (1975) showed that on two parallel machines when the jobs are all independent and identically distributed (i.i.d.) and exponentially distributed, the HLF policy minimizes the expected the makespan. Pinedo and Weiss (1985) generalize this result to a system where all jobs at level ℓ have i.i.d. exponentially distributed random variables with parameter μ_ℓ. Frostig (1986) shows that when the random variables are discrete and IFR (increasing failure rate), HLF policy with preemption minimizes the makespan. In this dissertation we study a system where we stochastically minimize the makespan. We also resolve a conjecture that appeared in Pinedo and Weiss (1985).

CHAPTER 3

SCHEDULING JOBS IN A FLOW SHOP WITHOUT ASSUMPTIONS OF INDEPENDENCE OR IDENTICAL DISTRIBUTIONS

3.1 Introduction.

In this chapter we determine the optimal schedule for minimizing the expected makespan of n jobs with nonoverlapping processing time distributions on m machines. We give some additional results that determine the optimal schedule in certain cases for other objective functions, such as stochastically minimizing the makespan. As pointed out earlier in Chapter 1, it is difficult to analyze stochastic flow shops without assumptions of independence and identical distributions. In this chapter we make no such assumptions; we allow the processing times to be dependent. Consequently, a study of this system requires a strong ordering of the processing times.

The system works as follows. We have n nonoverlapping jobs. Let $S_{i,j}$ represent the random processing time of job i on machine j. We say that job k is nonoverlapping greater than job i if

$$\min\{S_{k,j}:\ 1 \leq j \leq m\} \geq \max\{S_{i,j}:\ 1 \leq j \leq m\}.$$

Thus, the smallest processing time of the slow job is bigger than the largest processing time of the fast job.

3.2 Results.

We will show that the following sequence is optimal in an m machine flow shop. Separate the jobs into two sets A and B such that A contains jobs where

$$E[S_{i,1}] \leq E[S_{i,m}]$$

and B contains jobs where

$$E[S_{i,1}] > E[S_{i,m}].$$

An optimal sequence is obtained by arranging the jobs of set A in SEPT, arranging the jobs of set B in LEPT, and concatenating the two. This SEPT-LEPT sequence is the optimal sequence in the sense of minimizing the expected makespan.

Lemma 3.1. In an m machine flow shop with n nonoverlapping jobs, the makespan of any sequence that is not SEPT-LEPT can be decreased if that sequence is changed to SEPT-LEPT.

Proof: We argue that the makespan of any sequence with the slowest job in position k must be at least as large as that of the SEPT-LEPT sequence with the slowest job in the same position k, but with the jobs before k sorted in increasing order and the jobs after k sorted in decreasing order.

The makespan of the SEPT-LEPT sequence is

$$C_{max} = T_{SEPT} + T_k + T_{LEPT}$$

where

$$T_{SEPT} = S_{1,1} + S_{2,1} + S_{3,1} + \cdots + S_{k-1,1}$$

$$T_k = S_{k,1} + S_{k,2} + S_{k,3} + \cdots + S_{k,m}$$

$$T_{LEPT} = S_{k+1,m} + S_{k+2,m} + \cdots + S_{n,m}.$$

One may observe this by noting that the jobs are nonoverlapping and

$$T_{SEPT} = D_{k-1,1},$$

$$T_k = D_{k,m} - D_{k-1,1},$$

$$T_{LEPT} = D_{n,m} - D_{k,m},$$

where $D_{i,j}$ denotes the departure time of job i from machine j. For any other sequence with the slowest job in position k, the earliest that job k can begin processing on machine 1 is the sum of the processing times of the first k-1 jobs on machine 1, i.e., T_{SEPT}. Job k can then leave the last machine no earlier than $T_{SEPT} + T_k$, since T_k is the total processing time of job k. T_{LEPT} is the sum of the processing times of the remaining jobs on the last machine. Hence, the last job can leave the last machine no sooner than $T_{SEPT} + T_k + T_{LEPT}$, which is the makespan of the SEPT-LEPT sequence. □

Theorem 3.1. In an m machine flow shop with n nonoverlapping jobs, the SEPT-LEPT sequence is optimal where the SEPT segment of this sequence consists of jobs in which

$$E[S_{i,1}] < E[S_{i,m}]$$

and the LEPT segment consists of jobs in which

$$E[S_{i,1}] > E[S_{i,m}].$$

Proof: From Lemma 3.1, a SEPT-LEPT sequence is optimal. Take a SEPT-LEPT sequence. If there is a job in which

$$E[S_{i,1}] < E[S_{i,m}]$$

is scheduled after the slowest job, moving the job in front of the slowest job decreases the expected makespan by

$$E[S_{i,m}] - E[S_{i,1}] > 0.$$

Hence, jobs in which

$$E[S_{i,1}] < E[S_{i,m}]$$

should be scheduled before the slowest job. A similar argument shows that jobs in which

$$E[S_{i,1}] > E[S_{i,m}]$$

should be scheduled after the slowest job. □

Monma and Rinooy Kan (1983) solve a deterministic version of this problem by using the critical path method. Despite the fact that the jobs are nonoverlapping, this result is still stochastic. The optimal sequence is <u>not</u> optimal for every realization. Only the expected makespan is minimized by the previous theorem as the following example shows.

Suppose we have two machines and three jobs. Let the distributions of the processing times be as follows.

$$S_{1,1} = S_{1,2} = 4$$

$$S_{2,1} = S_{2,2} = 3$$

$$P\{S_{3,1} = i\} = \begin{cases} 0 \text{ w.p. } .5, \\ 1 \text{ w.p. } .5. \end{cases}$$

$$P\{S_{3,2} = i\} = \begin{cases} 0 \text{ w.p. } .5, \\ 2 \text{ w.p. } .5. \end{cases}$$

It is easy to see that job 1 is bigger than job 2, which in turn is bigger than job 3. A sequence that minimizes the expected makespan is (3,2,1). However, if $S_{3,1} = 1$ and $S_{3,2} = 0$, the makespan of the sequence (2,1,3) is 11; this is better than the makespan of (3.1.2), which is 12. With a stronger relationship between $S_{i,1}$ and $S_{i,m}$, we can obtain a deterministic result. For example, if for every i, either

$$S_{i,1} \geq S_{i,m} \text{ or } S_{i,m} \geq S_{i,1},$$

we can guarantee that the optimal sequence described in Theorem 1 is nonoverlapping optimal. That is, for every realization, the makespan of any other schedule is at least as large as that of the optimal schedule. This result would follow from the results of Monma and Rinooy Kan (1983).

Thus far, we have described two extreme cases. A large number of cases lying between these two extreme cases can be captured as follows. Specifically, assume that we have a (partial) ordering $\geq_g$ such that the following hold:

i) $X \geq_g Y \iff E[f(X)] \geq E[f(Y)]$ for all $f \in C_g$.

ii) $X \geq_g Y \implies X + a \geq_g Y + a$ for $a \geq 0$.

Requirement (i) states that we can define $X \geq_g Y$ by requiring that $E[f(X)] \geq E[f(Y)]$ for all f in a set C_g. The usual <u>stochastic ordering</u> is to take C_g to be the class of all non-negative, non-decreasing functions. Then X is said to be <u>stochastically greater</u> than Y. Other common orderings are the <u>convex ordering</u>, where C_g is taken to be all non-negative, non-decreasing, convex functions; and the <u>concave ordering</u>, where C_g is all non-negative, non-decreasing, concave functions. There are a variety of other orderings; for information see Stoyan (1977). (For more on stochastic orderings see the Appendix.) Requirement (ii) implies that if X is greater than Y then adding a non-negative constant to both random variables preserves the ordering.

<u>Theorem 3.2.</u> Consider an m machine flow shop with n nonoverlapping jobs. Let job k be the slowest job in the nonoverlapping sense. Suppose that for each i, all of the following conditions hold:

i) either $S_{i,1} \geq_g S_{i,m}$ or $S_{i,1} \geq_g S_{i,m}$.

ii) $S_{i,1}$ is mutually independent of $\sum_{\substack{j=1 \\ j \neq k,i}}^{n} S_{j,a_j} + \sum_{j=1}^{m} S_{k,j}$.

iii) $S_{i,m}$ is mutually independent of $\sum_{\substack{j=1 \\ j \neq k,i}}^{n} S_{j,a_j} + \sum_{j=1}^{m} S_{k,j}$.

where a_j can be either 1 or m, then the makespan of the optimal sequence, M*, is less than the makespan of any other sequence, M, in the same sense, g, i.e., $M^* \underset{g}{<} M$, where M* is the makespan of the sequence, analogous to Theorem 3.1.

Proof: From Lemma 3.1 we know that the makespan of any sequence which is not SEPT-LEPT can be decreased if that sequence is changed to SEPT-LEPT. Without loss of generality, let us assume that

$$S_{i,m} \underset{g}{<} S_{i,1}.$$

Let M be the makespan of the SEPT-LEPT sequence that has job i scheduled before the biggest job and let M' be the makespan of the same SEPT-LEPT sequence except with job i scheduled after the biggest job. In other words, in the first case job i is scheduled in the SEPT portion of the sequence and in the second case i is scheduled in LEPT portion. It suffices to show that

$$E[f(M')] < E[f(M)] \text{ for } f \in C_1.$$

First,

$$E[f(M)] = E[f(T_{SEPT} + T_k + T_{LEPT})]$$

$$= E[E[f(S_{i,1} + b) | T_{SEPT} + T_k + T_{LEPT} - S_{i,1} = b]].$$

Similarly,

$$E[f(M')] = E[f(T'_{SEPT} + T_k + T'_{LEPT})]$$

$$= E[E[f(S_{i,m} + b) | T'_{SEPT} + T_k + T'_{LEPT} - S_{i,m} = b]].$$

From Lemma 3.1 we know that

$$T'_{SEPT} = T_{SEPT} - S_{i,1}$$

and

$$T_{LEPT} = T'_{LEPT} - S_{i,m}.$$

Therefore,

$$T'_{SEPT} + T_k + T'_{LEPT} - S_{i,m} = T_{SEPT} + T_k + T_{LEPT} - S_{i,1}.$$

Since $S_{i,1} \underset{g}{\geq} S_{i,m}$ and $T_{SEPT} + T_k + T_{LEPT} - S_{i,1}$ is independent of $S_{i,1} + b$ (and similarly $T'_{SEPT} + T_k + T'_{LEPT} - S_{i,m}$ is independent of $S_{i,m} + b$), from conditions (ii) and (iii) we have

$$E[f(M')] \leq E[f(M)];$$

therefore,

$$M' \underset{g}{\leq} M.$$

Using this argument repeatedly, we have the same optimal sequence as in Theorem 3.1 with

$$M^* \underset{g}{\leq} M. \qquad \square$$

An example that satisfies conditions (ii) and (iii) of Theorem 3.2 is when the processing times of all jobs on the first and last machines, and the processing times of the slowest job on all machines are all mutually independent.

All results are true for any amount of storage, as is clear from the proof of Lemma 3.1. That is, even if storage is available, the makespan cannot be reduced. The results of Pinedo (1982a) and of Foley and Suresh (1984a) can be interpreted as follows. They assume that the processing times of job i on all machines are independent and identically distributed. From Theorem 3.1, if $S_{i,1} \underset{d}{=} S_{i,m}$, it does not matter whether job i is in set A or B. It is interesting to note that with these additional assumptions, any SEPT-LEPT sequence not only minimizes the expected makespan but also stochastically minimizes the makespan. However, without the additional assumption that the processing times are i.i.d., there is a unique SEPT-LEPT sequence that minimizes the expected makespan.

Since the orderings on the processing times for the above two theorems are extremely strong, we would like to get results with some weaker orderings. Let us consider a case where the processing times are LR (Likelihood Ratio) ordered. The LR-ordering lies between the nonoverlapping ordering and the d-ordering, i.e., the LR ordering is stronger than the d-ordering but weaker than the nonoverlapping ordering. However, in the case of flow shops with infinite storage and with the jobs LR-ordered, it is not possible to obtain an algorithm for the optimal sequence without additional information, as shown by the following example. In the example, we assume that the processing times of a job on all machines are identically distributed and that all processing times are mutually independent.

Example 3.1: We have five jobs and two machines and

$S_{1,j} \underset{L}{\geq} S_{2,j} \underset{L}{\geq} \cdots \underset{L}{\geq} S_{5,j}$.

i	3	4
$\Pr\{S_{1,j} = i\}$	0.4	0.6
$\Pr\{S_{2,j} = i\}$	0.65	0.35
$\Pr\{S_{3,j} = i\}$	0.75	0.25

i	3	4
$\Pr\{S_{4,j} = i\}$	0.85	0.15
$\Pr\{S_{5,j} = i\}$	0.95	0.05

Expected makespan for 5,2,1,3,4 = 20.29826. With some algebra, we can show that all other sequences have a expected makespan strictly greater than 20.29826. By changing the probabilities of $S_{1,j}$ to

$$\Pr\{S_{1,j} = 3\} = 0.5 \text{ and } \Pr\{S_{1,j} = 4\} = 0.5,$$

we can show that the sequence 5,2,1,3,4 is not optimal. With some algebra we can show that

Expected makespan for 5,2,1,3,4 = 20.157505,

Expected makespan for 5,3,1,2,4 = 20.157006.

Clearly, in this case 5,2,1,3,4 is <u>not</u> the optimal sequence. In this case, the optimal sequence is 5,3,1,2,4.

However, as a speical case, in flow shops with infinite intermediate storage, we can show from Weber's (1979) result and he duality result of Pinedo (1982a) that when the processing times are independent and exponentially distributed, the makespan is stochastically the same for any sequence.

3.3 Summary.

Until now, stochastic flow shops have been studied under different assumptions. Either the jobs are assumed to be identical, or the machines are assumed to be identical. In this chapter, we do not make either of these assumptions for Theorem 3.1. We also allow the processing times to be dependent in Theorem 3.1. It is rare in stochastic scheduling literature that a result is obtained without any assumptions about independence or identical distributions. All the results are true for any amount of storage. In the next few chapters we will study the m machine flow shop, assuming the processing times to be independent.

CHAPTER 4

SEQUENCING MACHINES IN A FLOW SHOP WITH ASSUMPTION IJ

In this chapter, we are concerned about sequencing the m machines in a flow shop. We assume that there is no intermediate storage between machines. We also use Assumption IJ.

Assumption IJ. Here we assume that all the jobs are identical (IJ). Specifically, we assume that the random vectors $S_{\cdot j} = (S_{1,j}, S_{2,j}, \ldots, S_{n,j})$ of processing times on machine j are independent, and that the elements within each vector are exchangeable for each j, where $S_{i,j}$ represents the processing time of job i on machine j. This set of assumptions implies that the processing times of all jobs on a particular machine are identically distributed but not necessarily independent.

Recall that Assumption IIJ (defined in Chapter 2) requires that all processing times be independent. In other words, Assumption IIJ is stronger than Assumption IJ. All our results use Assumption IJ.

In this section we are interested in finding the optimal sequence of machines, minimizing the expected makespan, given m-2 identical deterministic machines and two stochastic machines. Suresh, Foley and Dickey (1985) studied a similar problem, using a different assumption. They assumed that the processing times of a job on all machines were identically distributed but that there were n-2 identical deterministic jobs and two stochastic jobs. Therefore, they were interested in finding the optimal sequence for the jobs. The system described in this chapter can be considered a dual of the system studied by Suresh, Foley and Dickey (1985). The results obtained by these authors cannot be directly applied to the dual system using the duality result of Section 2.1, since we do not have infinite storage between machines.

We will first prove Lemma 4.1, which will be needed for the proof of our main result. Although we have required that all jobs be available in front of the first machine at time zero, Lemma 4.1 will also hold for an arbitrary arrival process $t_1 \leq t_2 \leq \ldots \leq t_n$ where t_i represents the arrival epoch of job i. In Lemma 4.1 we consider two systems. System A consists of $k > 0$ deterministic machines with unit processsing times placed between two stochastic machines X and Y with X placed in the first position and Y last. System B is the same as System A except that k+1 deterministic machines are placed between X and Y. X_i and Y_i represent the processing times of job i on the machines X and Y.

<u>Lemma 4.1</u>. Let C_i^a and C_i^b denote the completion times of job i in System A and System B respectively. Then we have

$$C_i^b - C_i^a \leq 1 \text{ for } i = 1,2,\ldots,n,$$

or, equivalently,

$$D_{i,k+3}^b - D_{i,k+2}^a \leq 1 \text{ for } i = 1,2,\ldots n, \tag{4.1}$$

where $D_{i,j}^a$ and $D_{i,j}^b$ represent the departure times of job i from machine j in Systems A and B, respectively. That is, the difference in the departure times of any job from the two systems can be at most 1. Furthermore,

$$D_{i,j}^b - D_{i,j}^a \leq 0 \text{ for } j = 1,2,\ldots,k+1;\ i = 1,2,\ldots n. \tag{4.2}$$

Remark: Note that Lemma 4.1 is a deterministic result: it holds for every realization of the random variables.

Proof. To prove (4.1) it suffices to prove

$$D^b_{i,k+2} - D^a_{i,k+1} \leq 1. \tag{4.1a}$$

since

$$D^b_{i,k+3} = D^b_{i,k+2} + Y_i,$$

and

$$D^a_{i,k+2} - D^a_{i,k+1} + Y_i.$$

Let $P_{i,j}$ be the proposition that (4.2) holds for the given i and j. Let Q_i be the proposition that (4.1a) holds for the given i. The proof of this lemma involves a messy induction, so we give a brief outline.

Clearly, $P_{1,j}$ holds for $1 \leq j \leq k+1$. Next, we show that for $1 \leq i < n$

$$P_{i,1} \wedge P_{i,2} \Longrightarrow P_{i+1,1}, \tag{4.3}$$

i.e., if $P_{i,1}$ and $p_{i,2}$ hold, then $P_{i+1,1}$ holds. Subsequently we will show that for $1 \leq i < n$,

$$P_{i+1,j-1} \wedge P_{i,j+1} \Longrightarrow P_{i+1,j} \text{ for } 2 \leq j < k. \tag{4.4}$$

Note that the proof of (4.3) and (4.4) do not prove $P_{2,k+1}$ (or $P_{i,j}$ where $i+j \geq k+3$), since (4.4) requires $P_{1,k+2}$ to prove $P_{2,k+1}$, but $P_{1,k+2}$ is undefined. To accomplish this, we note that Q_1 holds and show that

$$Q_i \wedge P_{i+1,k} \Longrightarrow Q_{i+1} \Longrightarrow P_{i+1,k+1}. \tag{4.5}$$

From these relations, the truth of $P_{i,j}$ and Q_i can be derived for $1 \leq i \leq n$ and $1 \leq j \leq k+1$. Thus, it suffices to prove (4.3), (4.4), and (4.5).

We will first prove (4.3). Since $P_{i,1}$ and $P_{i,2}$ hold, we have

$$D^b_{i,1} - D^a_{i,j} \leqslant 0, \tag{4.6}$$

and

$$D^b_{i,2} - D^a_{i,2} \leqslant 0. \tag{4.7}$$

For System A, we have

$$D^a_{i+1,1} = \max(t_{i+1} + X_{i+1}, D^a_{i,1} + X_{i+1}, D^a_{i,2}),$$

and similarly for System B,

$$D^b_{i+1,1} = \max(t_{i+1} + X_{i+1}, D^b_{i,1} + X_{i+1}, D^b_{i,2}).$$

Using (4.6) and (4.7), it is clear that

$$D^b_{i+1,1} - D^a_{i+1,1} \leqslant 0,$$

which is proposition $P_{i+1,1}$; thus, we have proved (4.3).

To prove (4.4), we note that the departure times of Systems A and B for $1 < j \leqslant k$ can be derived as follows:

$$D^a_{i+1,j} = \max(D^a_{i+1,j-1} + 1, D^a_{i,j+1}); \tag{4.8}$$

$$D^b_{i+1,j} = \max(D^b_{i+1,j-1} + 1, D^b_{i,j+1}). \tag{4.9}$$

Since $P_{i+1,j-1}$ and $P_{i,j+1}$ hold, we have

$$D^b_{i+1,j-1} - D^a_{i+1j-1} \leqslant 0, \tag{4.10}$$

and

$$D^b_{i,j+1} - D^a_{i,j+1} \leqslant 0. \tag{4.11}$$

Using (4.10) and (4.11) in (4.8) and (4.9), we obtain

$$D^b_{i+1,j} - D^a_{i+1,j} \leq 0 \text{ for } 1 < j \leq k$$

which is $P_{i+1,j}$; thus, we have proved (4.4).

In (4.5) since $P_{i+1,k}$ and Q_i hold, we have

$$D^b_{i+1,k} - D^a_{i+1,k} \leq 0, \tag{4.12}$$

and

$$D^b_{i,k+2} - D^a_{i,k+1} \leq 1. \tag{4.13}$$

For System A we have

$$\begin{aligned} D^a_{i+1,k+1} &= \max(D^a_{i+1,k} + 1,\ D^a_{i,k+2}), \\ &= \max(D^a_{i+1,k} + 1,\ D^a_{i,k+1} + Y_i), \end{aligned} \tag{4.14}$$

since

$$D^a_{i,k+2} = D^a_{i,k+1} + Y_i.$$

Similarly for System B we have

$$D^b_{i+1,k+2} = \max(D^b_{i+1,k} + 2,\ D^b_{i,k+2} + 1,\ D^b_{i,k+2} + Y_i).$$

Using (4.13), we have

$$D^b_{i+1,k+2} \leq \max(D^b_{i+1,k} + 2,\ D^a_{i,k+1} + 2,\ D^a_{i,k+1} + 1,\ Y_i). \tag{4.15}$$

We now consider two cases. If $Y_i \leq 1$, (4.15) using (4.12) can be rewritten as

$$\begin{aligned} D^b_{i+1,k+2} &\leq \max(D^b_{i+1,k} + 2,\ D^a_{i,k+1} + 2) \\ &\leq \max(D^b_{i+1,k} + 2,\ D^a_{i+1,k} + 2) \leq D^a_{i+1,k} + 2, \end{aligned} \tag{4.16}$$

since

$$D^a_{i+1,k} \geq D^a_{i,k+1}. \tag{4.17}$$

In this case, using (4.17), (4.14) can be rewritten as

$$D^a_{i+1,k+1} = D^a_{i+1,k} + 1. \tag{4.18}$$

Comparing (4.16) with (4.18), it is clear that

$$D^b_{i+1,k+2} - D^a_{i+1,k+1} \leq 1,$$

which is Q_{i+1}. If $Y_i > 1$, (4.15) can be rewritten as

$$D^b_{i+1,k+2} \leq \max(D^b_{i+1,k} + 2,\ D^a_{i,k+1} + 1 + Y_i). \tag{4.19}$$

Using (4.12) and then comparing (4.19) with (4.14), it is clear that

$$D^b_{i+1,k+2} - D^a_{i+1,k+1} \leq 1.$$

This proves the first assertion in (4.5). We prove the second assertion (4.5) by noting that

$$D^b_{i+1,k+2} - D^b_{i+1,k+1} \geq 1,$$

and

$$D^b_{i+1,k+2} - D^a_{i+1,k+1} \leq 1$$

imply

$$D^b_{i+1,k+1} - D^a_{i+1,k+1} \leq 0. \qquad \square$$

An interesting footnote to this lemma is that for the first k+1 jobs, the difference in the departure times from the two systems is exactly one unit. Lemma 4.1 also holds for a flow shop with infinite storage between machines, although the expressions for the departure times will be different. In the case of infinite storage for a general sequence with k machines, the departure times are given by the following equations:

$$D_{1,1} = t_1 + S_{1,1}$$

$$D_{1,j} = D_{1,j-1} + S_{1,j} \text{ for } j = 2,3,\ldots,k$$

$$D_{i,1} = \max(t_i, D_{i-1,1}) + S_{i,1} \text{ for } i = 1,2,\ldots,n$$

$$D_{i,j} = \max(D_{i,j-1}, D_{i-1,j}) + S_{i,j} \text{ for } i = 1,2,\ldots,n;\ j = 2,3,\ldots,k-1$$

$$D_{i,k} = D_{i,k-1} + S_{i,k} \text{ for } i = 1,2,\ldots,n.$$

However, all the steps of the proof for Lemma 4.1 are still valid when there is infinite storage between the machines.

In Theorem 4.1, we have m-2 deterministic machines with unit processing times and two stochastic machines denoted by X and Y. Here a machine sequence is represented by $I_kXI_gYI_h$, denoting k deterministic machines, followed by machine X, then g deterministic machines, then machine Y, and finally h deterministic machines. k, g, h are all nonnegative. If g = 0, then machines X and Y are adjacent.

Theorem 4.1. In a flow shop with m-2 deterministic machines with unit processing times and two stochastic machines (X and Y), under Assumption IJ, one of the following sequences minimizes the expected makespan:

(a) $Y\ I_{m-2}\ X$

(b) $I_{m-2}\ Y\ X$

(c) $I_{m-2}\ X\ Y$

Proof. Let us take a general sequence, $I_kXI_gYI_h$. We will show that the general sequence is stochastically dominated by one of the three sequences.

(a) If $g > 0$, then removing the k+h deterministic machines decreases the makespan at least by k+h units. Since $g > 0$, we know by Lemma 4.1 that placing these k+h machines in between X and Y increases the makespan at most by k+h units. Thus, we obtain $XI_{m-2}Y$.

(b) If $g = 0$, we have I_kXYI_h. Now consider a case where $k > 0$. Removing the last h deterministic machines will reduce the makespan by at least h units. Since $k > 0$, placing them in front will increase the makespan by exactly h units. Thus, we obtain the optimal sequence $I_{m-2}XY$.

(c) If $g = 0$ and $k = 0$, we have the sequence XYI_h. Using an extension of Muth's result (1979), we have the optimal sequence $I_{m-2}YX$. □

Nothing in the proof indicates how the sequences $I_{m-2}XY$, $I_{m-2}YX$ and $YI_{m-2}X$ are related to each other. Thus, we can prove optimality only in expectation, i.e., we cannot guarantee that a stochastically optimal sequence exists. Of course, if such a sequence exists, it must be one of the three mentioned in Theorem 4.1. Even though duality does not hold in general for flow shops with zero intermediate storage space, the results obtained by in Theorem 4.1 are the results that would be obtained from Suresh, Foley and Dickey (1985) if duality held.

Since the result in Theorem 4.1 gives us three possibly optimal sequences, we would like to know the conditions under which we can narrow the field of choices. To accomplish this we need to introduce additional assumptions about the stochastic machines.

<u>Proposition 4.1</u>. If $X_i \geq 1$ and $Y_i \geq 1$ for all i, then $XI_{m-2}Y$ stochastically minimizes the makespan.

<u>Proof</u>. From Theorem 4.1, it is sufficient to show that $XI_{m-2}Y$ is

stochastically smaller than $I_{m-2}XY$ and $I_{m-2}YX$. As in Lemma 4.1, we will consider two systems, A and B. System A consists of two stochastic machines, X and Y. System B is the same as System A except that there is one deterministic machine between the two stochastic machines X and Y. We need to prove the following deterministic relation for System A and System B:

$$D^b_{i,3} - D^a_{i,2} \leq 1 \text{ for all } i. \tag{4.20}$$

Note that this does not follow from Lemma 4.1, since we have no deterministic machines in System A. To prove (4.20), it is sufficient to prove

$$D^b_{i,2} - D^a_{i,1} \leq 1, \tag{4.21}$$

since

$$D^b_{i,3} = D^b_{i,2} + Y_i,$$

and

$$D^a_{i,3} = D^a_{i,2} + Y_i.$$

We will prove (4.21) by induction. For $i = 1$, (4.21) is trivially true. Assume that (4.21) is true for i; we will then show that (4.21) is true for $i+1$. By the induction hypothesis for i, we have

$$D^b_{i,2} - D^a_{i,1} \leq 1. \tag{4.22}$$

For $i+1$ we have

$$D^b_{i+1,2} = \max(D^b_{i,1} + X_{i+1} + 1,\ D^b_{i,2} + Y_i).$$

$$D^b_{i+1,2} \leq D^b_{i,2} + \max(X_{i+1},\ Y_i), \tag{4.23}$$

since

$$D^b_{i,2} - D^b_{i,1} \geq 1.$$

For System A, we have

$$D^a_{i+1,1} = D^a_{i,1} + \max(X_{i+1}, Y_i)$$

and therefore, we have

$$D^b_{i+1,2} - D^a_{i+1,1} \leq 1,$$

which proves (4.21). Thus, by placing an extra machine in between X and Y to obtain XIY, we increase the makespan at most by one. Using Lemma 4.1, we know that placing m-3 machines between XI and Y to obtain $XI_{m-2}Y$ increases the makespan at most by m-3. Thus, the difference in the makespan between XY and $XI_{m-2}Y$ is at most m-2 units. However, adding m-2 machines in front of XY to obtain $I_{m-2}XY$ increases the makespan by at least m-2 units. Thus, $XI_{m-2}Y$ is stochastically smaller than $I_{m-2}XY$. Using the same argument and using an extension of Muth's (1979) reversibility result we can show that $XI_{m-2}Y$ is stochastically smaller than $I_{m-2}YX$. □

A simple condition for $I_{m-2}YX$ to be stochastically optimal is that $Y_i = 0$ for all i. The dual of Proposition 4.1 does not hold. That is, with the assumptions mentioned in the proposition, it can be shown for the dual system that the sequence $XI_{m-2}Y$ does <u>not</u> minimize the expected makespan. As a special case of the results of Foley and Suresh (1984a), we can prove in the dual system that $XI_{m-2}Y$ is stochastically optimal if $X_i \leq 1$ and $Y_i \leq 1$ for all i.

Pinedo (1982b) proves the following related result: for m-2 machines with identical processing time distributions which are nonoverlapping and faster than the other two nonidentical machines, placing one of the slow machines first and the other one last maximizes the throughput. On the basis of Pinedo's (1982b) result, it is reasonable to ask whether Proposition 4.1 can be extended from m-2 identical deterministic machines to m-2 identical stochastic machines that are nonoverlapping and faster than the other two slow machines. Unfortunately, this is not the case, as the following example shows.

Example: We have two jobs and three machines and $S_{i,1} \underset{d}{=} S_{i,2}$ and $S_{i,3} < S_{i,1}$ for all i. So according to our conjecture the machine sequence 1,3,2 should be optimal.

j	1	2
$Pr\{S_{i,1} = j\}$	0.5	0.5

j	0	1
$Pr\{S_{i,3} = j\}$	0.5	0.5

Expected makespan for machine sequence 1,3,2 = 5.375. Expected makespan for machine sequence 1,2,3 = 5.25. Clearly, in this case 1,3,2 is not the optimal sequence.

From the results in this chapter, one may be tempted to conclude that if we have j stochastic machines and n-j identical deterministic machines, the optimal sequence will have the deterministic machines together. However, this may not be true in general. Assume we have 3 stochastic machines and m-3 deterministic machines with almost zero processing times. In this case, it probably best to place half of the m-3 deterministic machines between the first two stochastic machines and to place the remaining deterministic machines between the other two stochastic machines, since the deterministic machines behave like buffer spaces.

CHAPTER 5

SEQUENCING JOBS IN A FLOW SHOP WITH ASSUMPTION IM

In this chapter, we are concerned with scheduling j jobs in a flow shop with m machines and zero intermediate storage between machines. We also use Assumption IM through out this chapter.

Assumption IM. Here we assume that all the machines are identical (IM). specifically, we assume that the random vectors $S_{i,\cdot} = (S_{i,1}, S_{i,2}, \ldots, S_{i,m})$ of processing times for a job i are independent and that the elements within each vector are exchangeable for each i, where $S_{i,j}$ represents the processing time of job i on machine j. This implies that the processing times of a particular job on all machines are identically distributed, but not necessarily independent.

Work done in this area (Pinedo (1982a), Foley and Suresh (1984a, 1984b), and Muth (1979)) has used Assumption IIM (defined in Chapter 2), which requires that all processing times be independent. That is, Assumption IIM is stronger than Assumption IM. All our results use Assumption IM.

In Lemma 5.1, we prove an earlier result stated under stronger assumptions by Pinedo (1982a). Pinedo (1982a) attributed the proof to Muth (1979) and Dattatreya (1978).

Lemma 5.1. In an m machine flow shop with Assumption IM, the total time required to process a given sequence of n jobs has the same distribution as the total time required to process the same n jobs in reverse order.

Proof: We let M_1 be the matrix whose elements are the service times $S_{i,j}$. Thus, for n jobs ordered 1,2,...,n and machines ordered 1,2,...,m, we have

$$M_1 = \begin{bmatrix} S_{1,1} & \cdot & \cdot & \cdot & S_{1,m} \\ \cdot & & \cdot & \cdot & \cdot \\ \cdot & & \cdot & \cdot & \cdot \\ \cdot & & \cdot & \cdot & \cdot \\ S_{n,1} & \cdot & \cdot & \cdot & S_{n,m} \end{bmatrix}$$

The matrix corresponding to job reversal alone (that is, jobs ordered as n, n-1,...,2,1) is

$$M_2 = \begin{bmatrix} S_{n,1} & \cdot & \cdot & \cdot & S_{n,m} \\ \cdot & & \cdot & \cdot & \cdot \\ \cdot & & \cdot & \cdot & \cdot \\ \cdot & & \cdot & \cdot & \cdot \\ S_{1,1} & \cdot & \cdot & \cdot & S_{1,m} \end{bmatrix}$$

The matrix corresponding to job reversal and line reversal (that is, jobs ordered as n, n-1,...,2,1 and machines ordered m, m-1,...,2,1) is

$$M_3 = \begin{bmatrix} S_{n,m} & \cdot & \cdot & \cdot & S_{n,1} \\ \cdot & & \cdot & \cdot & \cdot \\ \cdot & & \cdot & \cdot & \cdot \\ \cdot & & \cdot & \cdot & \cdot \\ S_{1,m} & \cdot & \cdot & \cdot & S_{1,1} \end{bmatrix}$$

The makespan, denoted by C_{max}, as developed by Muth (1979), is a function of the matrix of service times, namely

$$C^i_{max} = f\{M_i\} \text{ for } i = 1,2,3$$

Muth (1979) showed that $C^1_{max} = C^2_{max}$, a deterministic result. It suffices to show

$$M_2 \underset{d}{=} M_3,$$

since this implies

$$C^2_{max} \underset{d}{=} C^3_{max}.$$

Under Assumption IM, the rows are independent and the elements within a row are exchangeable. Thus,

$$M_2 \underset{d}{=} M_3,$$

and this completes the proof. □

Notice that the proof of this lemma can be easily modified to prove the reversibility of machines under Assumption IJ. Under Assumption IJ, the matrix M_2, corresponding to the machine reversal would be

$$M_2 = \begin{bmatrix} S_{1,m} & \cdot & \cdot & \cdot & S_{1,1} \\ \cdot & & \cdot & \cdot & \cdot \\ \cdot & & \cdot & \cdot & \cdot \\ \cdot & & \cdot & \cdot & \cdot \\ S_{n,m} & \cdot & \cdot & \cdot & S_{n,1} \end{bmatrix}$$

We then prove that $M_2 \underset{d}{=} M_3$, arguing along the same lines as in Lemma 5.1. Henceforth we will refer to this lemma as the reversibility result. Many results have appeared in the literature under the assumption that the processing times of job i on the machines are independent and identically distributed random variables. (See Pinedo (1982a, 1982b), Foley and Suresh (1984a, 1984b), Suresh, Foley, Dickey (1985), and Muth (1979).) Now we can prove that all the results are true under this weaker assumption. Shantikumar (1985) pointed out that the result in Foley and Suresh (1984b) is true under this weaker assumption. In fact, since Pinedo (1982a) and Foley and Suresh (1984b) considered a two-machine flow shop, the results are true even if the processing times of a job are allowed to be dependent on the two machines.

Next we analyze an m machine flow shop under Assumption IM with n nonoverlapping jobs and two arbitrary stochastic jobs. Pinedo (1982a) has shown that in an m machine flow shop with n nonoverlapping jobs any SEPT-LEPT sequence minimizes the expected makespan. Foley and Suresh (1984a) extended this result to show that any SEPT-LEPT sequence not only minimizes the expected makespan but also stochastically minimizes the makespan. Suresh, Foley and Dickey (1985) studied a similar system with n identical deterministic jobs and showed that it is best to have the deterministic jobs, which are nonoverlapping, adjacent. On the basis of these results, one might expect that the optimal sequence in a system with n nonoverlapping jobs and two stochastic jobs would have the n nonoverlapping jobs adjacent in a SEPT-LEPT sequence. However, we will show that this is not true in general. Sometimes it pays to break up the SEPT-LEPT sequence.

First, we analyze a system with n nonoverlapping jobs and one stochastic job X. A sequence ARB(k)-X-SEPT(g)-LEPT(p)-ARB(h) represents k nonoverlapping jobs arranged arbitrarily, then stochastic job X, then g nonoverlapping jobs arranged in a SEPT sequence, then p nonoverlapping jobs arranged in a LEPT sequence, then finally h nonoverlapping jobs arranged arbitrarily. M[a] represents the makespan of Sequence a.

We will use Lemmas 5.2 and 5.3 to prove the main result.

Lemma 5.2. In an m machine flow shop under Assumption IM, with n nonoverlapping jobs and one stochastic job,

$$M[ARB(k)\text{-}X\text{-}ARB(g)] \geq M[SEPT(k)\text{-}X\text{-}ARB(g)]$$

where k+g=n and only the order of the first k jobs of the two sequences are allowed to change. We arrange the first k jobs of the sequence ARB(k)-X-ARB(g) in a SEPT sequence, while the order of the last g+1 jobs is the same in both sequences.

Proof. Note that in a SEPT sequence the jobs are not blocked. That is, a job that has finished processing on a machine never waits for the machine ahead of it to be free. Therefore, a SEPT sequence minimizes the departure time of the job k from any machine j. Since the last g+1 jobs are the same in both sequences, we have

$$M[ARB(k)\text{-}X\text{-}ARB(g)] \geq M[SEPT(k)\text{-}X\text{-}ARB(g)]. \qquad \square$$

For Lemma 5.3, we still use the same system. That is, we have an m machine flow shop under Assumption IM, with n nonoverlapping jobs and one stochastic job X.

Lemma 5.3. If in a sequence SEPT(k)-X-LEPT(g) the job before X has a larger processing time than any other job after X, then

$$M[SEPT(k)\text{-}X\text{-}LEPT(g)] \underset{d}{\geq} M[SEPT(k+g)\text{-}X].$$

Proof. By removing the last job from SEPT(k)-X-LEPT(g), we have

$$M[SEPT(k)\text{-}X\text{-}LEPT(g)] - M[SEPT(k)\text{-}X\text{-}LEPT(g-1)] \geq S_{*,m},$$

where $S_{*,m}$ is the processing time of the last job i on the last machine. By placing this job in front of X and maintaining the SEPT configuration, we have

$$M[SEPT(k+1)\text{-}X\text{-}LEPT(g-1)] - M[SEPT(k)\text{-}X\text{-}LEPT(g-1)] = S_{*,1},$$

where $S_{*,1}$ is the processing time of the inserted job on the first machine. Since we assume that the processing times of a job on any machine are exchangeable, we have

$$M[SEPT(k+1)-X-LEPT(g-1)] \underset{d}{\leq} M[SEPT(k)-X-LEPT(g)].$$

Doing this repeatedly, we can prove that

$$M[SEPT(k)-X-LEPT(g)] \underset{d}{\geq} M[SEPT(k+g)-X]. \qquad \square$$

Now we are ready for our main result.

Theorem 5.1. In an m machine flow shop under Assumption IM, with n nonoverlapping jobs and one stochastic job, an optimal sequence for stochastically minimizing the makespan is SEPT(n)-X.

Proof. Take a general sequence ARB(k)-X-ARB(g). Use Lemma 5.3 to obtain SEPT(k)-X-ARB(g), which is lower in makespan. Reverse the sequence to obtain ARB(g)-X-LEPT(k), which according to the reversibility result has stochastically the same makespan. Again, use Lemma 5.3 to obtain SEPT(g)-X-LEPT(k), which is lower in makespan. Without loss of generality, let us assume that the job before X has a higher processing time than the job after X. (If not, reverse the sequence.) Now use Lemma 5.4 to obtain the optimal sequence SEPT(n)-X, which stochastically minimizes the makespan. $\square$

The results are more complicated for two stochastic jobs and n nonoverlapping jobs. Let us represent the two stochastic jobs by X and Y. A sequence ARB(k)-X-SEPT(g)-LEPT(p)-Y-ARB(h) represents k nonoverlapping jobs arranged arbitrarily, then stochastic job X, then g nonoverlapping jobs arranged in a SEPT sequence, then p nonoverlapping jobs arranged in a LEPT sequence, then stochastic job Y, then finally h nonoverlapping jobs arranged arbitrarily. Let $D^a_{i,j}$ denote the departure time of job i from machine j in Sequence a.

We first prove Lemma 5.4, which will be used in our main result. In Lemma 5.4, we have an m machine flow shop under Assumption IM, with n nonoverlapping jobs and two stochastic jobs. We also assume that within the nonoverlapping jobs, the slowest job is in position p.

Lemma 5.4. Consider an m machine flow shop under Assumption IM with n nonoverlapping jobs and two stochastic jobs. Any sequence X-ARB(p+k)-Y can be stochastically improved to some X-SEPT(p)-LEPT(k)-Y; i.e.,

$$M[\text{X-ARB}(p+k)\text{-Y}] \underset{d}{\geq} M[\text{X-SEPT}(p)\text{-LEPT}(k)\text{Y}[$$

where the slowest job in the ARB(p+k) sequence is in position p+1 and remains in position p+1. That is, p-1 nonoverlapping jobs before the slowest job are sorted in SEPT, and k jobs after that slowest job are sorted in LEPT.

Proof. Let the job with the longest processing time among the nonoverlapping jobs be in the position p+1, and let there be k nonoverlapping jobs after this job. The processing time of these k jobs are represented by $S_{i_1,j}, S_{i_2,j}, \ldots, S_{i_k,j}$ on any machine j. The sequence X-ARB(p+k)-Y is denoted by a. Rearranging the k jobs in a LEPT sequence with $S_{1,j} \geq S_{2,j} \geq \ldots \geq S_{k,j}$ for all j, the new sequence X-ARB(p)-LEPT(k)-Y is denoted by b while the first p+1 jobs are the same in both sequences. Two different cases are considered.

Case 1: $m > k$. For Sequence a we have

$$D^a_{p+k+1,j} = D^a_{p+1,k+j}, \text{ for } 1 \leq j \leq c,$$

where c=m-k, since the job p+1 has the highest processing time and the jobs are nonoverlapping. For $j \geqslant c$, we have

$$D^a_{p+1+1,c+12} = D^a_{p+1,m} + \max(S_{i_2,m-1},\ldots,S_{i_k,1})$$

$$= D^a_{p+1,m} + S_{1,?}.$$

We define $S_{1,?}$ as follows: There exists a unique j such that $i_j = 1$ and $S_{1,?} = S_{i_j,m-j+1}$. Here, the departure of the job p+k+1 from machine c+1 machine has to wait until the sum of the departure time of the job p+1 from the last machine and the maximum processing time among the k jobs between jobs p+1 and p+k+1. The slowest of these jobs has a processing time $S_{1,?}$; where the question mark implies that we do not know which machine processed the slowest job. Continuing the argument, we have

$$D^a_{p+k+1,c+j} \geqslant D^a_{p+1,m} + S_{1,?} + S_{2,?} + \ldots + S_{j,?} \text{ for } 1 \leqslant j \leqslant k.$$

Since the first p+1 jobs are the same for both sequences, we have for Sequence b

$$D^b_{p+k+1,j} = D^a_{p+1,k+j}, \text{ for } 1 \leqslant j \leqslant c,$$

$$D^b_{p+k+1,c+j} = D^a_{p+1,m} + S_{1,m} + S_{2,m} + \ldots + S_{j,m} \quad \text{for } 1 \leqslant j \leqslant k.$$

By having $S_{i,m} = S_{1,?}$ for all i and the same job Y in both sequences, we have

$$M[X\text{-}ARB(p+k)\text{-}Y] \underset{d}{\geqslant} M[X\text{-}ARB(p)\text{-}LEPT(k)Y].$$

Case 2: $k \geqslant m$. Let $d=k-m$. $D^a_{p+k-d,j}$ is just a special situation of Case 1 in which c=1. Using the arguments as in Case 1, we have

$$D^a_{p+k+1,j} \geqslant D^a_{p+1,m} + S_{1,?} + S_{2,?} + \ldots + S_{j,?} \text{ for } 1 \leqslant j \leqslant m.$$

Since the first p+1 jobs are the same for both sequences, we have

$$D^{b}_{p+k+1,j} = D^{a}_{p+1,m} + S_{1,m} + S_{2,m} +\ldots+ S_{j,m}) \text{ for } 1 \leq j \leq m.$$

By having $S_{i,m} = S_{i,?}$ for all i and the same job Y in both sequences, we have

$$M[X\text{-}ARB(p+k)\text{-}Y] \underset{d}{\geq} M[X\text{-}ARB(p)\text{-}LEPT(k)Y].$$

Now reverse Sequence b to obtain Y-SEPT(k)-ARB(p)-X, which by the reversibility result has stochastically the same makespan as b. Using the same arguments, we can improve Y-SEPT(k)-ARB(p)-X to a stochastically smaller sequence Y-SEPT(k)-LEPT(p)-X. □

<u>Theorem 5.3</u>. Consider an m machine flow shop with Assumption IM, n nonoverlapping jobs, and two stochastic jobs. The optimal sequence to minimize the expected makespan will be of the form SEPT(k)-X-SEPT(g)-LEPT(n-k-g)-Y or SEPT(k)-Y-SEPT(g)-LEPT(n-k-g)-X, where $k \geq 0$, $g \geq 0$, and the processing times of the jobs between X and Y are greater than those of all the jobs before the first stochastic job.

<u>Proof</u>. Take a general sequence

$$ARB(q)\text{-}X\text{-}ARB(n-k)\text{-}Y\text{-}ARB(k-q).$$

Using the proof similar to Lemma 5.3, we can show

$$M[ARB(q)\text{-}X\text{-}ARB(n-k)\text{-}Y\text{-}ARB(k-q)] \geq M[SEPT(q)\text{-}X\text{-}ARB(n-k)\text{-}Y\text{-}ARB(k-q)].$$

Without loss of generality, we can assume that the job before X has a higher processing time than any job after Y. (If it does not, reverse the sequence.) Using Lemmas 5.3 and 5.4, we can show that

$$M[SEPT(q)\text{-}X\text{-}ARB(n-k)\text{-}Y\text{-}ARB(k-q)] \underset{d}{\geq} M[SEPT(k)\text{-}X\text{-}SEPT(g)\text{-}LEPT(n-k-g)\text{-}Y].$$

All jobs between X and Y have higher processing times than any job before X. If not, we can remove the fast job i, thus reducing the makespan by at least $S_{i,1}$ or $S_{i,m}$. Placing this job i in front of X and maintaining the SEPT configuration, we increase the makespan by exactly $S_{i,1}$. Thus removing the fast job i (which is after X) and placing it in front of X decreases the makespan. □

Nothing in the proof indicates how the different possible optimal sequences of the form SEPT(k)-X-SEPT(g)-LEPT(n-g-k)-Y are related to each other. Thus, we can prove optimality only in expectation.

Now we show that in some sense this theorem cannot be improved. Take two nonoverlapping jobs, N_1 and N_2. Suresh, Foley and Dickey (1985) have shown that when N_1 and N_2 are identical deterministic jobs, there are three possible optimal sequences: XN_1N_2Y, N_1N_2YX and N_1N_2XY. They have examples in which each one is strictly optimal.

Example 5.1. Now let us have the following processing time distribution for N_1, N_2, X, and Y:

i	0.5	2.1
$\Pr\{N_1 = i\}$	0.75	0.25

i	3.9
$\Pr\{N_2 = i\}$	1.00

i	2.0	4.0
$\Pr\{X = i\}$	0.25	0.75

i	2.0	4.0
$\Pr\{Y = i\}$	0.25	0.75

With some algebra we can show that N_1-X-N_2-Y minimizes the expected makespan on two machines yielding a value of 15.875. Any other sequence has an expected makespan of 16.150 or more. Thus, any of the possible optimal sequences can be strictly optimal in special cases, as shown by the above examples along with the examples in Suresh, Foley and Dickey (1985). Foley and Suresh (1986) also give examples in which sequences of the form X-SEPT(k)-LEPT(n-k)-Y are strictly optimal.

With n identical deterministic jobs and 0 stochastic jobs, any sequence is optimal. With one additional stochastic job, I(n)X is stochastically optimal where I(n) denotes the n deterministic jobs. With two stochastic jobs, either XI(n)Y or I(n)XY or I(n)YX is optimal for minimizing the expected makespan.

The general principle that seems to arise is to keep the identical deterministic jobs adjacent. The same general principle appears to arise with nonoverlapping jobs. With no other stochastic jobs, a SEPT sequence is stochastically optimal. With one additional stochastic job, SEPT(n)-X is optimal. The nonoverlapping jobs as a SEPT sequence fit together well: none of the jobs in the SEPT sequence is blocked. However, with two additional stochastic jobs, it may be optimal to break up the SEPT sequence into separate pieces as shown in Example 5.1. Notice that every pair of jobs are overlapping except for N_1 and N_2. Thus, the general principle of keeping nonoverlapping jobs adjacent in a SEPT sequence does not always yield the optimal result.

In Theorem 5.3, we considered a system with n nonoverlapping jobs and two arbitrary stochastic jobs, and we narrowed the set of optimal sequences. In a similar problem, Foley and Suresh (1984a) located an optimal sequence with n identical stochastic jobs and two stochastic jobs, which are nonoverlapping and are faster than the other n jobs. They show that an optimal sequence stochastically minimizing the makespan places one of the fast jobs in front and the other one last with the remaining n jobs in the middle. Sherali (1984) asked to what the optimum sequence would be if the two stochastic jobs were nonoverlapping and slower. Would they remain at the ends, would they be adjacent, or would some other sequences be optimal? In fact, in the next result we not only solve the problem of scheduling n identical stochastic jobs and two arbitrary stochastic jobs that are nonoverlapping and slower, we also solve a more general problem. We consider a case where we have p types of jobs, each nonoverlapping with the other. So within each type, all processing times are identically distributed. First, we prove several lemmas before stating the main result.

For the next lemma, we use the following notation to represent a sequence. In a sequence FAST1(g)-<u>SLOW1(h)</u>-FAST2(k)-<u>SLOW2(r)</u>-ARB(q), FAST1 represents g arbitrary jobs, which are nonoverlapping and faster than the h identical stochastic jobs denoted by SLOW1(h) and the r identical stochastic jobs denoted by SLOW2(r). Similarly, FAST2(k) denotes k arbitrary jobs that are nonoverlapping and faster than SLOW1(h) and SLOW2(r). The h SLOW1 jobs and r SLOW2 jobs need not have any stochastic ordering between them. ARB(q) represents q arbitrary jobs. We could have jobs that are nonoverlapping and slower than SLOW1 or SLOW2 among the last q arbitrary jobs. Lastly, we underline SLOW1(h) to emphasize that the j jobs in SLOW1(h) are identical stochastic jobs. Similarly, we underline SLOW2(r). $D^a_{i,j}$ represents the departure time of a job i from machine j in Sequence a. M[a] represents the makespan of Sequence a. For the system just described, we give the following deterministic result.

<u>Lemma 5.5</u>. In a flow shop with m machines, suppose we have a

sequence

$$\text{a:}\quad \text{FAST1(g)-}\underline{\text{SLOW1(h)}}\text{-FAST2(k)-}\underline{\text{SLOW2(r)}}\text{-ARB(q)}$$

with $r > 0$. Then we have

$$M[a] \geq M[c],$$

where

$$\text{c:}\quad \text{FAST1(g)-FAST2(k)-}\underline{\text{SLOW1(h)}}\text{-}\underline{\text{SLOW2(r)}}\text{-ARB(q)}$$

and the order of jobs within FAST1, FAST2, and ARB remain the same in Sequences A and B.

<u>Proof</u>: Let S1 denote the minimum of all the processing times in SLOW1(h) and let S2 denote the minimum of all the processing times in SLOW2(r). The approach will be to define an intermediate sequence b that is same as Sequence c, except that some of the service times of k arbitrary FAST2 jobs are increased to

$$r = \min(S1, S2).$$

Clearly

$$M[b] \geq M[c].$$

Thus, it will suffice to show that

$$M[a] \geq M[b].$$

We will consider two different cases.

(i) Let $k \geq m$, i.e., the number of arbitrary FAST2 jobs be greater than the number of machines.

The departure time of the first job among the SLOW2(r) jobs from the j^{th} machine is

$$D^a_{g+h+k+1,j} = D^a_{g+h+k+1} + S^z_{1,1} + S^z_{1,2} + \cdots + S^z_{1,j}.$$

$$= D^a_{g,1} + D^a_{g+h,m} - D^a_{g,1} + D^a_{g+h+k,1} - D^a_{g+h,m} + S^z_{1,1} + S^z_{1,2} + \cdots + S^z_{1,j},$$

where $S^z_{1,j}$ denotes the processing time of the first of the SLOW2(r) jobs on machine j. The last equality follows by cancelling terms on the r.h.s.. Note that $D^a_{g+h.m} - D^a_{g,1}$ is the same as the departure time of the last job from the last machine in Sequence d where Sequence d is simply

d: SLOW1(h).

Therefore,

$$D^a_{g+h+k+1,j} = D^a_{g,1} + D^d_{h,m} + D^a_{g+h+k,1}$$

$$- D^a_{g+h,m} + S^z_{1,1} + S^z_{1,2} +\cdots+S^z_{1,1}. \qquad (5.1)$$

Now we want to increase the service times of the k arbitrary FAST2 jobs in Sequence c to form Sequence b. That is,

b: FAST1(g)-FAST2*(k)-SLOW1(h)-SLOW2(r)-ARB(q),

wehre the * denotes that the processing time of FAST2*(k) jobs have been changed. Specifically,

$$S^{f*}_{i,j} = \begin{cases} S^f_{i,j} & 1 \leq j \leq m, \quad m+1-j \leq i \leq k+m+1-j \\ r & \text{otherwise} \end{cases}$$

where $S^f_{i,j}$ is the processing time of job i on machine j in FAST2(k). Clearly,

$$M[b] \geq M[c],$$

since

$$r \geq S^f_{i,j}.$$

For Sequence B, we have

$$D^b_{g+h+k+1,j} = \max(D^b_{g+h+k,1} + S^z_{1,1} + S^z_{1,2} + \cdots + S^z_{1,j};$$

$$D^b_{g+h+k,2} + S^z_{1,2} + \cdots + S^z_{1,j};$$

$$\vdots$$

$$D^b_{g+h+k,j+1}) \text{ for } 1 \leq j \leq m,$$

where $D^b_{g+h+k,j} = 0$ if $j > m$. For a fixed j, define s such that

$$D^b_{g+h+k+1,j} = D^b_{g+h+k,s} + S^z_{1,s} + S^z_{1,s+1} + \cdots + S^z_{1,j}. \tag{5.2}$$

Then,

$$D^b_{g+h+k+1,j} = D^b_{g+h+k,s} + D^b_{g+1,m} - D^b_{g,1} + D^b_{g,1} + D^b_{g+k,1} - D^b_{g+k,1}$$

$$- D^b_{g+1,m} + S^z_{1,2} + S^z_{1,s+1} + \cdots + S^z_{1,j}. \tag{5.3}$$

Again, note that $D^b_{g+h+k,s} - D^a_{g+k,1}$ is the same as the departure time of the last job from machine s in Sequence d. We also note the following relations:

$$D^b_{g+1,m} - D^b_{g,1} = (m-1)r \tag{5.4}$$

$$D^b_{g+k,1} - D^b_{g+1,m} = D^a_{g+h+k,1} - D^a_{g+h,m} \tag{5.5}$$

$$D^b_{g,1} = D^a_{g,1}. \tag{5.6}$$

$$D^d_{h,s} + qr \leq D^d_{h,q+s} \text{ for any } q \leq m-s. \tag{5.7}$$

Therefore, we have

$$D^b_{g+h+k+1,j} = D^b_{g,1} + (m-1)r + D^b_{g+k,1} - D^b_{g+1,m}$$

$$+ D^d_{h,s} + S^z_{1,s} + S^z_{1,s+1} + \cdots + S^z_{1,j}. \tag{5.8}$$

Comparing equations (5.8) and (5.1) and using equations (5.5)-(5.7), we have

$$D^a_{g+h+k+1,j} \geq D^b_{g+h+k+1,j} \text{ for all } 1 \leq j \leq m.$$

Therefore,

$$M[a] \geq M[b].$$

(ii) Let $k \leq m$, i.e., the number of arbitrary FAST2 jobs be less than the number of machines.

Using the same arguments as before, we have

$$\begin{aligned} D^a_{g+h+k+1,j} = D^a_{g,1} + \max(&D^d_{h,k+1} + S^z_{1,1} + S^z_{1,2} + \cdots + S^z_{1,j}; \\ &D^d_{h,k+2} + S^z_{1,2} + S^z_{1,3} + \cdots + S^z_{1,j}; \\ &\quad\vdots \\ &D^d_{h,k+j+1}) \text{ for } 1 \leq j \leq m, \end{aligned} \tag{5.9}$$

where

$$D^d_{h,k+j} = D^d_{h,k+j+1} + S^z_{1,m-k} + S^z_{1,m-k+1} + \cdots + S^z_{1,j} \text{ if } j > m-k.$$

Similar to case (i), create a new sequence

b: FAST1(g)-FAST2*(k)-<u>SLOW1(h)</u>-<u>SLOW2(r)</u>-ARB(q)

where

$$S^{f*}_{i,j} = r \quad 1 \leq j \leq m, \ 1 \leq i \leq k.$$

Clearly,

$$M[b] \geq M[c].$$

Similar to equation (5.3), we have

$$D^b_{g+h+k+1,j} = D^b_{g+h+k,s} + D^b_{g+1,m} - D^b_{g,1} + D^b_{g,1}$$

$$- D^b_{g+1,m} + S^z_{1,s} + S^z_{1,s+1} + \cdots + S^z_{1,j}. \qquad (5.10)$$

In this case we have

$$D^b_{g+1,m} - D^b_{g,1} = kr. \qquad (5.11)$$

Using equations (5.8)-(5.11), we have

$$D^b_{g+h+k+1,j} = D^b_{g,1} + D^d_{h,s} + kr + S^z_{1,s} + S^z_{1,s+1} + \cdots + S^z_{1,j}. \qquad (5.12)$$

$$D^b_{g+h+k+1,j} \leq D^a_{k,1} + D^d_{h,s+k} + S^z_{1,s} + S^z_{1,s+1} + \cdots + S^z_{1,j}. \qquad (5.13)$$

$$D^b_{g+h+k+1,j} \leq D^a_{g+h+k+1,j}$$

Therefore,

$$M[a] \geq M[b] \geq M[c]. \qquad \square$$

Notice that Lemma 5.5 is a deterministic result. The remaining lemmas and theorems are stochastic results. For Lemmas 5.6, 5.7, 5.8, and Theorem 5.4 we assume Hypothesis A.

Hypothesis A. Jobs are divided into p different types, and we have n_t jobs of type t, t = 1,...,p. $S^t_{i,j}$ is the random variable representing the processing time of the job i of type t on machine j. Let S^t be a matrix representing the processing times of the n_t jobs of type t on all machines and S^t_i be a random vector representing the processing times of job i of type t. We assume that

i) $S^t_{i_1,j} \leq S^{t+1}_{i_2,k}$ for all i_1, j, i_2, k

ii) $S^t_1,\ldots,S^t_{n_t}$ are exchangeable random vectors for each t.

Thus jobs of higher types have longer processing times, and jobs within each type are indistinguishable. The processing times of one job does not affect the processing times of other jobs, but the processing time of a job may be dependent on the processing times of the same job on other machines. In particular, Assumption (ii) includes both the case in which the processing times are independent and the case in which the processing times are equal. The strongest assumption, and the assumption that we would most like to relax, is Assumption (i).

Lemma 5.6. In an m machine flow shop using Hypothesis A, we need only consider SEPT-LEPT sequences.

Proof. Take any arbitrary sequence with a total of n jobs. Locate the slowest job in the sequence; let it be in position k. Using Lemma 5.5 repeatedly, we can show the makespan of the arbitrary sequence can be improved when we rearrange the first k jobs in a SEPT sequence. Using the reversibility theorem, we can reverse the sequence. The new sequence has the slowest job in position n-k, with the jobs after the slowest job arranged in a SEPT sequence. Again, using Lemma 5.5 repeatedly, we can decrease the makespan by rearranging the jobs in front of the slowest job in a SEPT sequence. Thus, we have the required SEPT-LEPT sequence. □

For Lemma 5.7, we consider an m machine flow shop with n identical jobs. That is, all the processing times are exchangeable.

Lemma 5.7. In an m machine flow shop with n identical jobs whose processing times are exchangeable, the expected departure time of the job k from the first machine is convex in k.

Proof. To prove the lemma, it suffices to show that

$$D_{k+1,1} - D_{k,1} \underset{d}{\geq} D_{k,1} - D_{k-1,1},$$

where $D_{i,j}$ denotes the departure time of the job i from machine j. Let $S_{i,j}$ denote the processing time of job i on machine j.

It is easy to note that

$$Pr\{D_{k,1} - D_{k-1,1} > t\} = Pr\{DS_{k+1,1} - D_{k,1} > t | S_{1,1}=0, S_{1,2}=0, \ldots, S_{1,m}=0\}$$

since all the processing times are exchangeable. Now fix $S_{2,j}, \ldots, S_{n+1,j}$ for all j. We will show that $D_{k+1,1} - D_{k,1}$ is an increasing function of $S_{1,j}$. Suppose $S_{1,j}$ is increased by a. Then

$$\Delta D_{1,j} = \Delta D_{1,j+1} = \ldots = \Delta D_{1,m} = a$$

where $\Delta D_{i,j}$ represents the change in $D_{i,j}$. It is also easy to see that following inequalities hold:

(i) $\Delta D_{i,j} \leq \Delta D_{i,j+1}$ for all i and $1 \leq j \leq m-1$;

(ii) $\Delta D_{i,j+1} \geq \Delta D_{i+1,j}$ for all i and $1 \leq j \leq m-1$;

(iii) $\Delta D_{i,1} \leq \Delta D_{i+1,1}$ for all i.

Thus,

$$Pr\{D_{k+1,1} - D_{k,1} > t\} \geq Pr\{D_{k,1} - D_{k-1,1} > t\}$$

and therefore $D_{k,1}$ is stochastically convex in k. □

In Lemma 5.8 we give an expression for any SEPT-LEPT sequence in an m machine flow stop using Hypothesis A. Let $M[(k_1, k_2, \ldots, k_{p-1}, n_p, m_{p-1}-k_{p-1}, \ldots, m_1-k_1)]$ denote the makespan of the SEPT-LEPT schedule with k_1 jobs of type 1 followed by k_2 jobs of type 2, and so on. Notice that this is a SEPT-LEPT sequence, that the slowest n_p jobs are together, and that any SEPT-LEPT sequence can be represented in this form.

Lemma 5.8. $M[(k_1,k_2,\ldots,k_{p-1},n_p,n_{p-1}-k_{p-1},\ldots,n_1-k_1)]$ can be represented as

$$D^1_{k_1,1} + D^2_{k_2,1} +\cdots+ D^{p-1}_{k_p-1,1} + M[n_p] + D^{p-1}_{n_{p-1}-k_{p-1},1} +\cdots+ D^1_{n_1-k_1,1},$$

where $D^t_{i,1}$ is the departure time of the job i from the first machine in an m machine flow shop with only type t jobs.

Proof. The k_1 type 1 jobs delay the starting time of the k_2 type 2 jobs by $D^1_{k_1,1}$. Due to Assumption (i), the type 2 jobs will not block type 1 jobs. Thus

$$M[(k_1,k_2,\ldots,k_{p-1},n_p,n_{p-1}-k_{p-1},\ldots,n_1-k_1)]$$

$$= D^1_{k_1,1} + M[(k_2,\ldots,k_{p-1},n_p,n_{p-1}-k_{p-1},\ldots$$

Repeating this procedure p-1 times yields

$$M[(k_1,k_2,\ldots,k_{p-1},n_p,n_{p-1}-k_{p-1},\ldots,n_1-k_1)]$$

$$= D^2_{k_2,1} +\cdots+ D^{p-1}_{k_p-1,1} + M[(n_p,n_{p-1}-k_{p-1},\ldots,n_1-k_1)].$$

Using the reversibility result we have

$$M[(n_p,n_{p-1}-k_{p-1},\ldots,n_1-k_1)] \underset{d}{=} M[(n_1-k_1,\ldots,n_{p-1}-k_{p-1},n_p)].$$

Using this and repeating the first procedure p-1 more times yields the desired result. □

Now we are ready for our main result.

Theorem 5.4. In an m machine flow shop according to Hypothesis A, an optimal schedule is

$$k_t^* = \begin{cases} \frac{n_t}{2} & \text{if n is even} \\ \\ \frac{(n_t+1)}{2} & \text{if n is odd} \end{cases}$$

<u>Proof</u>. From Lemma 5.6, we know we need to consider only SEPT-LEPT sequences. From Lemma 5.8, we know that for an optimal sequence minimizing the expected makespan we need to minimize

$$E[D^t_{k_t,1} + D^t_{n_t-k_t,1}] \text{ for each type t.}$$

Let $g_t(k) = E[D^t_{k,1}]$ and $f_t(k) = g_t(k) + g_t(n_t-k)$. Note that $f_t(k)$ is symmetric about $\frac{n_t}{2}$. If we show that $g_t(k)$ is convex, then we know $f_t(k)$ is convex, which in turn will prove our result. thus, it would suffice to show that $g_t(k)$ is convex in k. But Lemma 5.7 guarantees that $g_t(k)$ is convex. □

Thus, Lemma 5.5 and 5.7 answer the question in Sherali (1984) of finding the optimum sequence with n identical stochastic jobs and two other jobs that are nonoverlapping and slow. Lemma 5.5 proves that the two slow jobs should be together, and Lemma 5.7 proves that the optimum sequence minimizing the expected makespan has the two slow jobs in the middle, half of the identical fast jobs in front, and the remaining fast jobs at the end of the sequence. Theorem 5.4 gives the optimal sequence minimizing only the expected makespan. In special cases it can be shown that this sequence also stochastically minimizes the makespan. For example, in a system with two machines, the sequence in Theorem 5.4 stochastically minimizes the makespan. However, examples can be constructed to show that in general the optimal sequence does not stochastically minimize the makespan. Notice that the result given by Foley and Suresh (1984a) is a special case of Theorem 5.4 The authors show that the optimal sequence stochastically minimizes the makespan.

CHAPTER 6

SCHEDULING JOBS ON TWO PARALLEL MACHINES

6.1 Introduction.

In this chapter we consider scheduling n jobs on two identical parallel machines. The jobs are subjected to precedence constraints, described by a graph that has the form of an intree as shown in Figure 6.1. This implies that each job has at most one successor. The job at the root of the tree is said to be at level 0. A job is said to be at level ℓ if its successor is at level $\ell-1$. We assume that the processing time of the jobs are independent random variables. In this chapter we resolve a conjecture that appeared in Pinedo and Weiss (1985). The authors assume that the jobs at each level ℓ are identically and exponentially distributed random variables, with parameter μ_ℓ and proved that the HLF (Highest Level First) policy minimizes the expected makespan. They also conjecture that HLF policy stochastically minimizes the makespan. We prove that the HLF policy stochastically minimizes the makespan if all jobs at all levels are exponentially distributed with the same parameter μ. Furthermore, we give examples in which HLF does not stochastically minimize the makespan when the jobs have different parameters at different levels. Chandy and Reynolds (1975) proved that the HLF policy minimizes the expected makespan when all the jobs are exponentially distributed with the same parameter. We extend the result to prove that the HLF policy not only minimizes the expected makespan but also stochastically minimizes the makespan.

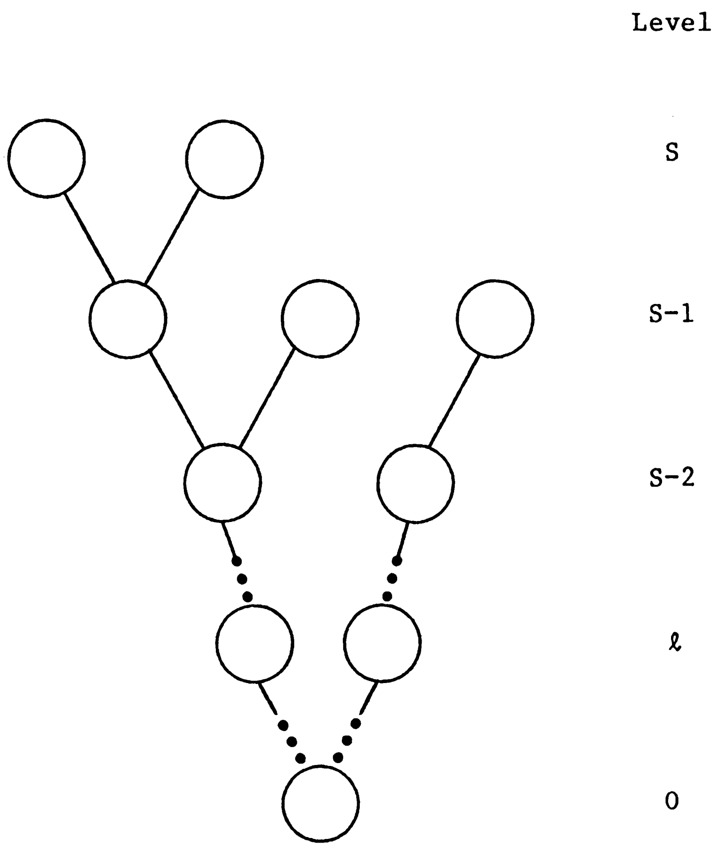

Figure 6.1

Due to the nature of the intree precedence constraints, there will be no idle time until only single jobs are left at levels $1,\ldots,m$; these jobs will be then processed by a single machine. This is true for any schedule. We will use the following notation for the HLF schedule. Let U denote the length of time until one of the machine goes idle. Let J(0) denote the level of the job being processed on the other machine at time U. Let V denote the makespan of the schedule. For any other schedule, we will use U*, V*, and J*(0) for the respective quantities.

We will assume that all the jobs are mutually independent and exponentially distributed with the same parameter. Without loss of generality, we can assume that they are all distributed with a mean of 1.

Theorem 6.1. (Pinedo and Weiss (1985)). For any intree precedence constraint structure with all jobs exponentially distributed, J*(0) is stochastically greater than J(0), i.e.,

$$J^*(0) \underset{d}{\geq} J(0).$$

Proof: See Theorem 1 of Pinedo and Weiss (1985). □

Theorem 6.2. If all jobs are exponentially distributed with mean 1, then the HLF policy stochastically minimizes the makespan.

Proof: It suffices to show that for every m

$$\Pr\{V^* > t \mid J^*(0) = m\} = \Pr\{V > t \mid J(0) = m\} \text{ for any } m,$$

since $\Pr\{V > t \mid J(0)\}$ is an increasing function of J(0) and

$$J^*(0) \underset{d}{\geq} J(0).$$

Let the total number of jobs to be processed be n. The conditional distribution of U given $J(0) = m$ is an Erlang $(n+m-1,2)$, where Erlang (b,μ) is the sum of b independent exponentials with mean $1/\mu$. This is true since both machines are busy until U and the jos are exponentially distributed. The conditional distribution of V-U, given $J(0) = m$, is an Erlang $(m+1,1)$. Since U and V-U are conditionally independent given J(0), the conditional distribution of V given $J(0) = m$ has the same distribution as the sum of Erlang $(n-m+1,2)$ and Erlang $(m+1,1)$.

Using the same arguments, we can show that the conditional distribution of V* given $J^*(0) = m$ has the same distribution as Erlang $(n-m+1,2)$ and Erlang $(m+1,1)$.

Therefore, we have

$$\Pr\{V^* > t\} \geq \Pr\{V > t\},$$

and thus the HLF policy stochastically minimizes the makespan. □

Note the result is true even if the job at level 0 is an arbitrary job with any distribution. These arguments fail if the jobs at each level are exponentially distributed with different parameters, since $\Pr\{V^* > t | J^*(0) = m\}$ may not be equal to $\Pr\{V > t | J(0) = m\}$. This can be seen in the following example.

Example 6.1. Let us consider processing five jobs on two parallel machines with intree precedence constraints, as shown in Figure 6.2. Let $1/\mu_i$ represent the mean for jobs at level i. Let $X(t) = (X_2(t), X_1(t), X_0(t))$ be a stochastic process where $X_i(t)$ is the number of unfinished jobs in level i at time t. Since the jobs within a level are i.i.d. and exponentially distributed, X(t) is a Markov process. The rate diagram for the HLF schedule is shown in Figure

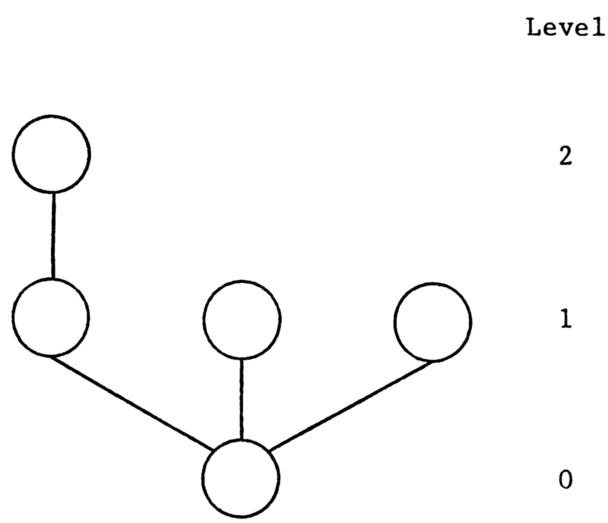

Figure 6.2

6.3. If $X_U = (1,1,1)$ then $J(0) = 2$.

Now consider another schedule where we start processing two jobs at level 1 and then the job at level 2. The rate diagram for this schedule is shown in Figure 6.4. If $X_{U*} = (1,1,1)$ then $J^*(0) = 2$. For the HLF schedule, we have

$$\Pr\{V > t \mid J(0) = 2\} = \Pr\{GE(5\colon \underline{\mu_1+\mu_2}, \mu_1+\mu_2, \mu_2, \mu_1, \mu_0) > t\},$$

where GE(•:•) denotes a generalized Erlang, with the first parameter representing the number of exponentials in the sum and the rest representing the rates of the exponentials. Similarly, for the other schedule, we have

$$\Pr\{V^* > t \mid J^*(0) = 2\} = \Pr\{GE(5\colon \underline{2\mu_1}, \mu_1+\mu_2, \mu_2, \mu_1, \mu_0) > t\}.$$

These two quantities are almost identical except for the underlined parameters in GE(•:•). Thus, $\Pr\{V^* > t \mid J^*(0) = 2\}$ is not equal to $\Pr\{V > t \mid J(0) = 2\}$.

In fact, for this particular example, we can numerically show that

$$\Pr\{V^* > t\} < \Pr\{V > t\} \text{ for some } t,$$

thus disproving the conjecture. With $\mu_1 = 1.0$, $\mu_2 = 0.03$, and $t = 1.0$, we have

$$\Pr\{V^* > t\} = 0.997268 \text{ and } \Pr\{V > t\} = 0.9973405.$$

Again, for the same example except $\mu_2 = 2.001$, and $t = 0.004$ we have

$$\Pr\{V^* > t\} = 0.9999814 \text{ and } \Pr\{V > t\} = 0.9999957.$$

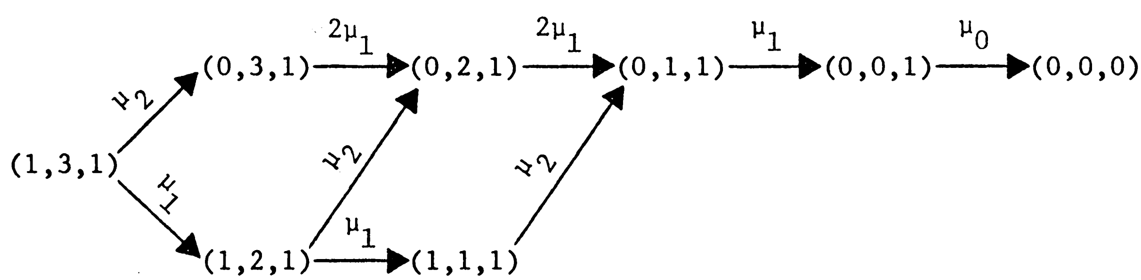
(1,3,1)
(0,3,1)
(0,2,1)
(0,1,1)
(0,0,1)
(0,0,0)
(1,2,1)
(1,1,1)
$2\mu_1$
$2\mu_1$
μ_1
μ_0
μ_2
μ_1
μ_2
μ_1
μ_2

Figure 6.3

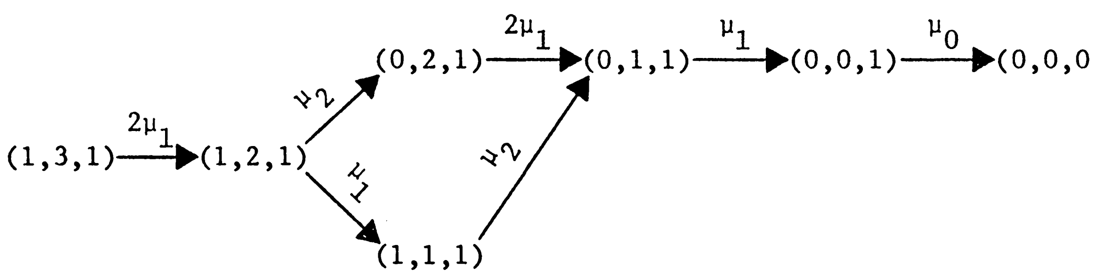

Figure 6.4

CHAPTER 7

CONCLUSIONS AND DIRECTIONS OF FUTURE RESEARCH

In this chapter we draw conclusions from our results. We also provide suggestions for future research.

In Chapter 3 we give results with no assumptions of independence or identical distributions. Consequently, we needed a strong ordering: the nonoverlapping ordering. In this case we can also exactly calculate the expected makespan of the optimal sequence. Tembe and Wolff (1974) consider a stohcastic flow shop with infinite storage and an arbitrary arrival process. They give an optimum sequence, using Assumption IJ and assuming that the machines are nonoverlapping. We assume that all jobs are available at time zero; this is a special case of an arbitrary arrival. Based on the example given in Chapter 3, it seems doubtful that we can relax the nonoverlapping assumption used by Tembe and Wolff (1974).

In Chapter 4, we consider a stochastic flow shop with identical jobs: Assumption IJ. We consider a case with 2 stochastic machines and n deterministic machines. Even though the duality results mentioned in Chapter 2 do not hold true in general, in this cease they are true. That is, the results for this system is the same result obtained by Suresh, Foley and Dickey (1985) for the dual system. It will be interesting to investigate the conditions under which the duality result holds in flow shops with zero intermediate storage. There may be a duality gap between the two systems. Pinedo (1982b) gives the optimal sequence for maximizing the throughput in a stochastic flow shop with zero intermediate storage. He uses Assumption IJ, and assumes n deterministic machines and 2 stochastic machines with identical processing time distributions, with a mean of one. It is still unknown whether this result minimizes the expected makespan.

From the results in Chapter 4, one may be tempted to conclude that if we have j stochastic machines and n-j identical deterministic machines, the optimal sequence will have the deterministic machines together. However, this may not be true in general. Assume that we have 3 stochastic machines and m-3 deterministic machines with almost zero processing time. In this case, it is probably best to place half of the m-3 deterministic machines between the first two stochastic machines and to place the remaining deterministic machines between the other two stochastic machines, since the deterministic machines behave like buffer spaces.

In Chapter 5, we consider the problem of scheduling jobs in an m machine flow shop with zero intermediate storage between machines. We assume that the processing time of a job on all the machines are identically distributed. We extend the reversibility result of Muth (1979) to exchangeable random variables. The reversibility result states that the makespan for any sequence of jobs 1,2,...,n is stochastically the same as the makespan of the reversed sequence n,...,2,1.

We also consider scheduling (n+2) stochastic jobs; n of them are nonoverlapping, and the other two need not have any stochastic ordering on them. Suresh, Foley and Dickey (1985) solve this problem when the n jobs are deterministic. Based on past results, one might expect that the optimum sequence to have the n nonoverlapping jobs arranged in SEPT-LEPT sequence. However, we show that this is not true in general. Sometimes it pays to break up an SEPT-LEPT sequence.

In our last result in Chapter 5, the objective is schedule a group of jobs to minimize the expected makespan. Jobs are divided into p different types, and there are n_t jobs of type t, t = 1,2,...,p. We assume that the processing times of one set of jobs are nonoverlapping with those of any other set. This result generalizes an earlier result by Foley and Suresh (1984a).

In Chapter 6, we consider scheduling n jobs on two parallel machines. The jobs are subject to some precedence constraints. We resolve a conjecture that appeared in Pinedo and Weiss (1985), giving conditions under which the conjecture is true and give examples to prove that the conjecture is false in general. We prove that when all the jobs have exponentially distributed processing time and same mean, then the HLF schedule not only minimizes the expected makespan but stochastically minimizes the makespan. We believe that this result is true if all processing time distributions are identical and DFR (Decreasing Failure Rate).

Throughout this thesis, the problem of computing the expected makespan has not been considered, except in Chapter 3. We need to develop some general procedures to compute, if not the actual expected makespan, at least some bounds for the expected makespan.

APPENDIX

There are many possible ways of establishing a relationship between random variables; these methods are referred to as partial orderings of random variables.

Nonoverlapping smaller

X_i is said to be nonoverlapping smaller than X_j (represented by $X_i < X_j$) if

$$Pr[X_i \leq X_j] = 1.$$

This is an extremely strong ordering because one random variable is always less than the other.

Likelihood Ratio Ordering

We say that X_i is smaller than X_j in likelihood ratio, if

$$\frac{f(x)}{g(x)} \geq \frac{f(y)}{g(y)} \text{ for all } x \leq y,$$

where f and g denote the densities of X_i and X_j respectively. This is denoted by $X_i \underset{L}{\leq} X_j$.

d-ordering

$X_i \underset{d}{\leq} X_j$ if $E[f(X_i)] \leq E[f(X_j)]$ for any f which is a nondecreasing monotone function. We call this a d-ordering; however it has been referred to as type 1 ordering by Stoyan (1977) and stochastic dominance by Bawa (1982). It is shown in the literature that $E[f(X_i)] \leq E[f(X_j)]$ for any nondecreasing monotone function f if and only if

$$Pr[X_i > t] \leq Pr[X_j > t] \text{ for any } t \geq 0.$$

Therefore, as a consequence, we can define $X_i \underset{d}{\leq} X_j$ if

$$Pr[X_i > t] \leq Pr[X_j > t] \text{ for any } t \geq 0.$$

Notice that the complimentary distribution function of X_j will always lie above that of X_i. By choosing $f(t) = t^n$ in $E[f(X_i)] \leq E[f(X_j)]$, we see that all moments of X_i are dominated by corresponding moments of X_j. The d-ordering has a nice economic interpretation. If $X_i \underset{d}{\geq} X_j$, then for any utility function f, which is always monotone nondecreasing, we know that the expected utility of X_i is always less than the expected utility of X_j.

<u>Convex ordering</u>

$X_i \underset{c}{\leq} X_j$ if $E[f(X_i)] \leq E[f(X_j)]$ for any nondecreasing nonnegative convex function f. This is known as c-ordering. It is also equivalent to

$$E[\max(0,X_i-a)] \leq E[\max(0,X_j-a)] \text{ for all } a$$

and to

$$\int_a^\infty (t-a)\ dF(t) \leq \int_a^\infty (t-a)\ dG(t) \text{ for all } a,$$

where F(t) and G(t) are distributions functions of X_i and X_j respectively.

<u>Concave Ordering</u>

$X_i \underset{cc}{\leq} X_j$ if $E[f(X_i)] \underset{cc}{\leq} E[f(X_j)]$ for any nondecreasing nonnegative concave function f. This is known as cc-ordering, It is also equivalent to

$$E[\min(0,x_i-a)] \leq E[\min(0,X_j-a) \text{ for all } a,$$

and to

$$\int_0^a (a-t)\ dF(t) \leq \int_0^a (a-t)\ dG(t) \text{ for all } a,$$

where F(t) and G(t) are distributions functions of X_i and X_j respectively.

Expected value ordering.

X_i is less than X^j in expected value (represented by $E[X_i] \leq E[X_j]$) if

$$\int_0^\infty t \, dF(t) \leq \int_0^\infty t \, dG(t),$$

where F(t) and G(t) are distributions functions of X_i and X_j respectively. The other orderings described earlier are in a sense partial orderings, i.e., it is possible to have

$$\Pr\{X_i > t\} \leq \Pr\{X_j > t\} \text{ for some } t,$$

and

$$\Pr\{X_i > t\} \geq \Pr\{X_j > t\} \text{ for some other } t.$$

On the contrary, the expected value ordering is a total ordering because it is not possible to have $E[X_i] \leq E[X_j]$ and $E[X_j] \leq E[X_i]$ unless $E[X_i] = E[X_j]$.

Relations between Stochastic Orderings

The stochastic orderings discussed here are related to one another. For example, nonoverlapping implies any other ordering, i.e., if $X_i \leq X_j$ then $X_i \leq_d X_j$, $X_i \leq_c X_j$, $X_i \leq_{cc} X_j$, and $E[X_i] \leq E[X_j]$. The following tree has been constructed to show the relation among the stochastic orderings. Here an ordering at the top implies all the orderings below it.

$$
\begin{tikzcd}
& X_i \leq X_j \arrow[d] & \\
& X_i \leq_L X_j \arrow[d] & \\
& X_i \leq_d X_j \arrow[dl] \arrow[dr] & \\
X_i \leq_{cc} X_j \arrow[dr] & & X_i \leq_c X_j \arrow[dl] \\
& E[X_i] \leq E[X_j] &
\end{tikzcd}
$$

It is clear that the makespan is a random variable that depends on the schedule. The processing times of different jobs on different machines are also random variables. We would like our hypothesis to be as weak as possible and our result to be as strong as possible. An excellent result would be one in which the processing times are expected value ordered and the makespan of the optimal sequence is nonoverlapping and smaller than the makespan of any other sequence. Clearly, this is impossible. For example, it is difficult to get a nonoverlapping result for makespan even if we make our assumption about the processing times as strong as that for nonoverlapping.

The work done so far in this area has optimal schedules that minimize only the expected makespan. As righly pointed out in Dempster et al (1981), the expected value criterion is not always a good measure of optimality where a 'once for all' decision must be taken. Sometimes we want an optimality criterion for any expected utility criterion involving a monotone utility function. (As mentioned earlier $E[f(X_i)] \leq E[f(X_j)]$ for any monotone utility function f if and only if $Pr[X_i > t] \leq Pr[X_j > t]$.) The expected value criterion is appropriate for these models, in which relatively small costs or gains are involved per unit time. In the contrary situation, where decisions have to be taken in the face of uncertainties involving relatively large gains or losses, the expected utility criterion is appropriate. Thus, we seek a schedule that stochastically minimizes the makespan. By this we mean that the makespan of the optimal sequence is stochastically less than that of any other sequence. Therefore, $Pr[A^* > t] \leq Pr[A > t]$ where A^* is the makespan of the optimal sequence and A is the makespan of any other sequence. Note that by this optimality criterion the optimal sequence maximizes the probability that a set of jobs will complete processing within a time 't'. Thus, this sequence is optimal not only for minimizing the expected makespan but also for other scheduling criteria such as due date.

REFERENCES

√ Bagga, P. C. (1970), "N-Job, 2-Machine Sequencing Problem with Stochastic Service Times," Opsearch, 7, pp. 184-197.

Baker, K. (1974), Introduction to Sequencing and Scheduling, John Wiley and Sons, Inc., New York.

Bawa, V. (1982), "Stochastic Dominance: A Research Bibliography," Management Sciences, 28, pp. 698-712.

Chandy, K. M., and P. F. Reynolds (1975), "Scheduling Partially Ordered Jobs with Probabilistic Execution Times," Proceedings of the Fifth Symposium on Operating Systems Principles, pp. 169-177.

√ Conway, R., Maxwell, W., and Miller, L. (1967), Theory of Scheduling, Addison-Wesley, Reading, MA.

Dattatreya, E., (1978), "Tandem Queueing Systems with Blocking," Ph.D. dissertation, IEOR Department, University of California, Berkeley.

Dempster, M. A. H., J. K. Lenstra, and A. H. G. Rinooy Kan (1982), Deterministic and Stochastic Scheduling, pp. 7-9, R. Reidel Publishing Company, Boston.

Foley, R. D. and S. Suresh (1984a), "Stochastically Minimizing the Makespan in Flow Shops," Naval Research Logistics Quarterly, 31, pp. 551-558.

Foley, R. D. and S. Suresh (1984b), "Minimizing the Expected Flowtime in Flow Shops," AIIE Transactions, 16, pp. 391-395.

Foley, R. D. and S. Suresh (1986), "Scheduling n Nonoverlapping Jobs and 2 Stochastic Jobs ni a Flow Shop," Naval Research Logistics, 33, pp. 123-128.

Garey, M. and D. S. Johnson (1979), Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman Press, San Francisco, California.

Garey, M. R., D. S. Johnson, R. E. Tarjan and M. Yannakakis (1982), "Scheduling Opposing Forests," SIAM J. Alg. Disc. Meth., 4, pp. 72-93.

Hu, T. C. (1961), "Parallel Sequencing and Assembly Line Problems," Operations Research, 9, pp. 841-848, Johnson, S. M. (1954), "Optimal Two- and Three-Stage Production Schedules with Set up Times Included," Naval Research Logistics Quarterly, 1, pp. 61-68.

Ku, P., and S. Niu (1986), "On Johnson's Two-Machine Flow Shop with Random Processing Times," Operations Research, pp. 130-137.

Lawler, E. L., J. K. Lenstra and A. H. G. Rinnoy Kan (1982), "Recent Developments in Deterministic Sequencing and Scheduling: A Survey," Deterministic and Stochastic Scheduling, pp. 35-74, Dempster et al (eds.), Reidel Dordrecht, The Netherlands.

Muth, J. E. (1979), "The Reversibility Property of Production Lines," Management Science, 25, pp. 152-157.

Pinedo, M. and L. SchragE, (1981), "Stochastic Shop Scheduling: A Survey," Deterministic and Stochastic Scheduling, pp. 181-196, Reidel Publishing Company, Boston.

Pinedo, M. (1982a), "Minimizing the Expected Makespan in Stochastic Flow Shops," Operations Research, 30, pp. 148-162.

Pinedo, M. (1982b), "On the OPtimal Order of Stations in Tandem Queues," Applied Probability-Computer Science: The Interface, Vol. II, (R. L. Disney and T. J. Ott editors), pp. 307-325, Birkhauser Boston, Inc.

Pinedo, M. and G. Weiss, (1985), "Stochastic Jobs with Exponentially Processing times and Intree Precedence Constraints on Two Parallel Machines," Operations Research, pp. 1381-1388.

Reddi, S. S. and C. V. Ramamoorthy (1972), "On the Flowshop Sequencing Problem with no Wait in Process," Operational Res. Quarterly, 23, pp. 323-330.

Shantikumar, G. (1985) Private Communication.

Sherali, H. D. (1984) Private Communication.

Suresh, S. (1984), "Stochastic Flow Shop Scheduling," Master's Thesis, Industrial Engineering and Operations Research Department, Virginia Polytechnic Institute and State University, Blacksburg.

Suresh, S., R. D. Foley and S. E. Dickey (1985), "On Pinedo's Conjecture the Scheduling of Two Stochastic Jobs and n Deterministic Jobs in a Flowshop," Operations Research, 33, pp. 1146-1153.

Stoyan, D. (1977), Qualitative Eigenschaften Abschatzungen stochastischer Modelle, Akacemie-Verlag, Berlin.

Weber, R. R. (1979), "The Interchangeability of Tandem •/M/1 Queues in Series," J. Appl. Probability, 16, pp. 690-695.

Wismer, D. A. (1972), "Solution of Flowshop Schedule with No Intermediate Queues," Operations Research, pp. 689-697.

Tembe, S. V. and R. W. Wolff (1974), "The Optimal Order of Service in Tandem Queues," Operations Research, 22, pp.

The vita has been removed from the scanned document