63/
3.12.05

ON SOME DETERMINISTIC AND STOCHASTIC SCHEDULING PROBLEMS

V. RAJENDRA PRASAD
INDIAN STATISTICAL INSTITUTE
DELHI CENTRE

A THESIS SUBMITTED TO THE INDIAN STATISTICAL INSTITUTE IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

NEW DELHI

1982

ACKNOWLEDGEMENTS

I am greatly indebted to Dr. K.G. Ramamurthy, my thesis supervisor, who introduced me to the theory of scheduling and gave me invaluable suggestions throughout the work. I am thankful to the Council of Scientific and Industrial Research (CSIR), India, for giving me financial assistance. I thank Indian Statistical Institute, Delhi Centre for providing me all the facilities.

I am grateful to the examiners for their helpful comments and suggestions which helped me in strengthening the results. I am thankful to Dr. S.R. Mohan for his valuable advice and full cooperation and to Dr. Thomas Mathew for his cooperation in proof reading the manuscript. I thank my friends and colleagues at Indian Statistical Institute, Delhi Centre, for their encouragement throughout the work.

Last but not least, I thank Mr. V.P. Sharma for quick, excellent and intelligent typing of this thesis and Mr. Munni Lal for duplicating the thesis enthusiastically.

V. RAJENDRA PRASAD

TO

MY PARENTS

C O N T E N T S

CHAPTER I :	INTRODUCTION		1
CHAPTER II :	DETERMINISTIC FLOW SHOP SCHEDULING PROBLEMS		
	2.1	Introduction	12
	2.2	Hybrid Flow Shop Scheduling Problems	13
	2.3	Flow Shops with no Intermediate Storages	37
	2.4	Ordered Flow Shop Scheduling Problems	49
CHAPTER II :	N-JOB 2-MACHINE STOCHASTIC FLOW SHOP SCHEDULING PROBLEMS		
	3.1	Introduction	55
	3.2	The Flow Shop with Geometric Processing times	57
	3.3	The Flow Shop with General Processing times	73
	3.4	Finite Dynamic Programming Formulation	81
CHAPTER IV :	SINGLE-PROCESSOR AND PARALLEL-PROCESSOR STOCHASTIC SCHEDULING PROBLEMS		
	4.1	Introduction	87
	4.2	Non-Identical Parallel-Processor Stochastic Scheduling Problems	89
	4.3	Single-Processor Stochastic Scheduling Problems	120

CHAPTER I
INTRODUCTION

Scheduling problems are quite common in nature. They arise whenever there is a need to plan the execution of various operations over time. Like many other real life problems such as inventories, networks, queues etc. almost all the scheduling problems can be represented by appropriate mathematical models. The theory of scheduling is a discipline which deals with the construction of suitable mathematical models for scheduling problems and their analysis. Scheduling theory came into prominance after Johnson (1954) had published his work on a flow shop scheduling problem.

The current research work in scheduling theory can be classified into two types: (1) on deterministic models and (2) on non-deterministic (stochastic) models. A large number of deterministic models that have been designed to represent various scheduling problems are combinatorial in nature. Unfortunately, the available mathematical tools are not sufficient to deal with such combinatorial models efficiently. For this reason, we presently depend upon branch and bound and heuristic methods, which are not efficient, to solve these models. Some of the scheduling problems can be formulated as standard models like integer programming but the magnitude of such formulations will be quite large.

Recently, researchers like Gittins, Glazebrook, Nash, Pinedo, Weber, Weiss etc. have been formulating the scheduling problems with random processing times as stochastic models. The tools and techniques required for analysing these stochastic models are different from those of deterministic models. Quite often, the theory of semi-Markov decision processes is found useful in analysing the stochastic models which represent the scheduling problems.

In this thesis, we mainly deal with the mathematical aspects of deterministic as well as stochastic scheduling problems . We give below a brief account of the work that is presented in this thesis.

Chapter II deals with deterministic flow shop scheduling problems. In section 2.2 of Chapter II we consider a more general kind of flow shop scheduling problems called hybrid flow-shop scheduling problems in which some of the machines can process simultaneously all the jobs that are to be processed, that is, each one of them can process all the jobs simultaniously. Jackson (1956) has considered a 3-machine, n-job problem of this kind in which the first and the third (last) machines can process only one job at a time whereas the middle one can process all the n jobs simultaneously and he has shown that this problem is equivalent to either of Johnson's (1954) special cases of the $(n/3/F/F_{max})$ problem. The flow-shops with preparatory operations (on the floor) of jobs which do not require any processing machine can be categorised as hybrid flow shops. We consider three types of special cases of the hybrid flow shop scheduling problem with the objective of minimising the total elapsed time (makespan). We deduce a majority of flow shop special cases considered in the literature from these three types of special cases. For hybrid flow shop problems we derive dominance criteria and lower bounds (on makespan) similar to those of $(n/m/F/F_{max})$ problems.

In section 2.3 we consider a flow shop scheduling problem with no intermediate storages (FSNIS). This problem was first considered independently by Wismer (1972) and Reddy and Ramamurthy (1972). Later Panwalker and Wollam (1979) have considered a special case of it assuming that the processing times are ordered and posed a conjecture regarding the minimisation of makespan. For a flow shop problem slightly more general than that of Panwalker and Wollam we obtain two results and prove their conjecture.

In section 2.4, we consider an $(n/m/F/F_{max})$ problem in which the processing times are ordered. This problem has been introduced by Smith et al. (1975). In their experimental investigation on this problem, Panwalker and Khan (1977) have numerically observed that the makespan exhibits a particular property called convex property. Assuming the convex property to hold true, they have presented an efficient algorithm to solve the ordered $(n/m/F/F_{max})$ problem. This convex property of makespan has not been established mathematically so far. We show that the convex property indeed holds true for ordered $(3/3/F/F_{max})$ problems.

In chapter III, we deal with n-job, 2-machine non-deterministic (stochastic) flow shop scheduling problems in which the processing times are independent random variables. Throughout this chapter the objective is to minimise the expected makespan. In section 3.2 we obtain optimality criteria for minimising the expected makespan when the processing times follow geometric distributions. From this, we derive the optimality criteria obtained by Cunningham and Dutta (1973) for the case of exponential distributions. We also give an iterative procedure for obtaining the value of expected makespan for a given permutation schedule.

Section 3.3 deals with general n-job, 2-machine stochastic flowshop problem with the objective of minimising the expected makespan. Optimality criterion for this objective is obtained first in general terms and later in terms of monotone likelihood ratio conditions which are easily verifiable. A number of problems can be solved using MLR optimality criterion. We solve problemsin which

(i) all processing times on first machine follow uniform or normal or gamma distributions,

(ii) all processing times on second machine follow uniform or normal or gamma distributions.

In section 3.4, we consider a problem in which (1) the processing times on first machine follow some known distributions and (2) the processing times on second machine follow exponential distributions (with known parameters) and (3) no passing is allowed, that is, the order of processing is same on both the machines. It is very difficult to solve this problem by the approach adopted in sections 3.2 and 3.3. For this reason, we formulate the above problem as a finite dynamic programming model. This model is numerically tractable (i) when each processing time on first machine follow Erlang distribution with $r = 2$ and (ii) when each processing time on first machine follow either gamma or uniform distribution and no two processing times on second machine are identically distributed.

In Chapter IV, we deal with parallel-processor stochastic scheduling problems. Considerable work has been done on this kind of scheduling problems by researchers like Pinedo, Weiss, Glazebrook, Gittins, Nash, Weber etc. Weiss and Pinedo (1980) have considered a parallel-processor stochastic scheduling problem in which there are 'n' tasks (requiring exponential amount of processing) to be processed on 'm' parallel processors (with different rates of processing). Pre-emptions and switches of a task from one processor to another are allowed. The rate of cost at time t is a function of the set of uncompleted tasks at that time. Formulating this problem as a semi-Markov decision process the authors have obtained optimality criteria for two types of cost functions. They have given seven important applications of these two criteria and conjectured an optimality criteria for another type of cost function. In section 4.2, we briefly describe this work of Weiss and Pinedo (1980) and disprove their conjecture by a numerical counter example. We consider a two-processor (parallel) stochastic scheduling problem in which the processing times of the tasks on one processor, say A, follow identical exponential distributions with parameter μ and on the other processor, say B, they follow different exponential distributions with parameters

λ_j's. We obtain optimal policy that minimises expected flowtime. Later, we consider a problem in which there are 'a' processors of type A and 'b' processors of type B. We obtain optimal policies for four special cases considering the object ives-minimisation of expected flowtime and expected makespan. In section 4.3, we consider a single-processor stochastic scheduling problem in which the processor is subject to failures and repairs. In single-processor scheduling problems, deterministic as well as stochastic, it is generally assumed that the processor is available continuously till the end of processing. But,in practical situations, the processor may break down while operating and require repair for some time. In our problem we assume that the continuous operating time of the processor, that is, the time for which the processor operates continuously and the processing times of tasks are exponential random variables and the repair time is a random variable with finite expectation. Pre-emptions are allowed at repair completion times. The objective of this problem is to minimise the expected weighted sum of task completion times. Formulating this problem as a semi-Markov decision process,we obtain an optimal policy that minimises the objective function.

Definitions and Concepts

We define below the flow shops that are considered in Chapters II and III and provide definitions and concepts of various terms used in those flow shops. For definitions of fundamental terms like schedule, feasible schedule, etc. we refer to Conway et al. (1967) and Baker (1974). In Chapters II and III, we consider only non-delay feasible schedules (the feasible schedules in which no machine remains idle unnecessarily) because the objective is to minimise the makespan and we refer to them simply as schedules. All the scheduling problems can be divided into various classes such as Job-shop, Flow-shop, Parallel-processor ,Single-processor etc .and each class can again be divided into two subclasses (1) deterministic and (2) non-deterministic (stochastic). For details of classification, we refer to Conway et. al (1967), Baker (1974) and Rinnooy Kan (1976). We deal with deterministic flow shop scheduling problems in Chapter II, non-deterministic flow shop scheduling problems in Chapter III and parallel processors and single processor non-deterministic scheduling problems in Chapter IV.

In Chapter II we consider three types of deterministic flow shops. The first type is simple called flow-shop and the second and third types are called flow shop with no intermediate storages and hybrid flow shop respectively.

I Flow-shop : We define the flow shop through the following assumptions.

A1 : There are 'n' jobs 1,2,...,n to be processed on 'm' machines $M_1, M_2, \ldots, M_m$.

A2 : Job i, $1 \leq i \leq n$, is available from time $\alpha_i (\geq 0)$ onwards and machine M_j, $1 \leq j \leq m$, is available from time $\beta_j (\geq 0)$ onwards.

A3 : Job i, $1 \leq i \leq n$, requires an amount of time p_{ij} on machine M_j, $1 \leq j \leq n$, for set-up and processing together (p_{ij} is called processing time of job i on machine M_j).

A4 : All p_{ij}'s are known and fixed.

A5 : Each job is processed on the machines in the order $(M_1, M_2, \ldots, M_m)$.

A6 : Interruption of processing of any job on any machine is not allowed, that is, once a machine M_j starts processing a job i it has to process continuously till the processing requirement of job j on it is met.

A7 : Each machine can process only one job at a time.

A8 : No job can be processed on more than one machine simultaniously.

In our flow shops we always assume that $\alpha_i = 0$ for $i = 1$ to n and $\beta_j = 0$ for $j = 1$ to m, i.e., all the n jobs and m machines are available at time $t = 0$.

The matrix $P = ((p_{ij}))_{n \times m}$ is called <u>processing time matrix</u> of the flow shop. p_{ij}'s are called <u>ordered processing times</u> when

(1) for any $r,s (1 \leq r,s \leq m)$ $p_{ir} \geq p_{is}$ for $i = 1$ to n if there exists a u, $1 \leq u \leq n$, such that $p_{ur} > p_{us}$ and

(2) for any k,ℓ $(1 \leq k,\ell \leq n)$, $p_{kj} \geq p_{\ell j}$ for $j = 1$ to m if there exists a v, $1 \leq v \leq m$, such that $p_{kv} > p_{\ell v}$.

When the processing times are ordered ones, the jobs are generally renumbered such that $p_{1j} \leq p_{2j} \leq \cdots \leq p_{nj}$ for $1 \leq j \leq m$ and the job n (with largest processing time on each machine) is called the <u>largest job</u>. We represent the index of the largest column of the matrix of ordered processing times by s.

Let c_i, $1 \leq i \leq n$, represent the completion time of job i on the last machine M_m. The objectives considered in general in flow shops are

(1) minimisation of maximum job completion time : $F_{max} = \max\limits_{1 \leq i \leq n} c_i$

(2) minimisation of flow time : $\bar{F} = \sum\limits_{i=1}^{n} (c_i - \alpha_i)$

(3) minimisation of weighted flow time : $\bar{F}_w = \sum\limits_{i=1}^{n} w_i(c_i - \alpha_i)$

(4) minimisation of maximum tardiness : $T = \max\limits_{1 \leq i \leq n} \{\max(0, c_i - d_i)\}$

where d_i is the due date of job i, and

(5) minimisation of number of late jobs : $L = \sum\limits_{i=1}^{n} k_i$

where

$$k_i = \begin{cases} 0 & \text{if} \quad c_i \le d_i \\ 1 & \text{otherwise} \end{cases} .$$

The problem of scheduling the entire processing of a flow shop in order to meet a give objective is called flow shop scheduling problem. Throughout Chapter II, our objective is to minimise the maximum job completion time (makespan). The n-job, m-machine flow shop scheduling problem with this objective is denoted as $(n/m/F/F_{max})$. Garey, et al. (1976) have shown that the $(n/m/F/F_{max})$ problem is NP-complete for $m \ge 3$. For reference to NP-complete, see Garey et al. (1979). So far, it is not known whether there exists a polynomially bounded algorithm for a problem which is NP-complete. However, it is strongly believed by several authors that there does not exist such an algorithm for an NP-complete problem.

In an n-job, m-machine flow shop scheduling problem, a schedule can be represented by a sequence of m permutations of 1,2,...,n. Under a schedule $f = (\pi^{(1)}, \pi^{(2)}, \ldots, \pi^{(m)})$ where $\pi^{(j)} = (\pi_1^{(j)}, \pi_2^{(j)}, \ldots, \pi_n^{(j)})$ for $1 \le j \le m$, we process the jobs on machine M_j, $1 \le j \le m$, in the order $(\pi_1^{(j)}, \pi_2^{(j)}, \ldots, \pi_n^{(j)})$. If $\pi^{(j)} = \eta$ for $1 \le j \le m$ where η is some permutation of 1,2,...,n, the schedule f is called permutation schedule and it can be simply represented by the permutation η. We refer to permutation schedules as sequences or permutations. A permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ is called pyramid permutation when $\pi_1 < \pi_2 < \ldots < \pi_r > \pi_{r+1} > \ldots > \pi_n$ for some $r, 1 \le r \le n$. A permutation $\pi = (\pi_n, \pi_{n-1}, \ldots, \pi_2, \pi_1)$ is said to be reverse permutation of $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$.

II Flow-shop with no intermediate storages : Considerthe assumption that jobs are not allowed to wait for processing before any machine except the first one (M_1), that is, once a job is processed on a machine $M_j (1 \le j < m)$, it must be immediately taken up for processing on the next machine M_{j+1}. The flow shop with the above assumption is called flow shop with no intermediate storages and denoted as FSNIS. In this flow shop (with no intermediate storages) also we can consider the same objectives as in the flow-shop. The problem of scheduling the entire processing of an FSNIS in order to meet a given objective is called flow shop scheduling problem with no intermediate storages and denoted as FSNIS problem. In FSNIS, the permutation schedules are the only feasible schedules when all the processing times are non zero.

III Hybrid flow-shop : The flow shop with the assumption A7 replaced by an assumption that some machines in the set $\{M_1, M_2, \ldots, M_m\}$ can process all the n jobs simultaniously is called hybrid flow shop. In a shop consisting of n jobs that are to be processed, a machine which can process all the n jobs simultaniously is called non-bottleneck machine. A machine which can process only one job at a time is called bottleneck machine.

For hybrid flow shops the objectives are same as those of the flow shop. In Chapter II, we consider only the objective of minimising the makespan for hybrid flow shops. The problem of scheduling the entire processing of a hybrid flow shop in order to meet a given objective is called hybrid flowshop scheduling problem.

In Chapter III, we consider n-job, 2-machine stochastic flow shop scheduling problems. Stochastic flow shop is a flow shop with the assumption A_4 replaced by an assumption that the processing times are independent random variables. In stochastic flow shops, our objective is to minimise the expected makespan.

Schedules of general (deterministic and stochastic) flow shop scheduling problems :

In general flow shop scheduling problems, a schedule means fixing an order of processing on each machine initially at time 0. We now introduce a term 'policy' the concept of which is more general than that of schedule.

Let the time 0 and job completion times on machines be denoted as decision moments. A policy of a general flow shop scheduling problem is a rule that determines which job is to be taken for processing on first machine at first decision moment (time 0) and determines which job is to be taken for processing on the machine that is just free at each decision moment later depending upon the state of the shop (the state of processing) at that moment. Below, we mathematically define these policies for n-job, 2-machine flow shop problems.

Let $N = \{1,2,\dots,n\}$ and $\Omega = (0,0) \cup N \times R^+$ where $R^+ = (0,\infty)$. Consider a one-one mapping $h[E,(k,x);\ F,(\ell,y)] = (h_1,h_2)$ from $2^N \times \Omega \times 2^N \times \Omega$ to $N' \times N'$ where $N' = \{0,1,2,\dots,n\}$ such that

(a) when $(k,x) \neq (0,0)$

$$h[E,(k,x);\ F,(\ell,y)] = \begin{cases} (k,\ell) \text{ for any } F \text{ if } (\ell,y) \neq (0,0) \\ (k,j), j\in F, \text{ if } F \neq \phi,\ (\ell,y) = (0,0) \\ (k,0) \text{ if } F = \phi,\ (\ell,y) = (0,0) \end{cases}$$

(b) when $E \neq \phi$, $(k,x) = (0,0)$

$$h[E,(k,x);\ F,(\ell,y)] = \begin{cases} (i,\ell),\ i\in E \text{ for any } F \text{ if } (\ell,y) \neq (0,0) \\ (i,j), i\in E, j\in F, \text{ if } F \neq \phi, (\ell,y) = (0,0) \\ (i,0),\ i\in E, \text{ if } F = \phi,\ (\ell,y) = (0,0) \end{cases}$$

(c) when $E = \phi, (k,x) = (0,0)$

$$h[E,(k,x);F,(\ell,y)] = \begin{cases} (0,\ell) \text{ for any } F \text{ if } (\ell,y) \neq (0,0) \\ (0,j), j\in F, \text{ if } F \neq \phi,\ (\ell,y) = (0,0) \\ (0,0) \text{ if } F = \phi,\ (\ell,y) = (0,0) \end{cases}$$

We interpret below the mapping 'h' as a rule for assigning the jobs to the two machines.

We can represent the state of the 2-machine flow shop at time t by $[E,(k,x);F,(\ell,y)]$ where E is the set of jobs that are waiting for processing before first machine at time t and k is the job that has been processed for time x and is still being processed on first machine at time t and the same discription holds for F and (ℓ,y) on second machine. At each job completion time on first (second) machine $(k,x) = (0,0)((\ell,y) = (0,0))$. Thus the set $2^N \times \Omega \times 2^N \times \Omega$ represents the state space (the set of all states) of the shop. Interpreting h as the job assigned to first machine and h_2 as the job assigned to second machine (and 0 as assignment of no job) we can follow the mapping h as a rule for assigning the jobs to the machines in the two machine flow shop. Let

$$H = \{h/h \text{ is a one-one mapping from } 2^N \times \Omega \times 2^N \times \Omega \text{ to } N' \times N' \text{ satisfying the conditions (a), (b) and (c)}\}$$

A sequence $f = (f_0, f_1, f_2, f_3, \ldots, f_{2n-1})$ of 2n mappings belonging to H is called a policy. The policy f is said to be followed for scheduling the processing when the mapping f_i, $0 \leq i \leq 2n-1$, is followed as a rule for assigning jobs at ith decision moment.

Remarks 1.2.1 : The policies are defined in such a way that under any policy no machine is idle as long as there is at least one job waiting before it. In a given general flow shop scheduling problem, for any schedule we can find a policy such that the course of processing is same under both the policy and the schedule. Thus, the class of policies is wider than the class of schedules. In deterministic flow shops, the state of the shop and the action (assignment of jobs) to be taken at any decision moment can be determined at time 0 for any policy and therefore each policy corresponds to a schedule. But in stochastic case it is not so since the processing times are random variables.

CHAPTER II

DETERMINISTIC FLOW SHOP SCHEDULING PROBLEMS

2.1 Introduction

In this chapter, we deal with deterministic flow shop scheduling problems in which the objective is to minimise the makespan. These problems, denoted as $(n/m/F/F_{max})$, have drawn the attention of a large number of researchers since Johnson (1954) solved the $(n/2/F/F_{max})$ problem and two special cases of the $(n/3/F/F_{max})$ problem. Garey, et. al. (1976) have shown that the $(n/m/F/F_{max})$ problem is NP-complete for $m \geq 3$. In the literature, there are mainly three ways to deal with the $(n/m/F/F_{max})$ problem, (1) developing heuristic rules for obtaining a near optimal solution, (2) developing branch and bound procedures for obtaining an exact optimal solution and (3) considering solvable special cases (special cases which can be solved by a polynomially bounded algorithm). For reference to various heuristic rules, see Giglio and Wagner (1964), Palmer (1965), Mc Mahon and Burton (1967), Campbell, et. al (1970), Ashour (1970), Dannenbring (1977) and Achuthan (1980). For the $(n/3/F/F_{max})$ problem, Achuthan (1980) has compared the efficiencies of all these heuristics computationally. For branch and bound procedures, we refer to Lomnicki (1965), Ignall and Schrage (1965), Brown and Lomnicki (1966). The solvable special cases of $(n/m/F/F_{max})$ problem have been considered by a large number of researchers. In these special cases, the processing times exhibit some kind of relationship which is exploited for developing a polynomially bounded algorithm.

From a practical point of view, there is no doubt that development of algorithms is more important than consideration of special cases.

Nevertheless, one should not look down upon solvable special cases for three reasons,(1) when these special cases arise in practical situations, the problem can be solved very efficiently, (2) they are sometimes useful in developing algorithms like heuristics and branch and bound procedures and (3) the study of special cases enhances the understanding of the problem.

We give below a brief account of the work done in this chapter. In section 2.2,we consider hybrid flow shop scheduling problems with the objective of minimising the makespan. We deduce a large number of special cases of the $(n/m/F/F_{max})$ problem from these problems . In section 2.3,we consider a flow shop scheduling problem with no intermediate storages (FSNIS), which is slightly more general than ordered FSNIS problem. We prove a conjecture (posed by Panwalker and Wollam (1979) for OFSNIS problem) and further obtain two more results. In section 3.4, we consider ordered flow shop scheduling problem and prove a conjecture (posed by Panwalker and Khan (1977)) for 3 x 3 case.

2.2 Hybrid Flow Shop Scheduling Problems

Introduction

In this section, we consider hybrid flow shop scheduling problems with the objective of minimising the makespan. First, we consider three types of special cases of a hybrid flow shop problem in which the bottleneck and non-bottleneck machines are at alternate stages. We deduce a majority of special cases of the $(n/m/F/F_{max})$ problem (considred in the literature) as subcases of these three types of hybrid flow shop problems. Next, we derive dominance criteria (with regard to bottleneck machines) and lower bounds similar to those of the $(n/m/F/F_{max})$ problem. We finally show that the hybrid flow shop problem considered at the beginning of the section represent the entire class of hybrid flow shop scheduling problems with the objective of minimising the makespan.

<u>Description and Notation</u> :

First, we consider a hybrid flow shop scheduling problem H in which

(i) the number of machines is 2m-1, $m \geq 1$.

(ii) the machines at odd numbered stages (1,3,5,...,2m-1) are all bottleck machines.

(iii) the machines at even numbered stages (2,4,6,...,2m-2) are all non-bottleneck machines.

For this problem, we use the following notation. Let $N = \{1,2,\ldots,n\}$ be the set of n jobs to be processed and $M_1, M_2, \ldots, M_m$ the bottleneck machin at stages 1,3,.....,(2m-1) and $W_1, W_2, \ldots, W_{m-1}$ the non-bottleneck machines at stages 2,4,.....,2(m-1) respectively. Let p_{ij}, $1 \leq i \leq n$ and $1 \leq j \leq m$, be the processing time of job i on machine M_j and q_{ij}, $1 \leq i \leq n$ and $1 \leq j \leq m-1$, the processing time of job i on machine W_j. We can represent the processing time matrix $\bar{P}$ of this problem as

$$\bar{P} = (P_1 Q_1 P_2 Q_2 \ldots P_{m-1} Q_{m-1} P_m)$$

where $P_j^T = (p_{1j}, p_{2j}, \ldots, p_{nj})$ for $1 \leq j \leq m$ and $Q_j^T = (q_{1j}, q_{2j}, \ldots, q_{nj})$ for $1 \leq j \leq m-1$. We denote this hybrid flow shop problem H by $(n/2m-1/F/\bar{F}_{max})$.

As in the case of $(n/m/F/F_{max})$ problem, we restrict our attention to the set of permutation schedules because it is very difficult to deal with non-permutation schedules. The makespan for a permutation (permutation schedule) $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ of the problem H is given by

$$\bar{T}(\pi) = \max_{1 \le v_1 \le v_2 \le \cdots \le v_{n-1} \le m} \Big[\sum_{j=1}^{v_1} p_{\pi_1 j} + \sum_{j=1}^{v_1 - 1} q_{\pi_1 j} + \sum_{j=v_1}^{v_2} p_{\pi_2 j} + \sum_{j=v_1}^{v_2 - 1} q_{\pi_2 j} \qquad (2.2.1)$$

$$+ \cdots + \sum_{j=v_{n-1}}^{m} p_{\pi_n j} + \sum_{j=v_{n-1}}^{m-1} q_{\pi_n j} \Big].$$

We can easily verify this by mathematical induction on n and m.

For any partial permutation $(i_1, i_2, \ldots, i_k)$ of $1, 2, \ldots, n$ and any u and v $1 \le u \le v \le m$, let

$$C((i_1, i_2, \ldots, i_k); u, v) = \max_{u \le s_1 \le s_2 \le \cdots \le s_{k-1} \le v} \Big\{ \sum_{j=u}^{s_1} p_{i_1 j} + \sum_{j=u}^{s_1 - 1} q_{i_1 j} + \sum_{j=s_1}^{s_2} p_{i_2 j}$$

$$+ \sum_{j=s_1}^{s_2 - 1} q_{i_2 j} + \cdots + \sum_{j=s_{k-1}}^{v} p_{i_k j} + \sum_{j=s_{k-1}}^{v-1} q_{i_k j} \Big\}.$$

Then, we can write for any arbitrary permutation π

$$\bar{T}(\pi) = \max_{1 \le u_1 \le u_2 \le \cdots \le u_{r-1} \le m} [\bar{T}(\pi^{(1)}, u_1) + C(\pi^{(2)}; u_1, u_2) + C(\pi^{(3)}; u_2, u_3)$$

$$+ \cdots + C(\pi^{(r)}; u_{r-1}, m)]$$

where $\bar{T}(\pi^{(1)}, u_1) = C(\pi^{(1)}; 1, u_1)$ and $\pi^{(1)} \pi^{(2)} \pi^{(3)} \cdots \pi^{(r)} = \pi$.

The hybrid flow shop problem H^* with the same set of jobs as above and machine order $M_m\ W_{m-1}\ M_{m-1} \cdots W_2\ M_2\ W_1\ M_1$ is called the <u>reverse problem</u> of H. Note that $\bar{T}(\pi) = \bar{T}^*(\pi^*)$ for any permutation π where π^* is the reverse permutation of π and $\bar{T}^*(\pi^*)$ is the makespan for the permutation π^* of the problem H^*. The processing time matrix of the problem H^* is $\bar{P}^* = (P_m\ Q_{m-1}\ P_{m-1} \cdots P_2 Q_1 P_1)$. We call $\bar{P}^*$ the reverse matrix of $\bar{P}$.

In the $(n/m/F/F_{max})$ problem, a path of the processing time matrix represents, according to Szwarc (1978), a sequence of (m+n-1) positions starting from (1,1) and ending with (n,m) such that each position (r,s) of the sequence is followed by either (r+1, s) or (r,s+1). For the hybrid flow shop problem H, we consider a subset (Γ) of paths of $\bar{P}$ in which each path contains only one position of each of the (m-1) columns $Q_1, Q_2, \ldots, Q_{m-1}$. We represent the set Γ mathematically as

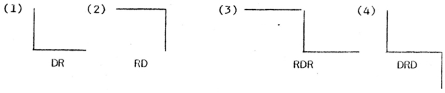
$$\Gamma = \{(\sigma_1(1,v_1), \sigma_2(v_1,v_2), \ldots, \sigma_n(v_{n-1},m)) / 1 \le v_1 \le v_2 \le \cdots \le v_{n-1} \le m\}$$

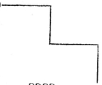
where $\sigma_i(k,\ell) = ((i,k),(i,k'),\ (i,k+1),\ (i,\overline{k+1}')\ldots(i,\ell-1),(i,\overline{\ell-1}'),(i,\ell))$ for $1 \le i \le n$ and $1 \le k \le \ell \le m$ and (i,j') is the i th position of column Q_j.

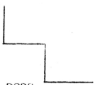
For a permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ of $1,2,\ldots,n$ and a path τ of $\bar{P}$ let $s(\pi,\tau)$ represent the sum of elements on the path τ of the matrix $\pi\bar{P}$ where $\pi\bar{P}$ is the matrix obtained from $\bar{P}$ by arranging the rows of $\bar{P}$ in the order $(\pi_1, \pi_2, \ldots, \pi_n)$. It is easy to see that for a fixed τ, the problem of minimising $s(\pi,\tau)$ over the set of permutations is an assignment problem. We can write

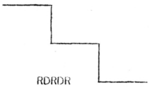
$$\bar{T}(\pi) = \max_{\tau \in \Gamma} s(\pi,\tau)$$

If $\bar{T}(\pi) = s(\pi,\tau^*)$ for a path $\tau^* \in \Gamma$, then the path τ^* is called a critical path. For each special case of the problem H considered later, the critical path takes one of the following seven shapes (given by Szwarc (1978)).



(5)

RDRD

(6)

DRDR

(7)

RDRDR

The letters R and D denote the horizontal and vertical segments.

Special Cases

Garey, et. al. (1976) have shown that the general $(n/m/F/F_{max})$ problem which is a particular case of the problem H is NP-**complete**. This means that it is very unlikely to develop a polynomially bounded algorithm for solving the problem H. In this kind of situation, the attention will generally be focussed on finding efficient lower bounds which can be used in branch and bound procedure for solving the problem. Quite often in real life, the processing times of the flow shop exhibit some kind of explicit relationship among themselves. Sometimes, we can easily solve the problem exploiting this relationship without using any enumeration procedure which involves a lot of computational work. In this section, we consider various kinds of relationship among the processing times of H and explore the possibilities of solving the problem under these relationships.

We first introduce two classes $\bar{A}$ and $\bar{B}$ of matrices(consisting of odd number of columns) which are helpful in obtaining various special cases.

Definition of $\bar{A}$ and $\bar{B}$ Classes : A matrix $\bar{P}$ (as described above) is said to belong to class $\bar{A}(\bar{B})$ if

$$\bar{T}(\pi) = \sum_{i=1}^{n} p_{\pi_i 1} + \sum_{j=2}^{m} (p_{\pi_n j} + q_{\pi_n j-1})\left(\sum_{j=1}^{m-1} (p_{\pi_1 j} + q_{\pi_1 j}) + \sum_{i=1}^{n} p_{\pi_i m}\right) \tag{2.2.3}$$

for any permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$.

The expression on the right hand side of the above equation can be replaced by $s(\pi,\tau_{n-1})(s(\pi,\tau^*_{n-1}))$ where $\tau_{n-1} = (\sigma_1(1,1), \sigma_2(1,1), \ldots, \sigma_{n-1}(1,1), \sigma_n(1,m))$ and $\tau^*_{n-1} = (\sigma_1(1,m), \sigma_2(m,m), \sigma_3(m,m), \ldots, \sigma_n(m,m))$.

Remark 2.2.1 : The reverse matrix $\bar{P}^* = (P_m\ Q_{m-1}\ P_{m-1} \cdots P_2\ Q_1\ P_1)$ of $\bar{P}$ belongs to $\bar{B}(\bar{A})$ iff $\bar{P}$ belongs to $\bar{A}(\bar{B})$. A matrix of single column can be considered to belong to $\bar{A}$ and $\bar{B}$.

We now give necessary and sufficient conditions for $\bar{P}$ to belong to $\bar{A}$ and $\bar{B}$ classes.

Theorem 2.2.2 : The necessary and sufficient conditions for $\bar{P}$ to belong to $\bar{A}$ are

$$\sum_{k=1}^{v} (p_{ik} + q_{ik}) \geq \sum_{k=1}^{v} (p_{jk+1} + q_{jk}) \tag{2.2.4}$$

for any i,j, $1 \leq i,j \leq n$, $i \neq j$ and $1 \leq v \leq m-1$.

Proof : First, we show that the inequalities (2.2.4) $\Rightarrow \bar{P} \in \bar{A}$

Consider an arbitrary path $\tau = (\sigma_1(1,v_1), \sigma_2(v_1,v_2), \ldots, \sigma_n(v_{n-1},m))$ and an arbitrary permutation π of $1,2,\ldots,n$. We can easily see that $s(\pi,\tau) \leq s(\pi,\tau_1)$ by inequalities (2.2.4), where $\tau_1 = \{\sigma_1(1,1), \sigma_2(1,v_2), \sigma_3(v_2,v_3) \ldots \sigma_n(v_{n-1},m)\}$. Similarly $s(\pi,\tau_1) \leq s(\pi,\tau_2)$ by inequalities (2.2.4) where $\tau_2 = (\sigma_1(1,1), \sigma_2(1,1), \sigma_3(1,v_3), \sigma_4(v_3,v_4) \ldots \sigma_n(v_{n-1},m))$ i.e., $s(\pi,\tau) \leq s(\pi,\tau_2)$.

Repeating this argument successively, we finally get

$$s(\pi,\tau) \leq s(\pi,\tau_{n-1})$$

where $\tau_{n-1} = (\sigma_1(1,1), \sigma_2(1,1), \ldots, \sigma_{n-1}(1,1), \sigma_n(1,m))$. Since τ and π

are arbitrary, it now follows from the above inequality that $\bar{T}(\pi) = s(\pi,\tau_{n-1})$ for any permutation π. Therefore $\bar{P} \in \bar{A}$.

By comparing $\bar{T}(\pi)$ with $s(\pi,\eta)$, where $\eta = (\sigma_1(1,1),\sigma_2(1,1),\ldots,\sigma_{n-2}(1,1),$ $\sigma_{n-1}(1,v),\ \sigma_n(v,m))$, $1 < v \leq m$, for each permutation π, we can show that $\bar{P} \in \bar{A} \Rightarrow$ the inequalities (2.2.4).

Corollary 2.2.3 : The necessary and sufficient conditions for $\bar{P}$ to belong to the $\bar{B}$ are

$$\sum_{k=v}^{m-1} (p_{ik+1}+q_{ik}) \geq \sum_{k=v}^{m-1} (p_{jk} + q_{jk}) \qquad (2.2.5)$$

for any i,j, $1 \leq i,j \leq n$, $i \neq j$ and $1 \leq v \leq m-1$

Proof : One can easily see by Theorem 2.2.2 that the inequalities (2.2.5) $\iff$ the necessary and sufficient conditions for $\bar{P}^*$ to belong to $\bar{A}$ $\iff \bar{P}^* \in \bar{A} \iff \bar{P} \in \bar{B}$. Therefore the inequalities (2.2.5) are necessary and sufficient conditions for $\bar{P}$ to belong to the class $\bar{B}$. Note that the conditions (2.2.4) and (2.2.5) are generalisations of Forward and Backward Dominance conditions introduced by Arthanari (1974) for the $(n/m/F/F_{max})$ problem.

Corollary 2.2.4 : Let the matrix $\bar{P}$ belong to $\bar{A}(\bar{B})$. Then

(1) A sub-matrix of order $r \times (2m-1)$ of $\bar{P}$ obtained by choosing any r rows of $\bar{P}$ also belongs to $\bar{A}(\bar{B})$.

(2) The sub-matrix obtained by deleting the last (first) $2k$, $k < m$, columns of $\bar{P}$ again belongs to $\bar{A}(\bar{B})$.

Below we use the classes $\bar{A}$ and $\bar{B}$ in order to obtain three types of special cases of $(n/(2m-1)F/\bar{F}_{max})$ problem. In all the special cases obtained below, the processing time matrix $\bar{P}$ can be partitioned column-wise in such a way that each submatrix belongs to $\bar{A} \cup \bar{B}$.

<u>Type I Special Cases</u>

In this type of special cases, the processing time matrix $\bar{P}$ is partitioned as

$$\bar{P} = (\bar{P}^{(1)} : Q_k : \bar{P}^{(2)})$$

where

$$\bar{P}^{(1)} = (P_1 Q_1 P_2 Q_2 \ldots P_{k-1}\ Q_{k-1}\ P_k)$$

and

$$\bar{P}^{(2)} = (P_{k+1}\ Q_{k+1} \ldots P_{m-1}\ Q_{m-1}\ P_m)$$

such that

$$\bar{P}^{(1)} \text{ and } \bar{P}^{(2)} \in \bar{A} \cup \bar{B}.$$

We consider all the four cases of this type

(1) $\bar{P}^{(1)} \in \bar{A}$, $\bar{P}^{(2)} \in \bar{A}$,

(2) $\bar{P}^{(1)} \in \bar{A}$, $\bar{P}^{(2)} \in \bar{B}$,

(3) $\bar{P}^{(1)} \in \bar{B}$, $\bar{P}^{(2)} \in \bar{A}$ and

(4) $\bar{P}^{(1)} \in \bar{B}$, $\bar{P}^{(2)} \in \bar{B}$

and show that $\bar{T}(\pi)$ takes a simple form in each case.

<u>Theorem 2.2.5</u> : Under the partition $\bar{P} = (\bar{P}^{(1)} : Q_k : \bar{P}^{(2)})$

(i) $\bar{P}^{(1)} \in \bar{A}, \bar{P}^{(2)} \in \bar{A} \implies$

$$\bar{T}(\pi) = \max_{1 \le v \le n} \left[\sum_{i=1}^{v} p_{\pi_i 1} + \left(\sum_{j=2}^{k} p_{\pi_v j} + \sum_{j=1}^{k} q_{\pi_v j} \right) + \sum_{i=v}^{n} p_{\pi_i k+1} \right] + \sum_{j=k+2}^{m} (p_{\pi_n j} + q_{\pi_n j-1}) \tag{2.2.6}$$

and the critical path is of the shape DRDR for any permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$.

(ii) $\bar{p}^{(1)} \in \bar{A}$, $\bar{p}^{(2)} \in \bar{B} \Rightarrow$

$$\bar{T}(\pi) = \max_{1\le v\le n} \left[\sum_{i=1}^{v} p_{\pi_i 1} + \left(\sum_{j=2}^{m-1} p_{\pi_v j} + \sum_{j=1}^{m-1} q_{\pi_v j} \right) + \sum_{i=v}^{n} p_{\pi_i m} \right] \qquad (2.2.7)$$

and the critical path is of the shape DRD for any permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$

(iii) $\bar{p}^{(1)} \in \bar{B}$, $\bar{p}^{(2)} \in \bar{A} \Rightarrow$

$$\bar{T}(\pi) = \sum_{j=1}^{k-1} (p_{\pi_1 j} + q_{\pi_1 j}) + \max_{1\le v\le n} \left[\sum_{i=1}^{v} p_{\pi_i k} + q_{\pi_v k} + \sum_{i=v}^{n} p_{\pi_i k+1} \right] + \sum_{j=k+2}^{m} (p_{\pi_n j} + q_{\pi_n j-1}) \qquad (2.2.8)$$

and the critical path is of the shape RDRDR for any permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$

(iv) $\bar{p}^{(1)} \in \bar{B}$, $\bar{p}^{(2)} \in \bar{B} \Rightarrow$

$$\bar{T}(\pi) = \sum_{j=1}^{k-1} (p_{\pi_1 j} + q_{\pi_1 j}) + \max_{1\le v\le n} \left[\sum_{i=1}^{v} p_{\pi_i k} + \left(\sum_{j=k+1}^{m-1} p_{\pi_v j} + \sum_{j=k}^{m-1} q_{\pi_v j} \right) + \sum_{i=v}^{n} p_{\pi_i m} \right] \qquad (2.2.9)$$

and the critical path is of the shape RDRD for any permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$.

<u>Proof</u> : (i) Let $\bar{p}^{(1)} \in \bar{A}$ and $\bar{p}^{(2)} \in \bar{A}$. For any arbitrary permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$, we can write

$$\bar{T}(\pi) = \max_{1\le v\le n} \{ C[(\pi_1, \pi_2, \ldots, \pi_v); 1, k] + q_{\pi_v k} + C[(\pi_v, \pi_{v+1}, \ldots, \pi_n); k+1, m] \} \qquad (2.2.10)$$

where

$$C[(\pi_1,\pi_2,\ldots,\pi_v);1,k] = \max_{1\le w_1\le w_2\le\cdots\le w_{v-1}\le k} [\sum_{j=1}^{w_1} p_{\pi_1 j} + \sum_{j=1}^{w_1-1} q_{\pi_1 j}$$

$$+\sum_{j=w_1}^{w_2} p_{\pi_2 j} + \sum_{j=w_1}^{w_2-1} q_{\pi_2 j} + \cdots + \sum_{j=w_{v-1}}^{k} p_{\pi_v j} + \sum_{j=w_{v-1}}^{k-1} q_{\pi_v j}]$$

and

$$C[(\pi_v,\pi_{v+1},\ldots,\pi_n);k+1,m] = \max_{k+1\le w_v\le w_{v+1}\le\cdots\le w_{n-1}\le m} \sum_{j=k+1}^{w_v} p_{\pi_v j} + \sum_{j=k+1}^{w_v-1} q_{\pi_v j}$$

$$+\sum_{j=w_v}^{w_{v+1}} p_{\pi_{v+1} j} + \sum_{j=w_v}^{w_{v+1}-1} q_{\pi_{v+1} j} + \cdots + \sum_{j=w_{n-1}}^{m} p_{\pi_n j} + \sum_{j=w_{n-1}}^{m-1} q_{\pi_n j}] .$$

We have

$$C[(\pi_1,\pi_2,\ldots,\pi_v);1,k] = \sum_{i=1}^{v} p_{\pi_i 1} + \sum_{j=2}^{k} (p_{\pi_v j} + q_{\pi_v j-1}) \quad \text{since} \quad \bar{P}^{(1)} \in \bar{A}$$

and

$$C[(\pi_v,\pi_{v+1},\ldots,\pi_n);k+1,m] = \sum_{i=v}^{n} p_{\pi_i k+1} + \sum_{j=k+2}^{m} (p_{\pi_n j} + q_{\pi_n j-1})$$

$$\text{since} \quad \bar{P}^{(2)} \in \bar{A} .$$

Now the required result follows from the substitution of the above simple forms of $C[(\pi_1,\pi_2,\ldots,\pi_v);1,k]$ and $C[(\pi_v,\pi_{v+1},\ldots,\pi_n); k+1,m]$ in equation (2.2.10). It is obvious from the equation (2.2.6) that the critical path is of the shape DRDR.

Proofs of (ii), (iii) and (iv) are similar.

Type II Special Case

In this special case, the processing time matrix $\bar{P}$ is partitioned as

$$\bar{P} = (\bar{P}^{(1)}:P_k:\bar{P}^{(2)})$$

where

$$\bar{P}^{(1)} = (P_1 Q_1 P_2 Q_2 \ldots P_{k-1} Q_{k-1})$$

and

$$\bar{P}^{(2)} = (Q_k \; P_{k+1} \; Q_{k+1} \; \cdots \; P_m)$$

such that

$$(\bar{P}^{(1)} : P_k) \in \bar{B} \quad \text{and} \quad (P_k : \bar{P}^{(2)}) \in \bar{A}.$$

Theorem 2.2.6 : Under the above partition of $\bar{P}$

$$\bar{T}(\pi) = \sum_{j=1}^{k-1} (p_{\pi_1 j} + q_{\pi_1 j}) + \sum_{i=1}^{n} p_{\pi_i k} + \sum_{j=k+1}^{m} (p_{\pi_n j} + q_{\pi_n j-1}) \qquad (2.2.11)$$

for any permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$.

Proof : Let $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ be an arbitrary permutation. We can write

$$\bar{T}(\pi) = \max_{1 \le v \le m} \{C[(\pi_1, \pi_2, \ldots, \pi_v); 1, k] + C[(\pi_v, \pi_{v+1}, \ldots, \pi_n); k, m) - p_{\pi_v k}\}. \qquad (2.2.12)$$

we have

$$C[(\pi_1, \pi_2, \ldots, \pi_v); 1, k] = \sum_{j=1}^{k-1} (p_{\pi_1 j} + q_{\pi_1 j}) + \sum_{i=1}^{v} p_{\pi_i k} \quad \text{since } (\bar{P}^{(1)} : P_k) \in \bar{B}$$

and

$$C[(\pi_v, \pi_{v+1}, \ldots, \pi_n); k, m] = \sum_{i=v}^{n} p_{\pi_i k} + \sum_{j=k+1}^{m} (p_{\pi_n j} + q_{\pi_n j-1}) .$$

Substitution of these simple forms of $C[(\pi_1, \pi_2, \ldots, \pi_v); 1, k]$ and $C[(\pi_v, \pi_{v+1}, \ldots, \pi_n); k, m]$ in the equation (2.2.12) gives the required result.

Proposition 2.2.6(a) : The equation (2.2.11) implies

$$(\bar{P}^{(1)} : P_k) \in \bar{B} \quad \text{and} \quad (P_k : \bar{P}^{(2)}) \in \bar{A}.$$

Proof : We can easily prove this by comparing, for each permutation π, $\bar{T}(\pi)$ with $s(\pi, \tau_u)$ for $1 \le u \le k-1$ where $\tau_u = (\sigma_1(1,u), \sigma_2(u,k), \sigma_3(k,k) \ldots \sigma_{n-1}(k,k), \sigma_n(k,m))$ and with $s(\pi, \eta_v)$ for $k+1 \le v \le m$ where $\eta_v = (\sigma_1(1,k), \sigma_2(k,k) \ldots \sigma_{n-2}(k,k), \sigma_{n-1}(k,v), \sigma_n(v,m))$.

Type III Special Cases

In this type of special cases, we partition the processing time matrix $\bar{P}$ as $\bar{P} = (\bar{P}^{(1)} : Q_k : \bar{P}^{(2)} : P_\ell : \bar{P}^{(3)})$ where $\bar{P}^{(1)} = (P_1 Q_1 \dots P_k)$, $\bar{P}^{(2)} = (P_{k+1}\, Q_{k+1} \dots P_{\ell-1}\, Q_{\ell-1})$ and $\bar{P}^{(3)} = (Q_\ell\, P_{\ell+1}\, Q_{\ell+1} \dots P_m)$ or $\bar{P} = (\bar{P}^{(1)} : P_k : \bar{P}^{(2)} : Q_\ell : \bar{P}^{(3)})$ where $\bar{P}^{(1)} = (P_1 Q_1 \dots P_{k-1}\, Q_{k-1})$, $\bar{P}^{(2)} = (Q_k\, P_{k+1}\, Q_{k+1} \dots P_\ell)$ and $\bar{P}^{(3)} = (P_{\ell+1}\, Q_{\ell+1} \dots P_m)$. We consider the cases

(1) $\bar{P}^{(1)} \in \bar{A}$, $(\bar{P}^{(2)} : P_\ell) \in \bar{B}$, $(P_\ell : \bar{P}^{(3)}) \in \bar{A}$

(2) $\bar{P}^{(1)} \in \bar{B}$, $(\bar{P}^{(2)} : P_\ell) \in \bar{B}$, $(P_\ell : \bar{P}^{(3)}) \in \bar{A}$

under the partition $\bar{P} = (\bar{P}^{(1)} : Q_k : \bar{P}^{(2)} : P_\ell : \bar{P}^{(3)})$ and

(3) $(\bar{P}^{(1)} : P_k) \in \bar{B}$, $(P_k : \bar{P}^{(2)}) \in \bar{A}$, $\bar{P}^{(3)} \in \bar{A}$

(4) $(\bar{P}^{(1)} : P_k) \in \bar{B}$, $(P_k : \bar{P}^{(2)}) \in \bar{A}$, $\bar{P}^{(3)} \in \bar{B}$

under the partition $\bar{P} = (\bar{P}^{(1)} : P_k : \bar{P}^{(2)} : Q_\ell : \bar{P}^{(3)})$.

Below we obtain simple form of $\bar{T}(\pi)$ for each case.

Theorem 2.2.9 : Under the partition $\bar{P} = (\bar{P}^{(1)} : Q_k : \bar{P}^{(2)} : P_\ell : \bar{P}^{(3)})$

(i) $\bar{P}^{(1)} \in \bar{A}$, $(\bar{P}^{(2)} : P_\ell) \in \bar{B}$, $(P_\ell : \bar{P}^{(3)}) \in \bar{A} \implies$

$$\bar{T}(\pi) = \max_{1 \le v \le n} \left[\sum_{i=1}^{v} p_{\pi_i 1} + \sum_{j=2}^{\ell-1} p_{\pi_v j} + \sum_{j=1}^{\ell-1} q_{\pi_v j} + \sum_{i=v}^{n} p_{\pi_i \ell} \right] + \sum_{j=\ell+1}^{m} (p_{\pi_n j} + q_{\pi_n j-1}) \tag{2.2.13}$$

and the critical path is of the shape DRDR for any permutation $\pi = (\pi_1, \pi_2, \dots, \pi_n)$.

(ii) $\bar{P}^{(1)} \in \bar{B}$, $(\bar{P}^{(2)}:P_\ell) \in \bar{B}$, $(P_\ell:\bar{P}^{(3)}) \in \bar{A} \implies$

$$\bar{T}(\pi) = \sum_{j=1}^{k-1}(p_{\pi_1 j}+q_{\pi_1 j}) + \max_{1\le v\le n}\Big[\sum_{i=1}^{v} p_{\pi_i k} + \sum_{j=k+1}^{\ell-1} p_{\pi_v j} + \sum_{j=k}^{\ell-1} q_{\pi_v j}$$

$$+ \sum_{i=v}^{n} p_{\pi_i \ell}\Big] + \sum_{j=\ell+1}^{m}(p_{\pi_n j} + q_{\pi_n j-1}) \qquad (2.2.14)$$

and the critical path is of the shape RDRDR for any permutation $\pi = (\pi_1,\ldots,\pi_n)$.

Proof : (i) Let $\pi = (\pi_1,\pi_2,\ldots,\pi_n)$ be an arbitrary permutation. We can write

$$\bar{T}(\pi) = \max_{1\le u\le v\le m} [C(\pi';1,k) + q_{\pi_u k} + C(\pi'';k+1,\ell) + C(\pi''';\ell,m) - P_{\pi_v \ell}] \quad (2.2.15)$$

where

$$\pi' = (\pi_1,\pi_2,\ldots,\pi_u),\ \pi'' = (\pi_u,\pi_{u+1},\ldots,\pi_v)$$

and

$$\pi''' = (\pi_v,\pi_{v+1},\ldots,\pi_n).$$

We have

$$C(\pi';1,k) = \sum_{i=1}^{u} p_{\pi_i 1} + \sum_{j=2}^{k}(p_{\pi_u j} + q_{\pi_u j-1}) \quad \text{since } \bar{P}^{(1)} \in \bar{A}$$

$$C(\pi'';k+1,\ell) = \sum_{j=k+1}^{\ell-1}(p_{\pi_u j} + q_{\pi_u j}) + \sum_{i=u}^{v} p_{\pi_i \ell} \quad \text{since } (\bar{P}^{(2)}:P_\ell) \in \bar{B}$$

and

$$C(\pi''';\ell,m) = \sum_{i=v}^{n} p_{\pi_i \ell} + \sum_{j=\ell+1}^{m}(p_{\pi_n j} + q_{\pi_n j-1}) \quad \text{since } (P_\ell:\bar{P}^{(3)}) \in \bar{A}$$

The required result follows from the substitution of above simple forms of $C(\pi';1,k)$, $c(\pi'';k+1,\ell)$ and $C(\pi''';\ell,m)$ in equation (2.2.15). From the structure of makespan given in (2.2.13) it is clear that the critical path is of the shape DRDR.

Proof of (ii) is similar to the above one.

Theorem 2.2.10 : Under the partition $\bar{P} = (\bar{P}^{(1)}:P_k:\bar{P}^{(2)}:Q_\ell:\bar{P}^{(3)})$

(iii) $(\bar{P}^{(1)}:P_k) \in \bar{B}$, $(P_k:\bar{P}^{(2)}) \in \bar{A}$, $\bar{P}^{(3)} \in \bar{A} \Rightarrow$

$$\bar{T}(\pi) = \sum_{j=1}^{k-1} (p_{\pi_1 j} + q_{\pi_1 j}) + \max_{1 \le v \le n} \left[\sum_{i=1}^{v} p_{\pi_i k} + \sum_{j=k+1}^{\ell} p_{\pi_v j} + \sum_{j=k}^{\ell} q_{\pi_v j} + \sum_{i=v}^{n} p_{\pi_i \ell+1} \right]$$

$$+ \sum_{j=\ell+2}^{m} (p_{\pi_n j} + q_{\pi_n j-1}) \tag{2.2.16}$$

and the critical path is of the shape RDRDR for any permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$.

(iv) $(\bar{P}^{(1)}:P_k) \in \bar{B}$, $(P_k:\bar{P}^{(2)}) \in \bar{A}$, $\bar{P}^{(3)} \in \bar{B} \Rightarrow$

$$\bar{T}(\pi) = \sum_{j=1}^{k-1} (p_{\pi_1 j} + q_{\pi_1 j}) + \max_{1 \le v \le n} \left[\sum_{i=1}^{v} p_{\pi_i k} + \left(\sum_{j=k+1}^{m-1} p_{\pi_v j} + \left(\sum_{j=k}^{m-1} q_{\pi_v j}\right) + \sum_{i=v}^{n} p_{\pi_i m} \right] \tag{2.2.17}$$

and the critical path is of the shape RDRD for any permutation $\pi = (\pi_1, \pi_2 \ldots \pi_n)$.

Proof : Writing

$$\bar{T}(\pi) = \max_{1 \le u \le v \le m} \{C(\pi';1,k) + C(\pi'';k,\ell) - p_{\pi_u k} + q_{\pi_v \ell} + C(\pi''' ; \ell+1,m)] \tag{2.2.18}$$

where π', π'' and π''' are as given above, we can easily prove (iii) and (iv) as in Theorem 2.2.9.

The solution procedures of the above three types of special cases of the hybrid flow shop problem H are similar to those of Szwarc (1978).

We shall now give briefly modified Szwarc's solution procedures for type I, type II and type III special cases of hybrid flowshop problem according to the shape of critical path.

1. <u>Shape of critical path is DR</u> : Find the job whose sum of processing times on machines W_1, M_2, W_2 ,..., M_m is least. Any sequence with this job at the end is optimal.

2. <u>Shape of critical path is RD</u> : This case can be solved by considering its reverse problem for which the shape of critical path is DR.

3. <u>Shape of critical path is RDR with the segment D occuring on column P_k</u> : Let $\sigma(i,j)$, $i \neq j$, be a permutation of $1,2,\ldots,n$ with $i(j)$ in the first (last) place and the remaining integers in the increasing order. There are $\binom{n}{2}$ such permutations and the one that gives minimum makespan among these is optimal.

4. <u>Shape of critical path is DRD</u> : Solve by Johnson's method the $(n/2/F/F_{max})$ problem with $P_1 + Q_1 + P_2 + \ldots + Q_{m-1}$ and $Q_1 + P_2 + Q_2 + \ldots + P_m$ as first and second columns of processing time matrix. The sequence obtained in this way is optimal.

5. <u>Shape of critical path is RDRD with first D segment occuring on column P_k</u> : Solve by Johnson's method the $(n/2/F/F_{max})$ problem with $P_k+Q_k+ \ldots + P_{m-1} + Q_{m-1}$ and $Q_k+P_{k+1} +\ldots+ Q_{m-1}+P_m$ as first and second columns. Suppose, without loss of generality, the sequence obtained in this way is $(1,2,\ldots,n)$. Let $S' = \{(i,1,2,\ldots \ldots i-1, i+1,\ldots n)/1 \leq i \leq n\}$. The sequence that gives minimum makespan in the set S' is optimal.

6. <u>Shape of critical path is DRDR</u> : This case can be solved by considering its reverse problem for which the shape of critical path is RDRD.

7. <u>Shape of critical path is RDRDR with first D segment occuring on column P_k and second D segment occuring on column P_ℓ</u> : Solve by Johnson's method, the $(n/2/F/F_{max})$ problem with $P_k+Q_k+\ldots+P_{\ell-1}+Q_{\ell-1}$ and $Q_k+P_{k+1}+\ldots+Q_{\ell-1} + P_\ell$ as first and second columns. Suppose without loss of generality, the sequence obtained in this way is $(1,2,\ldots,n)$. Now if the procedure suggested for RDR shape is applied, we obtain optimal sequence.

Doduction of Various $(n/m/F/F_{max})$ Spocial Casos

We now doduce various spocial casos of the $(n/m/F/F_{max})$ problem from the abovo spocial casos of hybrid flow shop problem. First, we introduce two classos of matrices A and B which are holpful in classifying a majority of $(n/m/F/F_{max})$ spocial casos.

Dcfinition of A and B : A matrix $P=((p_{ij}))_{nxm}$ is said to bolong to class A(B) if

$$T(\pi) = \sum_{i=1}^{n} p_{\pi_i 1} + \sum_{j=2}^{m} p_{\pi_n j} \left(\sum_{j=1}^{m} p_{\pi_1 j} + \sum_{i=2}^{n} p_{\pi_i m} \right)$$

for any pcrmutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ where

$$T(\pi) = \max_{1 \leq v_1 \leq v_2 \leq \ldots \leq v_{n-1} \leq m} \left[\sum_{j=1}^{v_1} p_{\pi_1 j} + \sum_{j=v_1}^{v_2} p_{\pi_2 j} + \ldots + \sum_{j=v_{n-1}}^{m} p_{\pi_n j} \right].$$

Let P be the procossing time matrix of an $(n/m/F/F_{max})$ problem. Then $T(\pi)$ givos the makospan of this problem for the pormutation π. The inequalitics (2.2.4) ((2.2.5)) with $q_{ij} = 0$ are the nccessary and sufficient conditions for P to bolong to A(B).

Remark 2.2.11 : $P \in A(B) \iff \bar{P} \in \bar{A}(\bar{B})$ and $T(\pi) = \bar{T}(\pi) - \sum_{j=1}^{m-1} q_j$ when $q_{ij} = q_j$ for $1 \leq i \leq n$ and $1 \leq j \leq m-1$ whero $\bar{P}$ is as given earlier and q_j a constant. Consequontly, the nocessary and sufficient conditions given in Theorem 2.2.2 (Corollary 2.2.3) are equivalent to the nccessary and sufficient conditions for P to bolong to A(B) when $q_{ij} = q_j$. If P bolongs to A(B), a sub-matrix of P obtainod by choosing any k rows $(1 \leq k \leq n)$ of P again bolongs to A(B) and a sub-matrix of P obtained by doloting last (first) k columns of P again bolongs to A(B).

Below, we consider three types of $(n/m/F/F_{max})$ special cases.

Type I Special Cases

In this type of $(n/m/F/F_{max})$ special cases, the processing time matrix P can be partitioned (column-wise) as $P = (P^{(1)} : P^{(2)})$ such that $P^{(1)}$ and $P^{(2)} \in A \bigcup B$. The four exhaustive cases of this type are

(1) $P^{(1)} \in A, \quad P^{(2)} \in A$,

(2) $P^{(1)} \in A, \quad P^{(2)} \in B$,

(3) $P^{(1)} \in B, \quad P^{(2)} \in A$
and

(4) $P^{(1)} \in B, \quad P^{(2)} \in B$.

The cases (1), (3) and (4) have been considered by Ramamurthy and Prasad (1978) whereas case (2) has been considered by Grabowski and Sislo (1975). For sub-cases of the above four cases, see Johnson (1954), Nabeshima (1961) and (1977), Arthanari, et. al. (1971) , Arthanari (1974), Burdyuk (1969), Gupta (1975), Szwarc (1974), (1977) and (1978). The simple forms that $T(\pi)$ takes in the above cases (1), (2), (3) and (4) can be obtained from equations (2.2.6), (2.2.7), (2.2.8) and (2.2.9) by substituting $q_{ij} = 0$.

Remark 2.2.12 : In the above four cases (1), (2), (3) and (4), the critical path takes the shapes DRDR, DRD, RDRDR, and RDRD respectively. We can easily see this by the simple forms of $T(\pi)$ in the above cases.

Type II Special Case

In this special case, the processing time matrix P can be partitioned as $P = (P^{(1)} : P_k : P^{(2)})$ such that $(P^{(1)} : P_k) \in B$ and $(P_k : P^{(2)}) \in A$. This case has been considered by Grabowski and Sislo (1975). For sub-cases of this, see Arthanari (1974), Burdyuk (1969) and Szwarc (1978).

Remark 2.2.12 : In the above type II special case, the critical path takes the shape RDR. This has been proved very easily first by Ramamurthy and Prasad (1978) and later by Szwarc (1979).

Type III Special Cases

In this type of special cases, the processing time matrix P can be partitioned as $P = (P^{(1)}:P^{(2)}:P_{\ell}:P^{(3)})$ such that $P^{(1)} \in A \bigcup B$, $(P^{(2)}:P_{\ell}) \in B$ and $(P_{\ell}:P^{(3)}) \in A$ or $P = (P^{(1)}:P_k:P^{(3)})$ such that $(P^{(1)}:P_k) \in B$, $(P_k:P^{(2)}) \in A$ and $P^{(3)} \in A \bigcup B$. The four cases of this type are

(1) $P^{(1)} \in A$, $(P^{(2)}:P_{\ell}) \in B$, $(P_{\ell}:P^{(3)}) \in A$,

(2) $P^{(1)} \in B$, $(P^{(2)}:P_{\ell}) \in B$, $(P_{\ell}:P^{(3)}) \in A$

under the partition $P = (P^{(1)}:P^{(2)}:P_{\ell}:P^{(3)})$ and

(3) $(P^{(1)}:P_k) \in B$, $(P_k:P^{(2)}) \in A$, $P^{(3)} \in A$,

(4) $(P^{(1)}:P_k) \in B$, $(P_k:P^{(2)}) \in A$, $P^{(3)} \in B$

under the partition $P = (P^{(1)}:P_k:P^{(2)}:P^{(3)})$.

These cases have been considered only by Ramamurthy and Prasad (1978).

Remarks 2.2.14 : The simple forms that $T(\pi)$ takes in the type III special cases (1), (2), (3) and (4) can be obtained from the equations (2.2.13), (2.2.14), (2.2.16) and (2.2.17) respectively by taking $q_{ij} = 0$. In the above type III cases (1), (2), (3) and (4), the critical path takes the shapes DRDR, RDRDR, RDRDR and RDRD respectively . This can be easily seen by the simple form of $T(\tau)$.

The above type I, type II and type III special cases of the $(n/m/F/F_{max})$ problem can be directly obtained from type I, type II and type III special cases of the problem H (discussed in previous sub-section) respectively by assuming $q_{ij} = q_j$ for $1 \leq i \leq n$ and $1 \leq j \leq m-1$.

Remarks 2.2.15 : The above discussion of $(n/m/F/F_{max})$ special cases is based on the unpublished work of Ramamurthy and Prasad (1978). Szwarc (1978) has considered seven $(n/m/F/F_{max})$ special cases based on the shape of critical path and gave solution procedures for solving them. In each of these seven cases, the critical path takes one of the seven shapes given in previous sub-section. Incidentally, the solution procedure of Szwarc (1978) for any case is valid only when each D segment of the critical path occurs on a particular column for every permutation of rows. Though Szwarc (1978) has not explicitly assumed this condition, it has been implicitly used in solution procedures. In fact, by Remarks 2.2.12, 2.2.13 and 2.2.14 all the above $(n/m/F/F_{max})$ special cases which are based on algebraic conditions on processing times can be considered to be sub-cases of Szwarc's (1978) critical path-based seven special cases. However, it is better to visualize the special cases algebraically because there is no other way to check that the critical path takes a particular shape for every permutation of rows of a given processing time matrix. The advantage of Szwarc's (1978) critical path approach is that it enables one to classify a large number of special cases in such a way that each class of special cases can be solved by one of Szwarc's (1978) solution procedures.

Dominance Criteria and Lower Bounds

Dominance criteria For the $(n/m/F/F_{max})$ problem, Szwarc (1971), Smith and Dudek (1967), Ignall and Schrage (1965), Gupta (1975) and Mc Mohan (1969) have derived certain dominance criteria which can be efficiently used in branch and bound procedure to eliminate some of the nodes and reduce consequently the computational work involved in the procedure. Below, we show that for the hybrid flow shop problem H, also these dominance criteria (with regard to the bottleneck machines) hold.

Let $\pi = (\pi_1,\pi_2,\ldots,\pi_n)$ be a permutation and $\pi^{(1)} = (\pi_1,\pi_2,\ldots,\pi_r)$, $\pi^{(2)} = (\pi_{r+1},\ldots,\pi_s)$, $\pi^{(3)} = (\pi_{s+1},\ldots,\pi_t)$ and $\pi^{(4)} = (\pi_{t+1},\ldots,\pi_n)$. (We follow the convention that $(\pi_u,\ldots,\pi_v) = \phi$ when $u > v$). We can write

$$\bar{T}(\pi) = \max_{1\le u\le v\le w\le m} [\bar{T}(\pi^{(1)},u) + C(\pi^{(2)};u,v) + C(\pi^{(3)};v,w) + C(\pi^{(4)};w,m)] \qquad (2.2.19)$$

For reference, see page 15.

Proposition 2.2.16 : Let σ and σ' be two partial permutations containing the same elements. Then

$$\bar{T}(\sigma',u) \le \bar{T}(\sigma,u) \quad \text{for} \quad 1 \le u \le m \qquad (2.2.20)$$

implies $\qquad \bar{T}(\sigma'\pi) \le \bar{T}(\sigma\pi)$

where π is a partial permutation such that $\sigma \cap \pi = \phi$ and $\sigma \cup \pi = N$.

(We represent the set of elements of a **partial** permutation η by η itself.)

Under the condition (2.2.20), the set of permutations of the type σ' ... dominates the set of permutations of the type σ .....

Proposition 2.2.17 : Let σ be a partial permutation and $k,\ell\ (k\ne\ell) \in N-\sigma$.

Then $$\bar{T}(\sigma k\ell;u) - \bar{T}(\sigma\ell;u) \leq p_{kv} \quad \text{for } 1 \leq u \leq v \leq m \qquad (2.2.21)$$

implies $$\bar{T}(\sigma k\ell\pi'\pi'') \leq \bar{T}(\sigma\ell\pi'k\pi'')$$

where π' and π'' are/any two disjoint partial permutations such that $\pi' \cup \pi'' = N-(\sigma\cup\{k,\ell\})$.

Under the condition (2.2.21), the set of permutations of the type $\sigma k\ell$ ... dominates the set of permutations of the type $\sigma\ell$...k...

Proposition 2.2.18 : For σ,k and ℓ given in proposition 2.2.17,

$$\bar{T}(\sigma k\ell,u) - \bar{T}(\sigma\ell,u) \leq p_{km} \quad \text{for } 1 \leq u \leq m \qquad (2.2.22)$$

implies $\bar{T}(\sigma k\ell\pi) \leq \bar{T}(\sigma\ell\pi k)$ for any partial permutation π satisfying $\pi \cap (\sigma k\ell) = \phi$ and $\pi \cup (\sigma k\ell) = N$.

Under the condition (2.2.22), the set of permutations of the type $\sigma k\ell$ ... dominates the set of permutationsof the type $\sigma\ell$ ... k.

Proposition 2.2.19 : For any fixed $k,\ell \in N$, $k \neq \ell$,

$$C((k\ell);u,v) \leq C((\ell k);u,v) \qquad (2.2.23)$$

for $1 \leq u \leq v \leq m$ implies $\bar{T}(\pi'k\ell\pi'') \leq \bar{T}(\pi'\ell k\pi'')$ for any disjoint partial permutations π' and π'' satisfying $\pi', \pi'' \subseteq N-\{k\ell\}$ and $\pi'\cup\pi'' \cup \{k,\ell\} = N$.

Under the condition (2.2.23), the set of permutations of the type ... $k\ell$ dominates the set of permutations of the type...ℓk....

Proposition 2.2.20 For any fixed $k,\ell \in N$, $k \neq \ell$,

$$C(k\ell);u,v) - C((\ell);u,v) \leq p_{kw} \qquad (2.2.24)$$

for $1 \leq u \leq v \leq w \leq m$ implies $\bar{T}(\pi' k\ell \pi'' \pi''') \leq \bar{T}(\pi' \ell \pi'' k \pi''')$ for any disjoint partial permutations π', π'' and π''' satisfying $\pi', \pi'', \pi''' \subseteq N - \{k,\ell\}$ and $\pi' \cup \pi'' \cup \pi''' \cup \{k,\ell\} = N$.

Under the condition (2.2.24), the set of permutations of the type .....$k\ell$...... dominates the set of permutations of the type .....ℓ.....k.... .

By the critical path approach Szwarc (1978) has proved the above dominance criteria for the $(n/m/F/F_{max})$ problem. Using the equation (2.2.19) and following the proofs of Szwarc (1978), we can easily show all the above implications.

Lower Bounds : In the literature, several authors have developed branch and bound procedure with different kinds of lower bounds (on makespan) to solve the $(n/m/F/F_{max})$ problem. We can find a fine classification of all those lower bounds in Lagewig et. al. (1978). On the same lines as in the case of $(n/m/F/F_{max})$ problem, we can derive various lower bounds for the $(n/2m-1/F/\bar{F}_{max})$ rproblem and develop branch and bound procedures using these lower bounds for solving the problem. For example, we give below two types of lower bounds on makespan for the $(n/2m-1/F/\bar{F}_{max})$ problem H.

Lower bound 1 : Let σ be an arbitrary partial permutation and let

$$LBJ(\sigma,u) = \bar{T}(\sigma,u) + \max_{k \in N-\sigma} \left[\sum_{j=u}^{m} p_{kj} + \sum_{j=u}^{m-1} q_{kj} + \sum_{i \in N-\sigma \cup \{k\}} \min(p_{iu}, p_{im}) \right]$$

for $1 \leq u \leq m-1$ and

$$LBJ(\sigma,m) = \bar{T}(\sigma,m) + \sum_{k \in N-\sigma} p_{km} .$$

Then $LBJ(\sigma) = \max_{1 \le u \le m} LBJ(\sigma,u)$ is a lower bound on the makespan for each completion of σ. This bound is a generalisation of the job-based bound given by Mc Mahon et al. (1967) for $(n/m/F/F_{max})$ problem.

<u>Lower bound 2 :</u> Let $LB(\sigma,u) = \min_{i \notin \sigma} d_{iu} + F(N-\sigma;u,m)$ for $1 \le u \le m-1$

where $d_{iu} = \max_{1 \le k \le u} [\bar{T}(\sigma,k) + \sum_{j=k}^{u-1} (p_{ij}+q_{ij})]$ and $F(N-\sigma;u,m)$ is the minimum makespan of an $(n/2/F/F_{max})$ problem with $N-\sigma$ as the set of jobs and $\sum_{j=u}^{m-1} (p_{ij}+q_{ij})$ and $\sum_{j=u+1}^{m} (p_{ij}+q_{ij-1})$ as processing times of job $i, i \in N-\sigma$, on first and second machines respectively.

Let

$$LB(\sigma,m) = \min_{i \notin \sigma} d_{im} + \sum_{i \notin \sigma} p_{im}$$

and

$$LB(\sigma) = \max_{1 \le u \le m} LB(\sigma,u) .$$

We can easily see that $LB(\sigma) \le \bar{T}(\sigma\pi)$ for every completion $\sigma\pi$ of σ, that is, $LB(\sigma)$ is a lower bound on the makespan for each completion of σ. This is a generalisation of a machine-based bound given in the classification of Lagewig, et. al. (1978). Similarly, all the remaining lower bounds suggested for $(n/m/F/F_{max})$ problem can be easily generalised for $(n/2m-1/F/\bar{F}_{max})$ problem.

General Hybrid Flow-Shop Scheduling Problem

Until now, we have considered only a particular hybrid flow shop scheduling problem (H) which consists of bottleneck and non-bottleneck machines at alternate stages with a restriction that the machines at the first and the last stages are bottleneck machines. We now show that any hybrid flow-shop scheduling problem (with the objective of minimising the makespan) consisting of at least one bottleneck machine is equivalent to **the problem $\overset{H}{/}$or a special case of it.**

Consider a 2k-stage, n-job hybrid flow shop scheduling problem H' consisting of k bottleneck machines at stages 1,3,5,... (2k-1) and k non-bottleneck machines at stages 2,4,6,...,2k. Let the processing time matrix of H' be $\bar{P}' = (P_1Q_1P_2Q_2 \ldots P_kQ_k)$, where P_i's and Q_i's are as described earlier. The makespan ($\bar{T}'(\pi)$) of H' for an arbitrary permutation $\pi = (\pi_1,\pi_2,\ldots,\pi_n)$ can be written as

$$\bar{T}'(\pi) = \max_{1\leq v\leq n} \{\bar{T}((\pi_1\pi_2,\ldots,\pi_v),k) + q_{\pi_v k}\}$$

where $\bar{T}((\pi_1\pi_2 \ldots \pi_v),k)$ is as defined earlier. This can be easily verified by mathematical induction on k and n. It is easy to see that $\bar{T}'(\pi) = \bar{T}(\pi)$ for any permutation π when $m = k+1$ and $P_m = 0$. Therefore the problem H' is equivalent to the special case of H where $m = k+1$ and $P_m = 0$.

<u>Remark 2.2.21</u> : For any fixed permutation π, the completion time of a job $i(1\leq i\leq n)$, that is, the time at which the job i comes out of the last machine, need not be same in the problem H' and the corresponding special case of H. In fact, for any fixed permutation the completion time of a job i in the problem H' is less than or equal to that in the corresponding special case of H but the maximum job completion time is same in both the problems.

Consider a 2k-stage, n-job hybrid flow-shop scheduling problem H" consisting of k bottleneck machines at stages 2,4,6,...,2k and k non-bottleneck machines at stages 1,3,5,... (2k-1).

Let the processing time matrix of H" be $\bar{P}" = (Q_1 P_2 Q_2 P_3 \dots P_k Q_k P_{k+1})$ where $P_{j+1} = (p_{1j}, p_{2j}, \dots, p_{nj})$ and $Q_j = (q_{1j}, q_{2j}, \dots, q_{nj})$ for $1 \leq j \leq k$ and $p_{ij}(q_{ij})$ is the processing time of job i on jth bottleneck (non-bottleneck) machine. For any permutation $\pi = (\pi_1, \pi_2, \dots, \pi_n)$, we can write the makespan $\bar{T}"(\pi)$ of H" as

$$\bar{T}"(\pi) = \max_{1 \leq v \leq m} \{q_{\pi_v 1} + C((\pi_v, \pi_{v+1}, \dots, \pi_n); 2, k+1)\}$$

where $C((\pi_v, \dots, \pi_n); 2, k+1)$ is as defined earlier. This can be easily verified by mathematical induction on k and n. It is easy to see that $\bar{T}'(\pi) = \bar{T}(\pi)$ for any permutation π when $m = k+1$ and $P_1 = 0$. Therefore the problem H" is equivalent to the special case of H where $m = k+1$ and $P_1 = 0$.

Consider a (2k-1)-stage, n-job hybrid flow shop problem H"' consisting of K-1 bottleneck machines at stages 2,4,6 ... and k non-bottleneck machines at stages 1,3,5,..., 2k-1. Assume that $k \geq 2$ and let the processing time matrix of H"' be $\bar{P}"' = (Q_1 P_2 Q_2 \dots P_k Q_k)$ where p_j's and Q_j's are as given above. For any permutation $\pi = (\pi_1, \pi_2, \dots, \pi_n)$, we can write the completion time of v th job in the permutation π as

$$T_v(\pi) = \max_{1 \leq u \leq v} \{q_{\pi_u 1} + C((\pi_u \pi_{u+1} \dots \pi_v); 2, k)\} + q_{\pi_v k}$$

where $C(\pi_u, \pi_{u+1}, \dots, \pi_v); 2, k)$ is as defined earlier, since

$\max_{1 \leq u \leq v} \{q_{\pi_u 1} + C(\pi_u, \pi_{u+1}, \dots, \pi_v); 2, k)\}$ is the completion time of v th job in the permutation π on the last bottleneck machine. Now we can write the makespan $\bar{T}"'(\pi)$ of H"' for a permutation $\pi = (\pi_1, \pi_2, \dots, \pi_n)$ as

$$\bar{T}'''(\pi) = \max_{1 \leq u \leq v \leq n} \{q_{\pi_u 1} + C((\pi_u, \pi_{u+1}, \dots, \pi_v); 2, k) + q_{\pi_v k}\} .$$

We can easily verify that $\bar{T}'''(\pi) = \bar{T}(\pi)$ for any permutation π when $m = k+1$ and $P_1 = P_m = 0$. Therefore the problem H''' is equivalent to the special case of H where $m = k+1$ and $P_1 = P_m = 0$.

Remark 2.2.22 : The problems H, H', H'' and H''' together represent the class of hybrid flow-shop scheduling problems consisting of bottleneck and non-bottleneck machines at alternate stages.

Remark 2.2.23 : If there are more than one non-bottleneck machines between any two successive bottleneck machines in a given hybrid flow shop problem, these non-bottleneck machines can be combined as a single non-bottleneck machine. If there is no non-bottleneck machine between two successive bottleneck machines we can assume that between those two successive bottleneck machines there is one non-bottleneck machine on which the processing time of each job is zero.

Since the hybrid flow shop problem without any bottleneck machines is trivial, we classify only those problems which consist of at least one bottleneck machine as hybrid flow shop problems. We can formulate an $(n/m/F/F_{max})$ problem as an $(n/2m-1/F/\bar{F}_{max})$ hybrid flow shop problem by Remarks 2.2.23. We can easily see by Remarks 2.2.22 and 2.2.23 that the problems H, H', H'' and H''' represent the class of all hybrid flow shop problems. Finally, we conclude that any hybrid flow shop problem is equivalent to the problem H or a special case of it.

2.3 Flow Shops with No Intermediate Storages

Introduction

In this section, we consider flow shops with no intermediate storages.

In practical life, there are some flow shops in which job are not allowed to wait at any intermediate stages. For example, in metal processing industries the metal should be rolled immediately after it becomes red hot otherwise the rolling will be extremely difficult. One can find a good illustration of no waiting at intermediate stages in computer systems in which the processing stages are memory unit, central processing unit and input-output unit. Wismer (1972) has considered an n-job, m-machine scheduling problem with no intermediate storages, with the objective of minimising the makespan. In this problem,each job is to be processed on m machines in a particular order and this order need not be the same for all jobs. Wismer has formulated this problem as a travelling salesman problem. Reddy and Ramamurthy (1972) have independently considered an n-job, m-machine flow shop scheduling problem with no intermediate storages (FSNIS), with the same objective as above and formulated it as a travelling salesman problem. Based on the work of Gilmore and Gomary (1964), they have shown that there exists a solution method requiring $O(n^2)$ steps to minimise the makespan for two-machine FSNIS problem. Later, Panwalkar and Wollam (1979) have considered an OFSNIS problem (FSNIS with ordered processing times) with the objective of minimising the makespan. The authors have posed a conjecture for the OFSNIS problem and have solved the OFSNIS special cases.

We consider a special case of FSNIS problem which is slightly more general than OFSNIS problem and obtain two results similar to those of Panwalkar and Wollam (1979). Using these two results, we finally show that the conjecture posed by Panwalkar and Wollam (for OFSNIS problem holds under a slightly more general set-up. In this section, the words 'sequence' and 'permutation' are synonymously used.

Notation and Preliminary Results

Let $N = \{1,2,\ldots,n\}$ be the set of n jobs to be processed and $M_1M_2 \ldots M_m$ the machine order. Let p_{ij}, $1 \leq i \leq n$ and $1 \leq j \leq m$, represent the processing time of job i on machine M_j. Let

$$D_{ik} = p_{i1} + \max\{0,(p_{i2}-p_{k1}),(p_{i2}+p_{i3}-p_{k1}-p_{k2}),\ldots,\sum_{j=2}^{m}(p_{ij}-p_{kj-1})\}$$

and $D'_{ik} = D_{ik}-p_{i1}$ for $1 \leq i,k \leq n$, $i \neq k$.

D'_{ik} is the idle time of the first machine after processing the job i and before taking up the job k in any permutation schedule in which the job k immediately follows job i. We can represent $T(\pi)$, the makespan for a permutation (permutation schedule) $\pi = (\pi_1,\pi_2,\ldots,\pi_n)$, as

$$T(\pi) = D_{\pi_1\pi_2} + D_{\pi_2\pi_3} + \ldots + D_{\pi_{n-1}\pi_n} + \sum_{j=1}^{m} p_{\pi_n j} \qquad (2.3.1)$$

Remark 2.3.1 : For an FSNIS problem, the permutation schedules are the only feasible schedules when all the processing times are positive. If some of the processing times are zero, we have to consider, in addition to permutation schedules, the non-premutation feasible schedules also in order to minimise the makespan over the set of schedules. However, we consider only the permutation schedules because of the complex nature of the non-permutation schedules.

Remarks 2.3.2 : Reddy and Ramamurthy (1972) have formulated the FSNIS problem with the objective of minimising the makespan as an (n+1)-city travelling salesman problem. We can see this by the equation (2.3.1). The FSNIS problem under consideration is equivalent to a travelling salesman problem with (n+1) cities 0,1,2,....,n and distances

$$d_{ij} = \begin{cases} D_{ij} & \text{for} \quad 1 \leq i,j \leq n, \ i \neq j \\ 0 & \text{for} \quad i = 0, \ 1 \leq j \leq n \\ \sum_{h=1}^{m} p_{ih} & \text{for} \quad 1 \leq i \leq n, \ j = 0. \end{cases}$$

Let

$$C'_{ki} = D'_{ik} + \sum_{j=2}^{m} (p_{kj-1} - p_{ij})$$

and

$$C_{ki} = C'_{ki} + p_{km} \quad \text{for} \quad 1 \leq i,k \leq n, \quad i \neq k.$$

Note that $C'_{ki} = D'^{*}_{ki}$ where D'^{*}_{ki}'s are the idle times of the machine M_m in the reverse problem. Using the above equations, we can also represent $T(\pi)$ as

$$T(\pi) = D_{\pi_1\pi_2} + D_{\pi_2\pi_3} + \ldots + D_{\pi_{r-1}\pi_r} + \sum_{j=1}^{m} p_{\pi_r j} + C_{\pi_n\pi_{n-1}} + C_{\pi_{n-1}\pi_{n-2}} +$$

$$+ C_{\pi_{n-1}\pi_{n-2}} \ldots + C_{\pi_{r+1}\pi_r} \quad \text{for} \quad 2 \leq r \leq n-1 \qquad (2.3.2)$$

or

$$T(\pi) = \sum_{j=1}^{m} p_{\pi_1 j} + C_{\pi_n\pi_{n-1}} + \ldots + C_{\pi_2\pi_1}. \qquad (2.3.3)$$

From the equation (2.3.3), we can easily see that $T(\pi) = T^*(\pi^*)$ where $T^*(\pi^*)$ is the makespan of the reverse problem for the permutation $\pi^* = (\pi_n, \pi_{n-1} \ldots \ldots \pi_2, \pi_1)$ and $\min_{\pi} T(\pi) = \min_{\pi} T^*(\pi)$. Therefore, any FSNIS problem and its corresponding reverse problem are equivalent when the objective is to minimise the makespan.

Panwalkar and Wollam (1979) have considered two types of ordered FSNIS problem (FSNIS with ordered processing times), (1) type A problem in which $p_{i1} \geq p_{ij}$ for $1 \leq i \leq n$ and $1 \leq j \leq m$ and (2) type B problem in which

$p_{im} \geq p_{ij}$ for $1 \leq i \leq n$ and $1 \leq j \leq m$. The authors have shown that the order of non-increasing processing times (LPT sequence) minimises the makespan for type A problem whereas in type B problem the order of non-decreasing processing times (SPT sequence) gives the minimum makespan. Finally the authors have conjectured that for any ordered FSNIS problem (with the objective of minimising the makespan), there exists an optimal solution which is a pyramid permutation.

We consider a special case of FSNIS problem in which the processing times p_{ij}'s satisfy

(i) for any $i,k \in N$, $p_{ij} \geq p_{kj}$ for $1 \leq j \leq m$ if there exists a u, $1 \leq u \leq m$, such that $p_{iu} > p_{ku}$ and

(ii) there exists an s, $1 \leq s \leq m$, such that $p_{is} \geq p_{ij}$ for $1 \leq i \leq n$ and $1 \leq j \leq m$.

We denote this problem by $\overline{\text{O}}$FSNIS. Note that the $\overline{\text{O}}$FSNIS problem is slightly more general than the OFSNIS (ordered FSNIS) problem. We assume that the jobs are named in the non-decreasing order of processing times.

Lemma 2.3.3 : For $\overline{\text{O}}$FSNIS problem with $s = 1$,

(i) $D'_{ik} = 0$ for $i < k$, (2.3.4)

(ii) $D'_{ik} \geq D'_{ii-1} + D'_{i-1k}$ for $i > k + 1$, (2.3.5)

(iii) $\sum\limits_{j=2}^{m} p_{ij} - \sum\limits_{j=2}^{m} p_{1j} \geq D'_{i1}$ for $i > 1$, (2.3.6)

(iv) $D'_{\pi_1\pi_2} + D'_{\pi_2\pi_3} + \dots + D'_{\pi_{n-1}\pi_n} \geq D'_{nn-1} + D'_{n-1\ n-2} + \dots + D_{\pi_n+1\ \pi_n}$

(2.3.7)

for any permutation $\pi = (\pi_1, \pi_2, \dots, \pi_n)$ with $\pi_n \neq n$.

Proof: (i) For $i < k$ and $2 \leq h \leq m$, we have

$$\sum_{j=2}^{h} p_{ij} - \sum_{j=1}^{h-1} p_{kj} = \sum_{j=2}^{h-1} (p_{ij}-p_{kj})-(p_{k1}-p_{ih}) \leq 0$$

since $p_{k1} \geq p_{kj} \geq p_{ij}$ for $1 \leq j \leq m$

i.e.,

$$D'_{ik} = 0 .$$

(ii) We have

$$D'_{i\,i-1} + D'_{i-1\,k} = \max\{0, p_{i2}-p_{i-1\,1}, p_{i2}+p_{i3}-p_{i-1\,1}-p_{i-1\,2}, \ldots \sum_{j=2}^{m}(p_{ij}-p_{i-1\;j-1})\}$$

$$+ \max\{0, p_{i-1\,2}-p_{k1}, p_{i-1\,2}+p_{i-1\,3}-p_{k1}-p_{k2}, \ldots \sum_{j=2}^{m}(p_{i-1\,j}-p_{k\;j-1})\}.$$

In order to prove $D'_{i\,i-1} + D'_{i-1\,k} \leq D'_{ik}$, it is enough to show that **(a)** for any h_1 and h_2, $2 \leq h_1, h_2 \leq m$, there exists a $h_3, 2 \leq h_3 \leq m$, such that

$$\sum_{j=2}^{h_1} (p_{ij}-p_{i-1\;j-1}) + \sum_{j=2}^{h_2} (p_{i-1\,j} - p_{k\;j-1}) \leq \sum_{j=2}^{h_3} (p_{ij}-p_{k\;j-1})$$

and (b)

$$\sum_{j=2}^{h} (p_{ij}-p_{i-1\;j-1}) \leq \sum_{j=2}^{h} (p_{ij}-p_{k\;j-1}),$$

$$\sum_{j=2}^{h} (p_{i-1\,j}-p_{k\,j-1}) \leq \sum_{j=2}^{h} (p_{ij}-p_{kj-1}) \quad \text{for} \quad 2 \leq h \leq m .$$

Consider any h_1 and h_2, $2 \leq h_1, h_2 \leq m$.

Case 1 : $h_1 > h_2$.

We have

$$\sum_{j=2}^{h_1} (p_{ij}-p_{i-1\,j-1}) + \sum_{j=2}^{h_2} (p_{i-1\,j} - p_{k\,j-1})$$

$$= \sum_{j=2}^{h_1} p_{ij} - \sum_{j=1}^{h_2-1} p_{kj} - p_{i-1\,1} - \sum_{j=h_2+1}^{h_1-1} p_{i-1\,j}$$

$$= \sum_{j=2}^{h_1} (p_{ij}-p_{k\,j-1}) + \sum_{j=h_2}^{h_1-1} p_{kj} - p_{i-1\,1} - \sum_{j=h_2+1}^{h_1-1} p_{i-1\,j}$$

$$\leq \sum_{j=2}^{h_1} (p_{ij}-p_{k\,j-1}) \quad \text{since } p_{i-1\,1} \geq p_{i-1j} \geq p_{kj} \quad \text{for } 1 \leq j \leq m$$

Case 2. $h_1 \leq h_2$.

In this case, we have

$$\sum_{j=2}^{h_1} (p_{ij}-p_{i-1\,j-1}) + \sum_{j=2}^{h_2} (p_{i-1\,j} - p_{k\,j-1})$$

$$= \sum_{j=2}^{h_1} p_{ij} - p_{i-1\,1} + \sum_{j=h_1}^{h_2} p_{i-1\,j} - \sum_{j=1}^{h_2-1} p_{kj}$$

$$\leq \sum_{j=2}^{h_2} (p_{ij}-p_{kj-1}) \quad \text{since } p_{ij} \geq p_{i-1j} \text{ and } p_{i-1\,1} \geq p_{i-1\,j} \quad \text{for } 1 \leq j \leq m$$

Now, we can write, for any h_1 and h_2, $2 \leq h_1, h_2 \leq m$,

$$\sum_{j=2}^{h_1} (p_{ij}-p_{i-1\,j-1}) + \sum_{j=2}^{h_2} (p_{i-1\,j} - p_{k\,j-1}) \leq \sum_{j=2}^{h_3} (p_{ij}-p_{kj-1})$$

where $h_3 = \max(h_1,h_2)$. It is obvious that $\sum_{j=2}^{h} (p_{ij}-p_{i-1\,j-1}) \leq \sum_{j=2}^{h} (p_{ij}-p_{k\,j-1})$

and $\sum_{j=2}^{h} (p_{i-1\,j}-p_{k\,j-1}) \leq \sum_{j=2}^{h} (p_{ij}-p_{k\,j-1})$ since $p_{ij} \geq p_{i-1\,j} \geq p_{kj}$ for $1 \leq j \leq m$

Therefore $D'_{ik} \geq D'_{ii-1} + D'_{i-1k}$ for $i > k+1$.

From inequality (2.3.5) it follows trivially that

$$D'_{ik} \geq D'_{ii-1} + D'_{i-1\ i-2} + \cdots + D'_{k+1\ k} \text{ for } i > k+1 \qquad (2.3.8)$$

(iii) For any h, $2 \leq h \leq m$, we have

$$\sum_{j=2}^{h} (p_{ij}-p_{1j-1}) = \sum_{j=2}^{m} p_{ij} - \sum_{j=2}^{m} p_{1j} - (\sum_{j=h+1}^{m} p_{ij} - \sum_{j=h}^{m} p_{1j} + p_{11})$$

$$\leq \sum_{j=2}^{m} p_{ij} - \sum_{j=2}^{m} p_{1j}$$

since $p_{ij} \geq p_{1j} \leq p_{11}$ for $1 \leq j \leq m$.

It is obvious that

$$\sum_{j=2}^{m} p_{ij} - \sum_{j=2}^{m} p_{1j} \geq 0.$$

Therefore

$$D'_{i1} \leq \sum_{j=2}^{m} p_{ij} - \sum_{j=2}^{m} p_{1j} .$$

(iv) Consider an arbitrary permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ with $\pi_r = n$ for a fixed r, $1 \leq r < n$. We have $D'_{\pi_r \pi_{r+1}} \geq D'_{nn-1} + D'_{n-1\ n-2} + \cdots$ $\cdots D'_{(\pi_{r+1}+1)\pi_{r+1}}$ by inequality (2.3.8). If $r+2 \leq n$ and $\pi_{r+1} < \pi_{r+2}$, then

$$D'_{\pi_r \pi_{r+1}} + D'_{\pi_{r+1} \pi_{r+2}} = D'_{\pi_r \pi_{r+1}} \quad \text{by the equation (2.3.4)}$$

$$\geq D'_{\pi_r\ \pi_{r+2}} \quad \text{since } \pi_{r+1} < \pi_{r+2}$$

$$\geq D'_{nn-1} + \cdots + D'_{\pi_{r+2}+1\ \pi_{r+2}} \quad \text{by (2.3.8).}$$

In case $\pi_{r+1} > \pi_{r+2}$

$$D'_{\pi_r \pi_{r+1}} + D'_{\pi_{r+1} \pi_{r+2}} \geq D'_{nn-1} + \dots + D_{\pi_{r+1}+1\ \pi_{r+1}} + D_{\pi_{r+1}\pi_{r+1}-1} + \dots + D_{\pi_{r+2}+1\ \pi_{r+2}}$$

(by inequality (2.3.8))

$$= D'_{nn-1} + D'_{n-1\ n-2} + \dots + D_{\pi_{r+2}+1\ \pi_{r+2}}.$$

Similarly when $r+3 \leq r$,

$$D'_{\pi_r\pi_{r+1}} + D'_{\pi_{r+1}\ \pi_{r+2}} + D'_{\pi_{r+2}\ \pi_{r+3}} \geq D'_{nn-1} + D'_{n-1\ n-2} + \dots + D'_{\pi_{r+3}+1\ \pi_{r+3}}.$$

Repeating this argument successively, we finally get

$$D'_{\pi_r\pi_{r+1}} + D'_{\pi_{r+1}\pi_{r+2}} + \dots + D'_{\pi_{n-1}\pi_n} \geq D'_{nn-1} + D'_{n-1\ n-2} + \dots + D'_{\pi_n+1\ \pi_n}$$

which implies the inequality (2.3.7).

Main Results

Theorem 2.3.4 : For $\tilde{O}$FSNIS problem with $s = 1$, LPT sequence $p^* = (n, n-1, \dots, 2, 1)$ minimises the makespan.

Proof : Consider an arbitrary sequence, say, $\pi = (\pi_1, \pi_2, \dots, \pi_n)$. We have

$$T(\pi) = \sum_{i=1}^{n} p_{i1} + D'_{\pi_1\pi_2} + D'_{\pi_2\pi_3} + \dots + D'_{\pi_{n-1}\pi_n} + \sum_{j=2}^{m} p_{\pi_n j}$$

and

$$T(p^*) = \sum_{i=1}^{n} p_{i1} + D'_{nn-1} + D'_{n-1\ n-2} + \dots + D'_{2\,1} + \sum_{j=2}^{m} p_{1j}$$

which imply

$$T(\pi) - T(p^*) = D'_{\pi_1\pi_2} + D'_{\pi_2\pi_3} + \dots + D'_{\pi_{n-1}\pi_n} - (D'_{nn-1} + D'_{n-1\ n-2} + \dots + D'_{2\,1}) + \sum_{j=2}^{m} p_{\pi_n j} - \sum_{j=2}^{m} p_{1j}$$

Case 1 : $\pi_n = n$.

In this case, $T(\pi)-T(p^*) \geq D'_{\pi_1\pi_2}+\ldots+D'_{\pi_{n-1}\pi_n} - (D'_{nn-1}+\ldots+D'_{21})$

$+ D'_{nn-1} + D'_{n-1\ n-2} + \ldots + D'_{21}$ by inequality (2.3.6)

≥ 0.

Case 2 : $\pi_n = 1$.

Now $T(\pi)-T(p^*) \geq D'_{nn-1}+D'_{n-1\ d-2}+\ldots+ D'_{21} -(D'_{nn-1}+D'_{n-1\ n-2}+\ldots+D'_{21})$

(by inequality (2.3.7))

$= 0$.

Case 3 : $\pi_n = k \neq 1,n$.

In this case, $T(\pi)-T(p^*) \geq (D'_{nn-1}+D'_{n-1\ n-2}+\ldots+D'_{k+1\ k})-(D'_{nn-1}+D'_{n-1\ n-2}+\ldots D'_{21})$

$+ (D'_{kk-1}+D'_{k-1\ k-2} + \ldots + D'_{21})$

(by the inequalities (2.3.5), (2.3.6) and (2.3.7)

$= 0$.

Therefore $T(\pi) \geq T(p^*)$ for any arbitrary sequence π i.e., p^* is an optimal solution.

Corollary 2.3.5 : For $\bar{O}$FSNIS problem with $s = m$, SPT sequence $p = (1,2,\ldots,n)$ minimises the makespan.

Proof : The reverse problem of $\bar{O}$FSNIS problem with $s = m$ is an $\bar{O}$FSNIS problem with $s = 1$, for which LPT sequence p^* is optimal. Therefore SPT sequence p is optimal for $\bar{O}$FSNIS problem with $s = m$.

Remark 2.3.6 : All the above results have been obtained by Panwalkar and

Wollam (1979) for OFSNIS problem. We now consider $\overline{\text{O}}$FSNIS problem with no restriction on s. Panwalkar and Wollam (1979) have conjectured that for any OFSNIS problem, the set of pyramid permutations contains an optimal solution.

Using the Theorem 2.3.4 and Corollary 2.3.5, we show that the conjecture of Panwalkar and Wollam holds for $\overline{\text{O}}$FSNIS problem also.

<u>Theorem 2.3.6</u> : For any $\overline{\text{O}}$FSNIS problem, the set of pyramid permutations contains an optimal solution.

<u>Proof</u> : Since the conjecture is true for s = 1 and m by Theorem 2.3.4 and Corollary 2.3.5, it is enough to consider the case $s = t,\ 1 < t < m$.

Let

$$_tD'_{ik} = \max\{0, p_{i2}-p_{k1}, p_{i2}+p_{i3}-p_{k1}-p_{k2}, \ldots, \sum_{j=2}^{t}(p_{ij}-p_{k\,j-1})\}$$

and

$$_tC'_{ki} = \max\{0, p_{km-1}-p_{im}, p_{km-1}+p_{km-2}-p_{im}-p_{im-1}, \ldots \sum_{j=t}^{m-1}(p_{kj}-p_{ij+1})\}$$

for $i,k \in N,\ i \neq k$.

For any $i,k \in N,\ i \neq k$,

$$D'_{ik} \geq {}_tD'_{ik} \text{ and } C'_{ki} \geq {}_tC'_{ki}\,. \tag{2.3.9}$$

One can easily verify that $D'_{ik} = {}_tD'_{ik}$ for $i < k$ and $C'_{ki} = {}_tC'_{ki}$ for $k < i$

Consider an arbitrary sequence $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ with $\pi_r = n$ for for a fixed r, $1 < r < n$. We can write

$$T(\pi) := (\sum_{i=1}^{r-1} p_{\pi_i 1} + D'_{\pi_1\pi_2} + D'_{\pi_2\pi_3} + \ldots + D'_{\pi_{r-1}\pi_r} + \sum_{j=1}^{t} p_{nj})$$

$$+ (\sum_{j=t}^{m} p_{nj} + C'_{\pi_n\,\pi_{n-1}} + C'_{\pi_{n-1}\pi_{n-2}} + \ldots + C'_{\pi_{r+1}\pi_r} + \sum_{i=r+1}^{n} p_{\pi_i m}) - p_{nt}$$

$$= I_1 + I_2 - p_{nt}$$

where

$$I_1 = (\sum_{i=1}^{r-1} p_{\pi_i 1} + t^{D'}_{\pi_1\pi_2} + t^{D'}_{\pi_2\pi_3} + \dots + t^{D'}_{\pi_{r-1}\pi_r} + \sum_{j=1}^{t} p_{nj})$$

and

$$I_2 = (\sum_{j=t}^{m} p_{nj} + t^{C'}_{\pi_n\pi_{n-1}} + t^{C'}_{\pi_{n-1}\pi_{n-2}} + \dots + t^{C'}_{\pi_{r+1}\pi_r} + \sum_{r+1}^{n} p_{\pi_i m}).$$

Note that I_1 is the makespan corresponding to a sequence $(\pi_1, \pi_2, \dots, \pi_{r-1}, n)$ of the $\bar{O}$FSNIS problem obtained from the original problem by ignoring the machines $M_{t+1}, M_{t+2}, \dots M_m$ and the jobs $\pi_{r+1}, \pi_{r+2}, \dots \pi_n$. Similarly I_2 is the makespan corresponding to the sequence $(n, \pi_{r+1}\pi_{r+2}, \dots, \pi_n)$ of the $\bar{O}$FSNIS problem obtained from the original problem by ignoring the machines $M_1, M_2, \dots, M_{t-1}$ and the jobs $\pi_1, \pi_2, \dots, \pi_{r-1}$. We can easily see, by Theorem 2.3.2 and Corollary 2.3.3,

$$I_1 \geq T_1(q')$$

and

$$I_2 \geq T_2(q'')$$

where $T_1(q')$ $(T_2(q''))$ is the makespan of the former (latter) $\bar{O}$FSNIS problem for the SPT(LPT) sequence $q'(q'')$ of jobs $\pi_1, \pi_2, \dots, \pi_{r-1}, n$ $(n, \pi_{r+1}, \pi_{r+2}, \dots, \pi_n)$ i.e., $T(\pi) \geq T_1(q') + T_2(q')-p_{nt}$.

Using (2.3.8), we can easily see that

$$T_1(q') + T_2(q'')-p_{nt} = T(q_1q_2\dots q_{r-1}n\, q_{r+1}\dots q_n)$$

where $(q_1q_2\dots q_r\ n) = q'$ and $(n, q_{r+1}, q_{r+2}, \dots, q_n) = q''$.

Therefore, $T(\pi) \geq T(q_1q_2\dots q_{r-1}n\ q_{r+1}\ \dots\ q_n)$ where $(q_1q_2, \dots, q_{r-1}\ n, q_{r+1}\dots$ $\dots q_n)$ is pyramid permutation. Similarly we can show that $T(\pi) \geq T(p)$ when $\pi_n = n$ and $T(\pi) \geq$ ~~$T(p)$~~ $T(p^*)$ when $\pi_1 = n$. Hence it is enough to consider the set of pyramid permutations for minimising the makespan of $\bar{O}$FSNIS problem.

Remark 2.3.7 : The results of this section are from the author's technical report, "On flowshop scheduling problem with no in-process waiting", No. 7940, December 1979, Indian Statistical Institute, Delhi Centre. S.P.Rana and R.K.Arora have considered, in "Scheduling in a semi-ordered flowshop without intermediate queues", AIIE Transactions, Vol. 12, pp. 263-272, September, 1980 , an FSNIS problem in which the processing times are required to satisfy only the condition that for any k and $\ell (1 \leq k,\ell \leq n)$, $p_{kj} \geq p_{\ell j}$ for $1 \leq j \leq m$ if there exists a v, $1 \leq v \leq m$, such that $p_{kv} > p_{\ell v}$. For this problem which is obviously more general than ŌFSNIS problem, Rana and Arora have obtained a result (Theorem 1 of Rana and Arora (1980)) from which the conjecture of Panwalkar and Wollam (1971) follows as a special case. The author has obtained the results of this section entirely independent of Rana and Arora. Rana and Arora's approach is completely different from that of the author, which is originally due to Panwalkar and Wollam (1979). Moreover, Theorem 2.3.4 and Corollary 2.3.5 which have led to the proof of the conjecture (of this section) have not been given in Rana and Arora (1980).

2.4 Ordered Flow Shop Scheduling Problems

Introduction :

In this section, we consider an ordered $(n/m/F/F_{max})$ problem. Smith et. al. (1975) have introduced this problem pointing out its practical basis. In a study on ordered $(n/m/F/F_{max})$ problems, Panwalkar and Khan (1977) have numerically observed a convex property of the makespan with respect to the position of the largest job in the permutation. Assuming the convex property to hold true, Panwalkar and Khan (1977) have presented an efficient algorithm to solve the ordered $(n/m/F/F_{max})$ problem. In this section, our main interest is to establish the convex property observed by Panwalkar and Khan. We show that the convex property indeed holds for the $(3/3/F/F_{max})$ problem.

Preliminary Results

Smith et. al. (1975) have obtained the following result for the ordered $(n/m/F/F_{max})$ problem .

Theorem 2.4.1 (Smith, et. al. (1975)) : For an ordered $(n/m/F/F_{max})$ problem with $s = 1(m)$, the permutation $(n,n-1,\ldots,2,1)((1,2,\ldots,n))$ is optimal. Later, Smith et. al. (1976) have proved the following interesting result.

Theorem 2.4.2 (Smith, et. al. (1976)) : For any ordered $(n/m/F/F_{max})$ problem, the set of pyramid permutations contains an optimal solution.

Moreover, Smith et. al. (1976) have presented an algorithm to evaluate all 2^{n-1} pyramid permutations and suggested incorporation of branch and bound technique for further improvement.

Let S_r be the set of all pyramid permutations with the largest job n in the r th place and T_r^* represent the minimum makespan over the set S_r. Panwalker and Khan (1977) have randomly generated 200 ordered $(n/m/F/F_{max})$ problems and observed in all of them that there exists an integer k, $1 \leq k \leq n$, such that

$$T_1^* \geq T_2^* \geq \cdots \geq T_k^* \leq T_{k+1}^* \leq \cdots \leq T_n^* \qquad (2.4.1)$$

For a permutation $\pi = (\pi_1,\pi_2,\ldots,\pi_n)$ of the problem under consideration, we have

$$T(\pi) = \max_{1\leq w_1 \leq w_2 \leq \cdots \leq w_{n-1} \leq m} [\sum_{j=1}^{w_1} p_{\pi_1 j} + \sum_{j=w_1}^{w_2} p_{\pi_2 j} + \cdots + \sum_{j=w_{n-1}}^{m} p_{\pi_n j}]$$

(2.4.2)

In order to prove the Theorem 2.4.1, Smith et. al. (1975) have obtained a result that for an ordered $(n/m/F/F_{max})$ problem with $s = 1(m)$, $T(v) \leq T(v')$ $(T(v) \geq T(v'))$ where $v = (v_1,v_2,\ldots,v_{i-1},v_i,v_{i+1},v_{i+2},\ldots,v_n)$ and $v' = (v_1,v_2,\ldots,v_{i-1}, v_{i+1},v_i, v_{i+2}\ldots,v_n)$ when $v_i = n$.

Remark 2.4.3 : By the above mentioned result of Smith et. al. (1975), we can easily see that for an ordered ($n/m/F/F_{\max}$) problem with $s = 1(m)$

$$T_1^* \leq T_2^* \leq \cdots \leq T_n^* (T_1^* \geq T_2^* \geq \cdots \geq T_n^*) \qquad (2.4.3)$$

In the following theorem, we prove the convex property for ordered ($3/3/F/F_{\max}$) problem.

Theorem 2.4.4 : For a 3 x 3 ordered flowshop problem

$$T_2^* \leq \max(T_1^*, T_3^*) \qquad (2.4.4)$$

Proof : For $s = 1(3)$, the inequality (2.4.4) holds trivially by Remark 2.4.3. So, we consider only the case $s = 2$. Note that the jobs are named in the increasing order of processing times, that is, $p_{1j} \leq p_{2j} \leq p_{3j}$ for $j = 1,2$ and 3.

Let T_1, T_2', T_2'' and T_3 be the makespans corresponding to the permutations (3,2,1), (1,3,2), (2,3,1) and (1,2,3) respectively. Since the largest element (processing time) p_{32} is common in all makespans, we can ignore this element for the purpose of comparing makespans. Thus, we get

$$T_1 = p_{32} + \max(p_{33}+p_{23}, p_{22}+p_{23}, p_{22}+p_{12}) + p_{13},$$

$$T_2' = \max(p_{11}+p_{12}+p_{33}+p_{23}, p_{11}+p_{31}+p_{33} +p_{23}, p_{11}+p_{12}+p_{22}+p_{23}, p_{11}+p_{31}+p_{22}+p_{23}),$$

$$T_2'' = \max(p_{21}+p_{22}+p_{33}+p_{13}, p_{21}+p_{31}+p_{33}+p_{13}, p_{21} +p_{22}+p_{12}+p_{13}, p_{21}+p_{31}+p_{12}+p_{13})$$

and

$$T_3 = \max(p_{11}+p_{12}+p_{22}+p_{33}, p_{11}+p_{21}+p_{22}+p_{23}, p_{11}+p_{21}+p_{31}+p_{33}).$$

and

$$T_1^* = T_1, \quad T_2^* = \min(T_2', T_2''), \quad T_3^* = T_3.$$

Case (1) : $p_{i2} \geq p_{i3} \geq p_{i1}$ for $i = 1,2,3,$

Suppose

$$T_2^* > T_3^* . \tag{2.4.5}$$

We shall now prove, in four exhaustive cases, that $T_1^* \geq T_2^*$

Case (a) : $p_{22} \leq p_{33}$, $p_{12} \leq p_{21}$.

In this case, we have

$$T_1 = p_{31}+p_{33}+p_{23}+p_{13}, \quad T_2' = p_{11}+p_{31}+p_{33}+p_{23}.$$

It is obvious that $T_1 \geq T_2'$ i.e. $T_1^* \geq T_2^*$.

Case (b) : $p_{22} > p_{33}$, $p_{12} \leq p_{21}$.

Here, we have

$$T_1 = p_{31}+p_{22}+p_{23}+p_{13} \quad \text{and} \quad T_2' = p_{11}+p_{31}+p_{22}+p_{23}$$

which imply $T_1^* \geq T_2^*$.

Case (c) $p_{22} \leq p_{33}$, $p_{12} > p_{21}$.

Here, we have

$$T_1 = p_{31}+p_{13}+ \max(p_{33}+p_{23}, p_{22}+p_{12}) \quad ,$$

$$T_2' = p_{11}+p_{23}+ \max(p_{12}, p_{31}) + p_{33}$$

$$T_2'' = p_{21}+p_{13}+p_{33}+ \max(p_{22}, p_{31})$$

and $T_3 = p_{11}+p_{33} + \max(p_{12}+p_{22}, p_{21}+p_{31})$.

If $p_{12} < p_{31}$,

$$\begin{aligned} T_1 - T_2' &= p_{31}-p_{11}+p_{13}-p_{23}+\max(p_{33}+p_{23}, p_{22}+p_{12}) - p_{33}-p_{31} \\ &\geq p_{13} - p_{11} \\ &\geq 0 \end{aligned}$$

i.e., $T_1^* \geq T_2^*$.

Otherwise, we have by inequality (2.4.5)

$$T_2'' - T_3 \geq 0$$

which implies $p_{21} + p_{13} + p_{22} - p_{11} - (p_{12} + p_{22}) \geq 0$

i.e.

$$p_{21} - p_{11} + p_{13} - p_{12} \geq 0 \;. \tag{2.4.6}$$

Now it is easy to see that

$$\begin{aligned} T_1 - T_2' &\geq p_{31}-p_{11}+p_{13}-p_{23}+(p_{33}+p_{23})-p_{12}-p_{33} \\ &= p_{31}-p_{11}+p_{13}-p_{12} \\ &\geq 0 \quad \text{by inequality (2.4.6)}, \end{aligned}$$

i.e. $T_1^* \geq T_2^*$.

<u>Case (d)</u> : $p_{22} > p_{33}$, $p_{12} > p_{21}$.

In this case, we have

$$T_1 = p_{31}+p_{22}+p_{13}+\max(p_{23}, p_{12}) \;,$$

$$T_2' = p_{11}+p_{22}+p_{23}+\max(p_{12}, p_{31}) \;,$$

$$T_2'' = p_{21}+p_{22}+p_{13}+\max(p_{33}, p_{12})$$

and $T_3 = p_{11}+p_{12}+p_{22}+p_{33}$.

If $p_{33} < p_{12}$, $T_1 - T_2'' = p_{31} - p_{21} \geq 0$ i.e. $T_1^* \geq T_2^*$.

Otherwise, we have, by inequality (2.4.5)

$$T_2'' - T_3 \geq 0$$

which implies $_{21} - p_{11} + p_{13} - p_{12} \geq 0.$

Now it easily follows that

$$T_1 - T_2' \geq p_{31} - p_{11} + p_{13} - \max\,(p_{12}, p_{31})$$

$$\geq 0 \ .$$

i.e., $T_1^* \geq T_2^*$

Case (2) : $p_{i2} \geq p_{i1} \geq p_{i3}$ for $i = 1,2,3,$

It is easy to see by case (1) that the condition (2.4.4) holds for the reverse problem. Consequently it holds for the original problem also. Hence the result.

Remark 2.4.6 : Even though the above result is small, it throws some light on the validity of the convex property using which Panwalkar and Khan (1977) have developed **a good** algorithm for solving the ordered $(n/m/F/F_{max})$ problem.

CHAPTER III

n-JOB, 2-MACHINE STOCHASTIC FLOW SHOP SCHEDULING PROBLEMS

3.1 Introduction

In this chapter, we deal with n-job, two-machine (nx2) stochastic flow shop scheduling problems with the objective of minimising the expected makespan. We have considered in previous chapter the deterministic flow shop scheduling problems in which the processing times are assumed to be known and fixed. But this assumption does not hold always. In some practical situations, the processing times are random following some known probabilistic laws. In such cases the objectives are generally in terms of expected values. We give below a brief account of the work done on stochastic flow shop scheduling problems. Makino (1965) was the first to consider the stochastic flow shop scheduling problems. Makino has considered two-job, two-machine and two-job, three-machine stochastic flow shop problems in which the processing times are assumed to be independent random variables following exponential distributions with known parameters. The objective of these problems is to minimise the expected makespan. Later, Talwar (1967) has conjectured an optimal rule for scheduling the jobs so as to minimise the expected makespan for nx2 stochastic flow shop problems with exponential processing times. Bagga (1970) has shown that Talwar's conjecture is true for $n \leq 4$. Cunningham and Dutta (1973) have rigorously proved the optimality of Talwar's rule and gave a recursive method for obtaining the value of expected makespan for a given permutation schedule. The above authors have considered only the class of schedules while dealing with stochastic flow shop problems. For a stochastic flow shop problem, the class of policies are, as mentioned in Chapter I, wider than the class of schedules and therefore it is proper to consider the class of policies instead of the class of schedules. Unfortunately, the problem becomes very complex when we deal with policies. For this reason, we restrict our attention to the class of schedules in all sections of this chapter except the last one in which we formulate a stochastic flow shop problem as a finite dynamic programming problem.

In section 3.2, we consider two nx2 stochastic flow shop problems – (I) with all the processing times following geometric distributions and (II) with processing times on one machine following geometric distributions and those on the other machine being identically distributed and we obtain optimal solutions for both the problems. We derive the optimal solution (for exponential case) of Cunningham and Dutta (1973) using the optimal solution of the problem I. In section 3.3, we derive optimality criteria in general terms for the case in which the processing times follow some known distributions. In section 3.4, we consider an nx2 stochastic flow shop problem in which (i) the processing times on first machine follow some known distributions, (ii) the processing times on second machine follow exponential distributions with known parameters and (iii) no-passing is allowed. We formulate this problem as a finite dynamic programming problem. The words 'sequence' and 'permutation' are synonymously used in this chapter.

Notation : For nx2 stochastic flow shop scheduling problems, we represent the first machine by A and the second one by B. The processing times of job i, $1 \leq i \leq n$, on machines A and B are represented by A_i and B_i respectively. The expected makespan for a permutation (permutation schedule) $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ is given by

$$T(\pi) = E[\max_{1 \leq k \leq n} \{ \sum_{i=1}^{k} A_{\pi_i} + \sum_{i=k}^{n} B_{\pi_i} \}] \quad . \qquad (3.2.1)$$

We know that the expression $\max_{1\le k\le n} \{ \sum_{i=1}^{k} A_{\pi_i} + \sum_{i=k}^{n} B_{\pi_i} \}$ gives the makespan of deterministic flow shop for the permutation π, when A_i's and B_i's are deterministic. We can easily verify that it holds for stochastic case also.

For the problem with the objective of minimising the expected makespan, Cunningham and Dutta (1973) showed that the set of permutation schedules dominates the set of non-permutation schedules, that is, it is enough to consider only the permutation schedules in order to minimise the expected makespan over the set of schedules. Therefore, we consider, in sections 3.2 and 3.3, only the permutations schedules and obtain a permutation p such that

$$T(p) = \min_{\pi \in S} T(\pi)$$

where S is the set of all permutations of $1,2,\ldots,n$.

3.2 The Flow Shop With Geometric Processing Times

Introduction

In this section, we shall study an n-job, 2-machine (nx2) stochastic flow shop scheduling problem with processing times following geometric distributions, with the objective of minimising the expected makespan. As mentioned earlier, we restrict our attention to the set of schedules. We first obtain a statistical result and deduce three corollaries from it. Using these corollaries, we obtain, for the problem under consideration, an optimal schedule that minimises the expected makespan. Also we give an optimal schedule for the case where the processing times on one machine are identically distributed and those on the other machine follow geometric distributions. We shall derive Cunningham and Dutta's (1973) optimality criteria (for exponential case) using the above optimal schedules and some limiting arguments. Finally, we present a recursive method for obtaining the value of expected makespan for a given permutation schedule of the problem under consideration. In the problem considered above, we make two more additional assumptions--(1) time proceeds in discrete steps, i.e., t takes values $0,1,2\ldots\ldots$ and (2) all the processing times follow geometric distributions with known parameters.

Let A_i and B_i, $1 \leq i \leq n$, follow geometric distributions with known parameters a_i and b_i, $0 < a_i, b_i < 1$, respectively. We have

$$P[A_i = m] = a_i(1-a_i)^{m-1} ,$$

$$P[B_i = m] = b_i(1-b_i)^{m-1} \quad \text{for } m = 1,2,\ldots \text{and } 1 \leq i \leq n.$$

Preliminary Results

Lemma 3.2.1 : Let X_1, X_2, Y_1 and Y_2 be independent discrete random variables following geometric distribution with parameters r_1, r_2, s_1 and s_2, respectively. Then

$$E[\min(a+X_1+X_2, b+Y_1+Y_2, X_1, Y_2)] \geq E[\min(a+X_1+X_2, b+Y_1+Y_2, X_2, Y_1)], \quad (3.2.2)$$

for all real values of a and b, if $\frac{1-r_1}{1-s_1} \geq \frac{1-r_2}{1-s_2}$.

Proof : First we prove the lemma for any integer values of a and b. It is enough to show that

$$P\cdot(a+X_1+X_2 \geq m,\ b+Y_1+Y_2 \geq m, X_1 \geq m, Y_2 \geq m)$$

$$\geq P\cdot(a+X_1+X_2 \geq m,\ b+Y_1+Y_2 \geq m, X_2 \geq m, Y_1 \geq m)$$

for any integer values of a,b and m when

$$\frac{1-r_1}{1-s_1} \geq \frac{1-r_2}{1-s_2} .$$

Let

$$P_1(m) = P\ (a+X_1+X_2 \geq m, b+Y_1+Y_2 \geq m,\ X_1 \geq m, Y_2 \geq m)$$

and $P_2(m) = P\ (a+X_1+X_2 \geq m, b+Y_1+Y_2 \geq m,\ X_2 \geq m, Y_1 \geq m)$.

Since X_1, X_2, Y_1 and Y_2 are independent, we can write

$$P_1(m) = P_{x_1}(m)\ P_{y_2}(m)$$

and

$$P_2(m) = P_{x_2}(m)\ P_{y_1}(m)$$

where

$$P_{x_1}(m) = P\ (a+X_1+X_2 \geq m, X_1 \geq m)\ ,$$

$$P_{x_2}(m) = P\ (a+X_1+X_2 \geq m, X_2 \geq m)\ ,$$

$$P_{y_1}(m) = P\ (b+Y_1+Y_2 \geq m, Y_1 \geq m)$$

and

$$P_{y_2}(m) = P\ (b+Y_1+Y_2 \geq m, Y_2 \geq m)$$

and m is an integer.

Case 1 : $(a \geq 0, b \geq 0)$.

We have

$$P_{x_1}(m) = (1-r_1)^{m-1}\ ,$$

$$P_{x_2}(m) = (1-r_2)^{m-1}\ ,$$

$$P_{y_1}(m) = (1-s_1)^{m-1}$$

and

$$P_{y_2}(m) = (1-s_2)^{m-1},\ \text{for } m \geq 1$$

and

$$P_{x_1}(m) = P_{x_2}(m) = P_{y_1}(m) = P_{y_2}(m) = 1 \text{ for } m \leq 0\ .$$

Hence $P_1(m) = P_2(m)$ for $m \leq 0$.

$$\frac{P_1(m)}{P_2(m)} = [(\frac{1-r_1}{1-s_1})/(\frac{1-r_2}{1-s_2})]^{m-1} \quad \text{for} \quad m \geq 1.$$

It is obvious that $P_1(m) \geq P_2(m)$ for $m \geq 1$ when

$$\frac{1-r_1}{1-s_1} \geq \frac{1-r_2}{1-s_2} .$$

Hence $P_1(m) \geq P_2(m)$ for all m when $\frac{1-r_1}{1-s_1} \geq \frac{1-r_2}{1-s_2}$.

<u>Case 2</u> : $(a < 0, b \geq 0)$.

For $a \leq m \leq 0$,

$$\begin{aligned} P_{x_1}(m) &= \sum_{i=1}^{m-a-1} P(X_2 \geq m-a-i)P(X_1=i) + P(X_1 > m-a-1) \\ &= \frac{r_1(1-r_2)^{m-a-1} - r_2(1-r_1)^{m-a-1}}{r_1 - r_2} \\ &= f(r_1,r_2,a;m), \text{ say} . \end{aligned}$$

Similarly, $P_{x_2}(m) = f(r_2,r_1,a;m)$ for $a \leq m \leq 0$. It is easy to see that

$$f(r_1,r_2,a;m) = f(r_2,r_1,a;m) .$$

We have $P_{y_1}(m) = 1 = P_{y_2}(m)$ for $m \leq 0$

and $P_{x_1}(m) = 1 = P_{x_2}(m)$ for $m \leq a$.

Hence $P_1(m) = P_2(m)$ for all $m \leq 0$.

For $m > 0$,

$$\begin{aligned} P_{x_1}(m) &= \sum_{i=m}^{m-a-1} (1-r_2)^{m-a-i-1} r_1(1-r_1)^{i-1} + (1-r_1)^{m-a-1} \\ &= (1-r_1)^{m-1} \frac{r_1(1-r_2)^{-a} - r_2(1-r_1)^{-a}}{r_1-r_2} \\ &= (1-r_1)^{m-1} g(r_1,r_2,a), \text{ say} . \end{aligned}$$

Similarly, $P_{x_2}(m) = (1-r_2)^{m-1} g(r_2,r_1,a)$ for $m > 0$.

Here also the equality $g(r_1,r_2,a) = g(r_2,r_1,a)$ holds true.

$$P_{y_1}(m) = (1-s_1)^{m-1} \text{ and } P_{y_2}(m) = (1-s_2)^{m-1} \quad \text{for} \quad m > 0 \ .$$

Therefore

$$\frac{P_1(m)}{P_2(m)} = [(\frac{1-r_1}{1-s_1}) / (\frac{1-r_2}{1-s_2})]^{m-1} \quad \text{for} \quad m > 0.$$

It is obvious that $P_1(m) \geq P_2(m)$ for $m > 0$ when

$$\frac{1-r_1}{1-s_1} \geq \frac{1-r_2}{1-s_2} \ .$$

Hence $P_1(m) \geq P_2(m)$ for all m when $\frac{1-r_1}{1-s_1} \geq \frac{1-r_2}{1-s_2}$

<u>Case 3</u> : $(a \geq 0, b < 0)$.

The argument for this case is similar to the above one.

<u>Case 4</u> : $(a < 0, b < 0)$.

For this case, $P_{x_1}(m) = P_{x_2}(m)$, $P_{y_1}(m) = P_{y_2}(m)$ for $m \leq 0$

and

$$P_{x_1}(m) = (1-r_1)^{m-1} g(r_1,r_2,a) \ ,$$

$$P_{x_2}(m) = (1-r_2)^{m-1} g(r_2,r_1,a) \ ,$$

$$P_{y_1}(m) = (1-s_1)^{m-1} g(s_1,s_2,b)$$

and

$$P_{y_2}(m) = (1-s_2)^{m-1} g(s_2,s_1,b), \quad \text{for} \quad m \geq 1 \ .$$

It is easy to verify that

$$P_1(m) \geq P_2(m) \quad \text{for all} \quad m,$$

when

$$\frac{1-r_1}{1-s_1} \geq \frac{1-r_2}{1-s_2} \ .$$

Therefore

$$E[\min(a+X_1+X_2, b+Y_1+Y_2, X_1, Y_2)] \geq E[\min(a+X_1+X_2, b+Y_1+Y_2, X_2, Y_1)],$$

when

$$\frac{1-r_1}{1-s_1} \geq \frac{1-r_2}{1-s_2} .$$

Now, it is easy to see that the above statement holds for any real values of a and b.

Corollary 3.2.2 : If X_1 and X_2(Y_1 and Y_2) in Lemma 3.2.1 are identical an follow some general distribution,

$$E[\min(a+X_1+X_2, b+Y_1+Y_2, X_1, Y_2)] \geq E[\min(a+X_1+X_2, b+Y_1+Y_2, X_2, Y_1)]$$

for all real values of a and b if $s_1 \geq s_2 (r_1 \leq r_2)$.

Proof : If X_1 and X_2 are i.i.d. random variables, it is enough to show that

$$P_{y_2}(m) \geq P_{y_1}(m) \ \forall \ m \quad \text{when } s_1 \geq s_2.$$

This follows by the argument similar to the one in Lemma 3.2.1. Similarly, if Y_1 and Y_2 are i.i.d. random variables, the inequality(3.2.2)holds for $r_1 \leq r_2$.

Corollary 3.2.3 : Let X_1, X_2, Y_1 and Y_2 be independent discrete random variables following geometric distribution with parameters r_1, r_2, s_1 and s_2, respectively. Let Z_1 and Z_2 be two random variables which are independent of X_1, X_2, Y_1 and Y_2. Then

$$E[\min\{Z_1+X_1+X_2, Z_2+Y_1+Y_2, X_1, Y_2\}] \geq E[\min \{Z_1+X_1+X_2, Z_2+Y_1+Y_2, X_2, Y_1\}]$$

if

$$\frac{1-r_1}{1-s_1} \geq \frac{1-r_2}{1-s_2} .$$

Proof : Since Z_1 and Z_2 are independent of X_1, X_2, Y_1 and Y_2, one can easily see, by Lemma 3.2.1 that the above statement is true,

Corollary 3.2.4 : If X_1 and $X_2 (Y_1$ and $Y_2)$ in the Corollary 3.2.3 are i.i.d. random variables,

$$E[\min\{Z_1+X_1+X_2, Z_2+Y_1+Y_2, X_1, Y_2\}] \geq E[\min\{Z_1+X_1+X_2, Z_2+Y_1+Y_2, X_2, Y_1\}]$$

for $s_1 \geq s_2 (r_1 \leq r_2)$.

Main Results

In the following theorem, we obtain an optimal sequence that minimises the expected makespan of the problem under consideration.

Theorem 3.2.5 : A sequence $p = (p_1, p_2, \dots, p_n)$ satisfying the inequality $C_{p_1} \leq C_{p_2} \leq \dots \leq C_{p_n}$ where $C_i = \frac{1-a_i}{1-b_i}$ minimizes the expected makespan.

Proof : Consider the sequence $v = (1,2,\dots,n)$. The corresponding expected makespan is given by

$$T(v) = E[\max_{1\leq k\leq n} \{\sum_{i=1}^{k} A_i + \sum_{i=k}^{n} B_i\}] .$$

Suppose $C_i \geq C_{i+1}$ for a fixed $i, 1 \leq i \leq n-1$. We can write

$$\max_{1\leq k\leq n} \{\sum_{i=1}^{k} A_i + \sum_{i=k}^{n} B_i\} = \max\{F+A_i+A_{i+1}, G+B_i+B_{i+1}, H+A_i+B_{i+1}+\max(A_{i+1}, B_i)\}$$

where

$$F = \sum_{j=1}^{i-1} A_j + \max_{i+2\leq k\leq n} (\sum_{j=i+2}^{k} A_j + \sum_{j=k}^{n} B_j) ,$$

$$G = \max_{1\leq k\leq i-1} (\sum_{j=1}^{k} A_j + \sum_{j=k}^{i-1} B_j) + \sum_{j=i+2}^{n} B_j$$

and

$$H = \sum_{j=1}^{i-1} A_j + \sum_{j=i+2}^{n} B_j .$$

Note that $\sum_{j=n_1}^{n_2} = 0$ if $n_2 < n_1$.

Hence

$$\max_{1\le k\le n} \{\sum_{j=1}^{k} A_j + \sum_{j=k}^{n} B_j\}$$

$$= H+A_i+A_{i+1}+B_i+B_{i+1} + \max(F-H-B_i-B_{i+1},G-H-A_i-A_{i+1},-B_i,-A_{i+1})$$

$$= H+A_i+A_{i+1}+B_i+B_{i+1} - \min(Z_1+A_i+A_{i+1},Z_2+B_i+B_{i+1},A_{i+1},B_i)$$

where

$$Z_1 = H-G \quad\text{and}\quad Z_2 = H-F .$$

Now we have, for the sequence v,

$$T(v) = E(H) + \frac{1}{a_i} + \frac{1}{a_{i+1}} + \frac{1}{b_i} + \frac{1}{b_{i+1}}$$

$$- E[\min(Z_1+A_i+A_{i+1},Z_2+B_i+B_{i+1},A_{i+1},B_i)]$$

Similarly,for thesequence $v' = (1,2\ldots i-1, i+1,i,i+2,\ldots,n)$, which is obtained from v by interchanging i and i+1, we have

$$T(v') = E(H) + \frac{1}{a_i} + \frac{1}{a_{i+1}} + \frac{1}{b_i} + \frac{1}{b_{i+1}}$$

$$- E[\min(Z_1+A_i+A_{i+1},Z_2+B_i+B_{i+1},A_i,B_{i+1})].$$

$$T(v)-T(v') = E[\min(Z_1+A_i+A_{i+1},Z_2+B_i+B_{i+1},A_i,B_{i+1})]$$

$$- E[\min(Z_1+A_i+A_{i+1},Z_2+B_i+B_{i+1},A_{i+1},B_i)].$$

Since $C_i \geq C_{i+1}$ and Z_1 and Z_2 are independent of A_i,A_{i+1}, B_i and B_{i+1}, we have, by Corollary 3.2.3,

$$T(v) \geq T(v').$$

Since the relation $C_i \geq C_{i+1}$ is transitive, it is easy to see that the sequence p for which $C_{p_1} \leq C_{p_2} \leq \ldots \leq C_{p_n}$ holds minimises the expected makespan.

Corollary 3.2.6 If all A_i's (B_i's) are identical following some general distribution and all B_i's (A_i's) following geometric distributions as mentioned above, a sequence π satisfying conditions $b_{\pi_1} \leq b_{\pi_2} \leq \cdots \leq b_{\pi_n}$ ($a_{\pi_n} \geq a_{\pi_2} \cdots \geq a_{\pi_n}$) is an optimal one.

By applying Corollary 3.2.4 instead of Corollary 3.2.3 in Theorem 3.2.5 one can easily verify this result.

Remarks : In the nx2 stochastic flow shop scheduling problem considered above, we have assumed that all processing times are discrete random variables following geometric distributions. This assumption is not new in stochastic scheduling theory. For reference, see Glazebrook (1979). Cunningham and Dutta's (1973) result for exponential case can easily be proved on the same lines as above.

Cunningham and Dutta's Optimality Criteria

Now, we obtain two results and prove Cunningham and Dutta's optimality criteria(for minimizing expected makespan of nx2 flow shop scheduling problem with exponential processing times) using those results and Theorem 3.2.5.

Lemma 3.2.7 : Let $a_1, a_2, \ldots, a_n$; $b_1, b_2, \ldots, b_n$ be finite positive numbers such that

$$a_1 - b_1 \geq a_2 - b_2 \geq \cdots \geq a_n - b_n \qquad (3.2.3)$$

Then we can construct sequences $\{a_{1m}\}$, $\{a_{2m}\}$, ... $\{a_{nm}\}$; $\{b_{1m}\}, \{b_{2m}\}, \ldots, \{b_{nm}\}$, $m = 1,2$ ... in such a way that

(α) $0 < a_{im}, b_{im} < 1, 1 \leq i \leq n, \forall\ m < \infty$,

(β) $a_{im} \to 0$, $b_{im} \to 0$, $ma_{im} \to a_i$, $mb_{im} \to b_i$, $1 \leq i \leq n$, as $m \to \infty$,

(γ) there exists an integer M such that

$$\frac{1-a_{1m}}{1-b_{1m}} \le \frac{1-a_{2m}}{1-b_{2m}} \le \cdots \le \frac{1-a_{nm}}{1-b_{nm}} \quad \forall\ m \ge M.$$

Proof : We prove this Lemma, separately, for two cases.

Case 1 : $a_1-b_1 > a_2-b_2 > \cdots > a_n-b_n$.

Consider an integer $N > \max(a_1,a_2,\ldots,a_n,b_1,b_2,\ldots,b_n)$. Let

$$a_{im} = \frac{a_i}{m},\ b_{im} = \frac{b_i}{m}, 1 \le i \le n,\ \forall\ m \ge N \qquad (3.2.4)$$

Now the condition (β) is satisfied. We can choose some values for a_{im}, b_{im}, $1 \le i \le n, m < N$ so that the condition (α) is also satisfied. Using(3.2.4) it is easy to find an integer $M(\ge N)$ such that

$$\frac{1-a_{1m}}{1-b_{1m}} \le \frac{1-a_{2m}}{1-b_{2m}} \le \cdots \le \frac{1-a_{nm}}{1-b_{nm}} \quad \forall\ m \ge M.$$

Case 2 : There exists at least one equality in(3.2.3).

Consider any fixed i and i+1, $i+1 \le n$. Given two sequences $\{a_{i+1m}\}$, $\{b_{i+1m}\}$ satisfying the conditions (α) and

$$a_{i+1m} = \frac{a_{i+1}}{m} + \frac{c_{i+1}}{m^2},\ b_{i+1m} = \frac{b_{i+1}}{m}\ \forall\ m \ge N_{i+1}\ , \qquad (3.2.5)$$

where N_{i+1} is a positive integer and c_{i+1} a finite non-negative value, we construct two sequences $\{a_{im}\}$, $\{b_{im}\}$ in such a way that they satisfy the conditions (α) and (β) and the four sequences $\{a_{im}\}$,$\{b_{im}\}$,$\{a_{i+1m}\}$ and $\{b_{i+1m}\}$ together satisfy the condition (γ).

Let $c_i = a_i b_{i+1} - a_{i+1} b_i + c_{i+1} + \varepsilon$ where $\varepsilon > 0$ if $a_i b_{i+1} - a_{i+1} b_i + c_{i+1} \ge 0$; otherwise, $c_i = 0$. Now, we can easily find an integer N_i such that

$$\frac{a_i}{m} + \frac{c_i}{m^2}\ ,\ \frac{b_i}{m} < 1\ \forall\ m \ge N_i\ .$$

Let

$$a_{im} = \frac{a_i}{m} + \frac{c_i}{m^2} ,$$

$$b_{im} = \frac{b_i}{m} \ \forall\ m \geq N_i \quad . \tag{3.2.6}$$

Combining (3.2.5) and (3.2.6), we get

$$a_{im} = \frac{a_i}{m} + \frac{c_i}{m^2} ,\ a_{i+1m} = \frac{a_{i+1}}{m} + \frac{c_{i+1}}{m^2} ,$$

$$b_{im} = \frac{b_i}{m} \text{ and } b_{i+1m} = \frac{b_{i+1}}{m} \ \forall\ m \geq N = \max(N_i, N_{i+1}). \tag{3.2.7}$$

Note that

$$\frac{1 - \frac{a_i}{m} - \frac{c_i}{m^2}}{1 - \frac{b_i}{m}} \leq \frac{1 - \frac{a_{i+1}}{m} - \frac{c_{i+1}}{m^2}}{1 - \frac{b_{i+1}}{m}} \iff$$

$$\frac{1}{m^2}[(a_i b_{i+1} - a_{i+1} b_i) - c_i(1 - \frac{b_{i+1}}{m}) + c_{i+1}(1 - \frac{b_i}{m})] \leq \frac{1}{m}[(a_i - b_i) - (a_{i+1} - b_{i+1})]. \tag{3.2.8}$$

In case $a_i - b_i = a_{i+1} - b_{i+1}$,

$$\frac{1 - \frac{a_i}{m} - \frac{c_i}{m^2}}{1 - \frac{b_i}{m}} \leq \frac{1 - \frac{a_{i+1}}{m} - \frac{c_{i+1}}{m^2}}{1 - \frac{b_{i+1}}{m}} \iff$$

$$(a_i b_{i+1} - a_{i+1} b_i) - c_i(1 - \frac{b_{i+1}}{m}) + c_{i+1}(1 - \frac{b_i}{m}) \leq 0, \tag{3.2.9}$$

for any positive integer m.

We have either $a_i - b_i > a_{i+1} - b_{i+1}$ or $a_i - b_i = a_{i+1} - b_{i+1}$.

Using (3.2.8) or (3.2.9) according as $a_i - b_i > a_{i+1} - b_{i+1}$ or $a_i - b_i = a_{i+1} - b_{i+1}$ and (3.2.7), we can find an integer ($m \geq N$) such that

$$\frac{1 - a_{im}}{1 - b_{im}} \leq \frac{1 - a_{i+1m}}{1 - b_{i+1m}} \ \forall\ m \geq M.$$

Starting with i+1 = n and $c_n = 0$, we can construct, by the above procedure, sequences $\{a_{1m}\}$, $\{b_{1m}\}, \ldots, \ldots, \{a_{nm}\}$ $\{b_{nm}\}$ satisfying the conditions (α),(β) and (γ).

Lemma 3.2.8 : Let $A_1A_2, \ldots, A_n$; $B_1, B_2, \ldots, B_n$ be independent exponential random variables and $\{A_{im}\}$,$\{B_{im}\}$ m = 1,2 ... sequences of geometric random variables converging in distribution to A_i and B_i respectively for i = 1 to n. Assume that for each m, $A_{1m}, A_{2m}, \ldots, A_{nm}, B_{1m}, B_{2m}, \ldots, B_{nm}$ are defined on sample space $\{\frac{1}{m}, \frac{2}{m}, \frac{3}{m} \ldots\}$ and are independent.

Let

$$T_{wm} = E[\max_{1 \le k \le n} (\sum_{i=1}^{k} A_{w_i m} + \sum_{i=k}^{n} B_{w_i m})],$$

and

$$T_w = E[\max_{1 \le k \le n} (\sum_{i=1}^{k} A_{w_i} + \sum_{i=k}^{n} B_{w_i})] ,$$

where $w = (w_1, w_2, \ldots, w_n)$ is permutation of 1,2,...,n.

Then $T_{vm} = \min_{w \in S} T_{wm}$ $\forall$ $m \geq M$ implies $T_v = \min_{w \in S} T_w$, where M is a positive integer and S is the set of all permutations of 1,2,...,n.

Proof : First, we prove that $T_{wm} \to T_w$ as $m \to \infty$ for any arbitrary permutation w. Consider the permutation $p = (1,2,\ldots,n)$. By the assumption that $A_{1m}, A_{2m}, \ldots, A_{nm}, B_{1m}, B_{2m}, \ldots, B_{nm}$ are independent for each m, we have

$$(A_{1m}, A_{2m}, \ldots, A_{nm}, B_{1m}, B_{2m}, \ldots, B_{nm}) \xrightarrow{\text{in law}} (A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_n)$$

as $m \to \infty$.

Now, by means of an argument involving characteristic functions, it is easy to see that

$$(u_{1m}, u_{2m}, \ldots, u_{km}) \xrightarrow{d} (u_1, u_2, \ldots, u_k) \text{ as } m \to \infty ,$$

where $\overset{d}{\rightarrow}$ denotes convergence in law or in distribution, u_{jm}'s are linear combinations of $A_{1m}, A_{2m}, \ldots, A_{nm}, B_{1m}, B_{2m}, \ldots, B_{nm}$, and u_j's are the same linear combinations of $A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_n$ as u_{jm}'s of A_{im}'s and B_{im}'s. In particular, it follows that

$$F_m(x) \overset{d}{\rightarrow} F(x),$$

where $F_m(x) = P[u_{1m} \le x, u_{2m} \le x, \ldots, u_{km} \le x]$,

and $F(x) = P[u_1 \le x, u_2 \le x, \ldots, u_k \le x]$,

i.e.,

$$\max(u_{1m}, u_{2m}, \ldots, u_{km}) \overset{d}{\rightarrow} \max(u_1, u_2, \ldots, u_k). \quad (3.2.10)$$

Now, take $k = n$ and

$$u_{jm} = \sum_{i=1}^{j} A_{im} + \sum_{i=j}^{n} B_{im} ,$$

$$u_j = \sum_{i=1}^{j} A_i + \sum_{i=j}^{n} B_i \quad \text{for } j = 1,2,\ldots,n.$$

It is easy to see that

$$\max(u_{1m}, u_{2m}, \ldots, u_{nm}) \le \sum_{i=1}^{n} A_{im} + \sum_{i=1}^{n} B_{im}$$

and

$$\begin{aligned} E[\,|\max(u_{1m},\ldots,u_{nm})|^2] &= E[\max(u_{1m},\ldots,u_{nm})^2] \\ &\le E[\{\sum_{i=1}^{n} A_{im} + \sum_{i=1}^{n} B_{im}\}^2] \\ &\le 2[E(\sum_{i=1}^{n} A_{im})^2 + E[(\sum_{i=1}^{n} B_{im})^2] \\ &= 2[\sum_{i=1}^{n} \left(\frac{2}{m^2 a_{im}^2} - \frac{1}{m^2 a_{im}} \right) \\ &\quad + 2\sum_{i<j} \frac{1}{(ma_{im})(ma_{jm})}] \\ &\quad + 2[\sum_{i=1}^{n} \left(\frac{2}{m^2 b_{im}^2} - \frac{1}{m^2 b_{im}} \right) + 2\sum_{i<j} \frac{1}{(mb_{im})(mb_{jm})} \end{aligned}$$

where a_{im}'s and b_{im}'s are parameters of A_{im}'s and B_{im}'s and are $< \infty \ \forall m$.
Also

$$\lim_{m \to \infty} E[|\max(u_{1m} \cdots u_{nm})|^2] \leq 4[\sum_{i=1}^{n} (\frac{1}{a_i^2} + \frac{1}{b_i^2}) + \sum_{i<j} (\frac{1}{a_i a_j} + \frac{1}{b_i b_j})] < \infty.$$

Therefore

$$\sup_m E[|\max(u_{1m}, \ldots, u_{nm})|^2] < \infty . \qquad (3.2.11)$$

Below, we shall state a convergence theorem of Chung(1968)(p. 88, Theorem 4.5.2) which is useful in showing $E[\max(u_{1m},\ldots,u_{nm})] \to E[\max(u_1,\ldots,u_n)]$ as $m \to \infty$.

Statement of the Theorem: If $\{X_m\}$ converges in distribution to X and for some $p > 0$, $\sup_m E[|X_m|^p] = M < \infty$, then for each $r < p$

$$\lim_{m \to \infty} E(|X_m|^r) = E(|X|^r) < \infty .$$

If r is positive integer, then we replace $|X_m|^r$ and $|X|^r$ above by X_m^r and X^r.

By the above theorem, it follows from (3.2.10) and (3.2.11) that

$$E[\max\ (u_{1m}, u_{2m}, \ldots, u_{nm})] \to E[\max\ (u_1, u_2, \ldots, u_n)] \text{ as } m \to \infty .$$

i.e.,

$$T_{pm} \to T_p \quad \text{as} \quad m \to \infty .$$

Similarly, for any arbitrary permutation w of 1,2,...,n,

$$T_{wm} \to T_w \quad \text{as} \quad m \to \infty .$$

It is easy to see that $T_v = \min_{w \in S} T_w$ if $T_{wm} \to T_w$ for any permutation and $T_{vm} = \min_{w \in S} T_w$, $\forall\ m \geq M$ where m is a positive integer.

<u>Theorem 3.2.9</u> : Let $A_1, A_2, \ldots, A_n$; $B_1, B_2, \ldots, B_n$ be independent exponential random variables with parameters $a_1, a_2, \ldots, a_n$; $b_1, b_2, \ldots, b_n$, respectively and $a_{v_1} - b_{v_1} \geq a_{v_2} - b_{v_2} \geq \cdots \geq a_{v_n} - b_{v_n}$. Then

$$T_v = \min_{w \in S} T_w .$$

<u>Proof</u> : Let $\{A_{1m}\}$, $\{A_{2m}\}$,..., $\{A_{nm}\}$; $\{B_{1m}\}$,$\{B_{2m}\}$,..., $\{B_{nm}\}$ be sequences of geometric random variables converging in distribution to $A_1, A_2, \ldots, A_n$, B_1, $B_2, \ldots, B_n$, respectively. Assume that for each m, A_{1m}, A_{2m}, ..., A_{nm}; B_{1m}, $B_{2m}, \ldots, B_{nm}$ are defined on sample space $\{\frac{1}{m}, \frac{2}{m}, \ldots\}$ and are independent. By Lemma 3.2.7 we can construct sequences $\{a_{1m}\}$, $\{a_{2m}\}$, ..., $\{a_{nm}\}$ $\{b_{1m}\}$, $\{b_{2m}\}, \ldots$, $\{b_{nm}\}$, m = 1,2,..., satisfying the conditions (α) and (β) in such a way that there exists an integer M such that

$$\frac{1-a_{v_1 m}}{1-b_{v_1 m}} \leq \frac{1-a_{v_2 m}}{1-b_{v_2 m}} \leq \cdots \frac{1-a_{v_n m}}{1-b_{v_n m}} \;\forall\; m \geq M. \tag{3.2.12}$$

Let these sequences $\{a_{im}\}$, $\{b_{im}\}$, $1 \leq i \leq n$ be parameters of $\{A_{im}\}$, $\{B_{im}\}$ $1 \leq i \leq n$, respectively. Then, by Theorem 3.2.5 it follows that

$$T_{vm} = \min_{w \in S} T_{wm} \;\forall\; m \geq M.$$

Hence, by Lemma 3.2.8, it follows that

$$T_v = \min_{w \in S} T_w.$$

<u>Calculation of the value of expected makespan for a given sequence</u>

We consider the state of the flow shop at time t = 0 and job completion times on both the machines. Suppose a job ℓ is being processed on machine A at a job completion time (τ) on machine B. Due to the memoryless property of geometric distribution, the residual processing time required by the job ℓ on machine A at time τ again follows geometric distribution with the same parameter a_ℓ and therefore the job ℓ can be considered to be waiting for processing on machine A at time τ. The above argument holds for processing of jobs on machine B also. Now, we can represent the state of the flow shop at a job completion time τ by (U,V) where $U \subseteq N$ is the set of jobs waiting for processing before machine A and $V \subseteq N-U$ the set of jobs waiting for processing before machine B at time τ.

Let $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ be a given sequence. Without loss of generality we assume that $\pi = (1,2,\ldots,n)$. Under this sequence (permutation schedule) π, we process the jobs in the order $(1,2,\ldots,n)$ on each machine. Let $E(U,V)$ denote the expected makespan from a job completion time τ onwards under the sequence π where (U,V) is the state of the flow shop at time τ. We can write

$$\left.\begin{aligned}
E(U,V) &= E[\min(A_\ell, B_k)] + P(A_\ell < B_k)E(U-\{\ell\}, V \cup \{\ell\}) \\
&\quad + P(A_\ell > B_k)\, E(U, V-\{k\}) + P(A_\ell = B_k)E(U-\{\ell\}, V \cup \{\ell\} - \{k\}) \\
&\qquad \text{for } U,V \neq \phi , \\
E(U,\phi) &= E(A_\ell) + E(U-\{\ell\},\{\ell\}) \text{ for } U \neq \phi , \\
E(\phi,V) &= \sum_{j\in V} E(B_j) \quad \text{for} \quad V \neq \phi \quad \text{and} \\
E(\phi,\phi) &= 0 ,
\end{aligned}\right\} \qquad (3.2.13)$$

where ℓ and k are the jobs with least indices in the sets U and V respectively. Since A_i and B_i, $1 \leq i \leq n$, follow geometric distributions with parameters a_i and b_i (with means $1/a_i$ and $1/b_i$) respectively, we can rewrite the equation (3.2.13) as

$$\left.\begin{aligned}
E(U,V) &= \frac{1}{a_\ell+b_k-a_\ell b_k} + \frac{a_\ell(1-b_k)}{a_\ell+b_k-a_\ell b_k} E(U-\{\ell\}, V \cup \{\ell\}) + \frac{b_k(1-a_\ell)}{a_\ell+b_k-a_\ell b_k} E(U,V-\{k\}) \\
&\quad + \frac{a_\ell b_k}{a_\ell+b_k-a_\ell b_k} E(U-\{\ell\}, V \cup \{\ell\} - \{k\}) \text{ for } U,V \neq \phi , \\
E(U,\phi) &= \frac{1}{a_\ell} + E(U-\{\ell\},\{\ell\}) \text{ for } U \neq \phi , \\
E(\phi,V) &= \sum_{j\in V} \frac{1}{b_j} \quad \text{for } V \neq \phi \\
\text{and} \\
E(\phi,\phi) &= 0 .
\end{aligned}\right\} \qquad (3.2.14)$$

Under the sequence π, at any job completion time, U and V are of the form

$$U = \{\ell,\ell+1,\ldots,n\}, \quad V = \{k,k+1,\ldots,\ell-1\}, \quad 1 \leq k < \ell \leq n$$

or $$U = \{\ell,\ell+1,\ldots,n\}, \quad V = \phi, \quad 1 \leq \ell \leq n$$

or $$U = \phi, \quad V = \{k,k+1,\ldots,n\}, \quad 1 \leq k \leq n$$

or $$U = \phi, \quad V = \phi .$$

Therefore, considering only the states of the above form we can solve the equations 3.2.14 iteratively and obtain the value of $E(N,\phi)$ which gives the expected makespan for the sequence π. We can also apply this method for obtaining the value of expected makespan for a given sequence when the processing times A_i's and B_i's follow exponential distributions with known parameters.
The results of this section are from Prasad (1980).

3.3 The Flow Shop with General Processing Times

Introduction

So far, only two types of distributions, viz., exponential and geometric distributions are considered for the nx2 stochastic flow shop scheduling problem. Now, we make an attempt on general distributions and obtain optimality criteria in general terms for minimising the expected makespan. We also derive optimality criteria in terms of mondtone likelihood ratio and deduce several special cases.

Under certain conditions, we obtain optimal policy when all A_i's follow uniform or normal or gamma distributions and all B_i's follow uniform or normal or gamma distributions.

Preliminary Results

First, we obtain two statistical results which are useful in obtaining the optimality criteria for the nx2 flow shop problem with general processing times.

Lemma 3.3.1 : Given any four non-negative independent r.v.'s X_1, X_2, Y_1 and Y_2, the inequality(3.2.2) holds under the following four conditions.

(1) $\bar{\alpha}_1(1,2) : P(X_2 \leq t,\ X_1+X_2 > t+c) \geq P(X_1 \leq t, X_1+X_2 > t+c)\ \forall\ c > 0,\ t > 0$,

(2) $\bar{\alpha}_2(1,2) : P(Y_1 \leq t,\ Y_1+Y_2 > t+c) \geq P(Y_2 \leq t, Y_1+Y_2 > t+c)\ \forall\ c > 0,\ t > 0$,

(3) $\bar{\beta}_1(1,2) : P(X_1 \leq t) \leq P(X_2 \leq t)\ \forall\ t$ and

(4) $\bar{\beta}_2(1,2) : P(Y_2 \leq t) \leq P(Y_1 \leq t)\ \forall\ t$.

Proof : It is enough to show that

$$P[\min(a+X_1+X_2,\ b+Y_1+Y_2,\ X_1, Y_2) > t]$$

$$\geq P[\min(a+X_1+X_2,\ b+Y_1+Y_2, X_2, Y_1) > t]\ \forall\ a,b \text{ and } t, \tag{3.3.1}$$

under the given four conditions.

We can write

$$P[\min(a+X_1+X_2,\ b+Y_1+Y_2,\ X_1, Y_2) > t]$$

$$= P[a+X_1+X_2 > t,\ X_1 > t]\ P[b+Y_1+Y_2 > t,\ Y_2 > t] \tag{3.3.2}$$

and $P[\min(a+X_1+X_2,\ b+Y_1+Y_2,\ X_2, Y_1) > t]$

$$= P[a+X_1+X_2 > t,\ X_2 > t]\ P[b+Y_1+Y_2 > t, Y_1 > t] \qquad (3.3.3)$$

since X_1 and X_2 are independent of Y_1 and Y_2.

We prove the inequality(3.3.1)by showing that

$$P[a+X_1+X_2 > t, X_1 > t] \geq P[a+X_1+X_2 > t,\ X_2 > t] \qquad (3.3.4)$$

$\forall$ a and t under the condition $\bar{\alpha}_1(1,2)$ and $\bar{\beta}_1(1,2)$ and

$$P[b+Y_1+Y_2 > t,\ Y_2 > t] \geq P[b+Y_1+Y_2 > t,\ Y_1 > t] \qquad (3.3.5)$$

$\forall$ b and t under the conditions $\bar{\alpha}_2(1,2)$ and $\bar{\beta}_2(1,2)$.

Consider the inequality (3.3.4). We can easily see that this inequality holds by the condition $\bar{\beta}_1(1,2)$ for all values of t when $a \geq 0$. Let us assume $a = -c$ when c is a positive number.

We write

$$P[a+X_1+X_2 > t, X_1 > t] = P[X_1+X_2 > t+c,\ X_1 > t]$$

$$= P[X_1+X_2 > t + c] - P[X_1 \leq t, X_1+X_2 > t+c]\ .$$

Similarly,

$$P[a+X_1+X_2 > t, X_2 > t] = P[X_1+X_2 > t + c] - P[X_2 \leq t, X_1+X_2 > t+c]\ .$$

Now it follows by $\bar{\alpha}_1(1,2)$ that

$$P[a+X_1+X_2 > t, X_1 > t] \geq P[a+X_1+X_2 > t, X_2 > t]$$

Similarly by the same arguments it holds under the conditions $\bar{\alpha}_2(1,2)$ and $\bar{\beta}_2(1,2)$ that

$$P[b+Y_1+Y_2 > t,\ Y_2 > t] \geq P[b+Y_1+Y_2 > t,\ Y_1 > t].$$

and hence the lemma.

Corollary 3.3.2 : Let U and V be any two random variables independent of X_1, X_2, Y_1 and Y_2 of Lemma 3.3.1. Then

$$E[\min(U+X_1+X_2, V+Y_1+Y_2, X_1, Y_2)] \geq E[\min(U+X_1+X_2, V+Y_1+Y_2, X_2, Y_1)]$$

under the four conditions given in lemma 3.3.1.

Proof : Proof is similar to that of Corollary 3.2.3.

Now, we consider the nx2 flow shop scheduling problem with processing times following some known distributions. Let

$\alpha_1(i,j)$, $1 \leq i,j \leq n$, $i \neq j$,

represent the condition

$P[A_j \leq t, A_i+A_j > t+c] \geq P[A_i \leq t, A_i+A_j > t+c]$ for $c > 0$ and $t > 0$.

and $\alpha_2(i,j)$ represent the condition

$P[B_i \leq t, B_i+B_j > t+c] \geq P[B_j \leq t, B_i + B_j > t+c]$ for $c > 0$ and $t > 0$.

Let

$\beta_1(i,j)$, $\beta_2(i,j)$, $1 \leq i,j \leq n, i \neq j$,

represent the conditions

$P(A_i \leq t) \leq P(A_j \leq t)$ $\forall$ t and $P(B_j \leq t) \leq P(B_i \leq t)$ $\forall$ t

respectively.

Main Results

In the following theorem, we obtain optimality criteria for minimising the expected makespan of the problem under consideration.

Theorem 3.3.3 : Under the assumptions

(1) for any pair (i,j), $1 \leq i,j \leq n$, $i \neq j$, either the set of conditions $\alpha_1(i,j)$, $\alpha_2(i,j)$, $\beta_1(i,j)$ and $\beta_2(i,j)$ or the set of conditions $\alpha_1(j,i)$, $\alpha_2(j,i)$, $\beta_1(j,i)$ and $\beta_2(j,i)$ hold.

(2) the conditions α_1 and α_2 are transitive, a sequence $p = (p_1', p_2, \dots, p_n)$ satisfying $\alpha_k(p_i, p_{i-1})$ and $\beta_k(p_i, p_{i-1})$ for $k = 1,2$ and $1 < i \leq n$ minimises the expected makespan.

<u>Proof</u> : By applying Corollary 3.3.2 instead of 3.2.3 in Theorem 3 2.5 we can easily prove this result.

Since the conditions $\alpha_k(i,j)$ and $\beta_k(i,j)$ are not easily verifiable, we shall now give optimality criteria in terms of easily verifiable monotone likelih ratio (MLR) conditions that imply $\alpha_k(i,j)$ and $\beta_k(i,j)$. First we shall derive a simple result which is useful in obtaining the required optimality criteria.

<u>Lemma 3.3.4</u> For any non-negative independent r.v.'s, X_1, X_2, Y_1 and Y_2 with densities $f_i(x)$, $f_2(x)$, $g_1(x)$ and $g_2(x)$ respectively, the conditions $\bar{\alpha}_k(1,2)$ and $\bar{\beta}_k(1,2)$, $k = 1,2$ of Lemma 3.3.1 hold when the following MLR conditions hold.

(i) $f_1(x)/f_2(x)$ is nondecreasing in x

(ii) $g_1(x)/g_2(x)$ is nonincreasing in x.

<u>Proof</u> : Since X_1 and X_2 are non-negative r.v.'s, we can write for arbitrary $t \geq 0$ and $c \geq 0$

$$P[X_2 \leq t,\ X_1+X_2 > t+c] - P[X_1 \leq t, X_1+X_2 > t+c]$$

$$= \underset{\substack{x_2 \leq t \\ x_1+x_2 > t+c}}{\int_0^\infty \int_0^\infty} f_1(x_1)f_2(x_2)dx_1dx_2 - \underset{\substack{x_1 \leq t \\ x_1+x_2 > t+c}}{\int_0^\infty \int_0^\infty} f_1(x_1)f_2(x_2)dx_1dx_2$$

$$= \underset{\substack{x_2 \leq t \\ x_1+x_2 > t+c}}{\int_0^\infty \int_0^\infty} f_1(x_1)f_2(x_2)dx_1dx_2 - \underset{\substack{x_2 \leq t \\ x_1+x_2 > t+c}}{\int_0^\infty \int_0^\infty} f_1(x_2)f_2(x_1)dx_1dx_2$$

(by interchanging x_1 and x_2 in the second term)

$$= \int_0^\infty \int_0^\infty [f_1(x_1)f_2(x_2)-f_1(x_2)f_2(x_1)]dx_1dx_2$$
$x_2\le t,\ x_1+x_2>t+c$

$$= \int_0^\infty \int_0^\infty [f_1(x_1(x_1)f_2(x_2)-f_1(x_2)f_2(x_1)]dx_1dx_2$$
$x_2\le t,\ x_1\le t$
$x_1+x_2 > t+c$

$$+ \int_0^\infty \int_0^\infty [f_1(x_1)f_2(x_2)-f_1(x_2)f_2(x_1)]dx_1dx_2$$
$x_2\le t,\ x_1>t$
$x_1+x_2>t+c$

$$\ge \quad 0$$

because the first term is zero whether $t > c$ or not and the second term is nonnegative since $f_1(x_1)/f_2(x_1) \ge f_1(x_2)/f_2(x_2)$ for $x_1 \ge x_2$.

It can be easily seen that the above MLR condition implies $\bar{\beta}_1(1,2)$ also which can be expressed as

$$\int_0^t [f_2(x)-f_1(x)]dx \ \ge\ 0 \quad \forall \quad t \ge 0 .$$

We can prove this implication by contraction. Similarly the MLR condition (ii) implies $\bar{\alpha}_2(1,2)$ and $\bar{\beta}_2(1,2)$.

Remark 3.3.5 : From Lemmas 3.3.1 and 3.3.4, we conclude that MLR conditions (i) and (ii) imply the inequality 3.2.2.

Corollary 3.3.6 Let U and V be any two r.v.'s independent of X_1, X_2, Y_1 and Y_2 of Lemma 3.3.4. The MLR conditions (i) and (ii) imply

$$E[\min(U+X_1+X_2,\ V+Y_1+Y_2,\ X_1,Y_2)] \ge E[\min(U+X_1+X_2, V+Y_1+Y_2, X_2,Y_1)] .$$

Proof is similar to that of corollary 3.2.3.

Let us come back to the flowshop problem under consideration. Suppose $f_i(x)$ $(g_i(x))$, $1 \leq i \leq n$, is the density function of the processing time $A_i(B_i)$. The following theorem gives optimality criteria in terms of MLR conditions.

Theorem 3.3.7 : The sequence $(p_1,p_2,\cdots,p_n)$ minimises the expected makespan when $f_{p_i}(x)/f_{p_{i+1}}(x)$ is nonincreasing and $g_{p_i}(x)/g_{p_{i+1}}(x)$ is nondecreasing in x for $1 \leq i < n$.

Proof : Using corollary 3.3.6 instead of 3.2.3 and noting that the above MLR conditions are transitive, we can easily prove the theorem.

Special cases

We shall now give a list of some cases that can be solved using Theorem 3.3.7. In some of the cases we consider processing times to be normal variables even though normal variables take on negative values. If the mean is sufficiently large and standard deviation is sufficiently small, a normal variable can be considered as a nonnegative r.v. for practical purposes as is done statistical quality control.

Let

$$u(a,x) = \frac{1}{a-s} \quad \text{if} \quad s \leq x \leq a \quad \text{and} \quad 0 \quad \text{otherwise}$$

$$\phi(\mu,x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\{-\frac{1}{2\sigma^2}(x-\mu)^2\} \quad \text{for} \quad -\infty < x < \infty$$

and

$$p(\alpha,x) = \frac{\alpha^r x^{r-1}}{\Gamma(r)} e^{-\alpha x} \quad \text{for} \quad 0 \leq x < \infty.$$

The optimal criteria given in the following table are sufficient conditions for the sequence $(1,2,\ldots,n)$ to be optimal.

Case	$f_i(x)$	$g_i(x)$	Optimal Criteria	
1	$u(a_i;x)$	$u(b_i;x)$	$a_1 \leq a_2 \leq \cdots \leq a_n$ and	$b_1 \geq b_2 \geq \cdots \geq b_n$
2	$u(a_i;x)$	$\phi(v_i;x)$	"	$\nu_1 \geq \nu_2 \geq \cdots \geq \nu_n$
3	"	$p(\beta_i;x)$	"	$\beta_1 \leq \beta_2 \leq \cdots \leq \beta_n$
4	$\phi(u_i;x)$	$u(b_i;x)$	$\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$ and	$b_1 \geq b_2 \geq \cdots \geq b_n$
5	"	$\phi(v_i;x)$	"	$\nu_1 \geq \nu_2 \geq \cdots \geq \nu_n$
6	"	$p(\beta_i;x)$	"	$\beta_1 \leq \beta_2 \leq \cdots \leq \beta_n$
7	$p(\alpha_i;x)$	$u(b_i;x)$	$\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_n$ and	$b_1 \geq b_2 \geq \cdots \geq b_n$
8	"	$\phi(v_i;x)$	"	$\nu_1 \geq \nu_2 \geq \cdots \geq \nu_n$
9	"	β		$\beta_1 \leq \beta_2 \leq \cdots \leq \beta_n$

Case 1	A_i's follow	uniform	and	B_i's	follow uniform
2		"		"	normal
3		"		"	gamma
4	A_i's follow	normal	and	B_i s	follow uniform
5		"		"	normal
6		"		"	gamma
7	A_i's follow	gamma	and	B_i's	follow uniform
8		"		"	normal
9		"		"	gamma

The above optimal criteria can be easily verified using Theorem 3.3.7 For example, when A_i, $1 \leq i \leq n$, follows uniform with interval $[s, a_i]$ and B_i, $1 \leq i \leq n$, follow normal with mean ν_i and standard deviation σ, we can easily verify that $f_i(x)/f_{i+1}(x)$ is nonincreasing in x and $g_i(x)/g_{i+1}(x)$ is non-decreasing in x for $1 \leq i < n$. Now it easily follows from Theorem 3.3.7 that the sequence $(1,2,\ldots,n)$ is optimal.

Remark 3.3 : The optimality criteria of all the theorems in sections 3.2 and 3.3 hold even when the machine B is not available for time $t = z$ where z is a r.v. independent of all A_i's and B_i's.

3.4 Finite Dynamic Programming Formulation

Introduction

In this section, we consider an nx2 stochastic flowshop scheduling problem (P) in which

(1) the processing times A_i's follow some known distributions ,

(2) the processing times B_i's follow exponential distributions,

(3) no passing is allowed and

(4) the objective is to minimise the expected makespan.

We formulate this problem as a Dynamic Programming (DP) problem which we show is numerically tractable when A_i's follow gamma or uniform distributions.

Formulation as a DP Problem

We follow the same notation as in sections 3.2 and 3.3 to denote jobs, machines and processing times. Since the objective is to minimise the expected makespan we do not allow any machine to remain idle as long as there is at least one job waiting for processing before it, that is, immediately after a job is completed on machine A (B) a, job has to be taken for processing (on machine A(B)) from the set of jobs waiting for processing before A(B). In the problem P considered above, we allow decisions only at time $t = 0$ and at the completion time of each job on each machine due to non-preemptive restriction. A decision at a job completion time on machine A(B) is to choose a job, for proce-ssing from the set of jobs waiting for processing before A(B). Because of no-passing assumption the jobs should be processed on machine B in the order in which they are selected for processing on machine A. Therefore, we need to take decisions only at time $t = 0$ and at each job completion time on A.

The state of the flow shop at a job completion time τ on A can be represented by (U,σ) when $U \subseteq N$ is the set of jobs yet to be processed on machines A and $\sigma \subseteq N-U$ a sequence of jobs waiting for processing before machine B at time $t = \tau$. Due to the memoryless property of the exponential distributions of processing times B_i's, the job that is being processed on B at time τ can be taken as the first element of σ. We denote by $E_s(U,\sigma)$ the minimum expected time from τ onwards, required for the entire processing of jobs in U and σ, where $s = |U|$.

Let $\sigma = (i_1, i_2, \ldots, i_\ell)$. We can write

$$E_s(U,\sigma) = \min_{j \in U} [E(A_j) + \sum_{(U',\sigma') \in G_j(U,\sigma)} p^j(U,\sigma;U',\sigma')E_{s-1}(U',\sigma')] \qquad (3.4.1)$$

and

$$E_0(\phi,\sigma) = \sum_{j \in \sigma} E(B_j) \qquad (3.4.2)$$

where

$$G_j(U,\sigma) = \{(U-\{j\}, (i_k, i_{k+1}, \ldots, i_\ell, i_{\ell+1})) | 1 \leq k \leq \ell+1\}, \quad i_{\ell+1} = j$$

and $p^j(U,\sigma;U',\sigma')$ is the probability that the flow shop is in the state (U',σ') at the completion time of job j on machine A when the job j is taken for processing on A in the state (U,σ).

By recursive method we find the value of $E_n(N,\phi)$ which gives the minimum expected makespan and obtain an optimal policy which minimises the expected makespan.

Calculation of $P^j(U,\sigma;U',\sigma')$

It is obvious that

$$P(U,\sigma;U-\{j\},\ \{i_k,i_{k+1},\ldots,i_{\ell+1}\}) = P(\sum_{h=1}^{k-1} B_{i_h} \le A_j < \sum_{h=1}^{k} B_{i_h})$$

for $k = 1$ to ℓ (3.4.3)

and

$$P(U,\sigma;U-\{j\},\{j\}) = P(\sum_{h=1}^{\ell} B_{i_h} \le A_j) . \qquad (3.4.4)$$

We follow the convention $\sum_{h_1}^{h_2} = 0$ for $h_2 < h_1$.

We rewrite the equation (3.4.3) as

$$P(U,\sigma;U-\{j\},(i_k,i_{k+1},\ldots,i_{\ell+1})) = P(\sum_{h=1}^{k-1} B_{i_h} \le A_j)-P(\sum_{h=1}^{k} B_{i_h} \le A_j)$$

for $k = 1$ to ℓ. (3.4.5)

Thus, the difficulty of calculating $p^j(U,\sigma;U',\sigma')$ lies in the calculation of the terms $P(B_{v_1} + B_{v_2} + \ldots + B_{v_\ell} \le A_j)$ for any $v_1,v_2,\ldots,v_\ell \in N$ and $1 \le j \le n$.

The Case of Uniformly Distributed A_j's

Below, we give a procedure for calculating the terms $P(B_{v_1}+B_{v_2}+ \ldots + B_{v_\ell} \le A_j)$ when A_j's follow uniform distributions and $b_i \ne b_j$ for $i \ne j$. Assume that A_j follows uniform distribution with interval $[s_j,c_j]$, $0 \le s_j \le c_j$, and B_j follows exponential distribution with parameter b_j for $1 \le j \le n$ and $b_i \ne b_j$ for $i \ne j$.

Since B_i's are independent exponential random variables with different b_i's, we have

$$P[B_{v_1} + B_{v_2} + \dots + B_{v_\ell} \leq t] = 1 - \sum_{i=1}^{\ell} d_{v_i} e^{-b_{v_i} t} \qquad (3.4.6)$$

where

$$d_{v_i} = \prod_{\substack{j=1 \\ j \neq i}}^{\ell} \frac{bv_j}{bv_j - bv_i} .$$

We can easily see this by finding the Laplace transform of the distribution of $B_{v_1} + B_{v_2} + \dots + B_{v_\ell}$ and splitting it into partial fractions and finding the inverse Laplace transform of each term. Since A_j follows uniform distribution with the interval $[s_j, c_j]$ we have

$$P[B_{v_1} + B_{v_2} + \dots + B_{v_\ell} \leq A_j] = 1 - \frac{1}{c_j - s_j} \sum_{i=1}^{\ell} \frac{d_{v_i}}{b_{v_i}} \{e^{-b_{v_i} s_j} - e^{-b_{v_i} c_j}\} . \qquad (3.4.7)$$

We can easily calculate the value of $P[B_{v_1} + \dots + B_{v_\ell} \leq A_j]$ using the equation (3.4.7).

The Case of A_j's Following Gamma Distributions

Now we give a procedure for calculating the terms $P[B_{v_1} + B_{v_2} + \dots + B_{v_\ell} \leq A_j]$ when A_j's follow Gamma distributions. Assume that A_j, $1 \leq j \leq n$, follows Gamma distribution with density

$$f_j(t) = \frac{a_j^{r_j}}{\Gamma(r_j)} t^{r_j - 1} e^{-a_j t} .$$

Here also we assume that $b_i \neq b_j$ for $i \neq j$.

We write

$$P[B_{v_1} + B_{v_2} + \dots + B_{v_\ell} \le A_j] = \int_0^\infty \{1 - \sum_{i=1}^{\ell} d_{v_i} e^{-b_{v_i} t}\} \frac{a_j^{r_j}}{\Gamma(r_j)} t^{r_j - 1} e^{-a_j t} dt$$

$$= 1 - \sum_{i=1}^{\ell} d_{v_i} \int_0^\infty \frac{a_j^{r_j}}{\Gamma(r_j)} t^{r_j - 1} e^{-(a_j + b_{v_i})t} dt$$

$$= 1 - a_j^{r_j} \sum_{i=1}^{\ell} \frac{d_{v_i}}{(a_j + b_{v_i})^{r_j}}.$$

Using this equation we can easily calculate the value of $P[B_{v_1} + B_{v_2} + \dots + B_{v_\ell} \le A_j]$ for any $v_1, v_2, \dots, v_\ell \in N$, and $1 \le j \le n$.

The Case of A_j's Following Erlang Distributions

We now consider the case where A_j's follow Erlang distributions with $r = 2$. Here, we do not assume $b_i \neq b_j$ for $i \neq j$. Assume that $A_j, 1 \le j \le n$, follows Erlang distribution with density

$$f_j(t) = a_j^2 \, t \, e^{-a_j t} , \quad a_j > 0.$$

We can write

$$P[B_{v_1} + B_{v_2} + \dots + B_{v_\ell} \le A_j] = \int_0^\infty G_\ell(t) \, f_j(t) dt$$

where

$$G_\ell(t) = P[B_{v_1} + B_{v_2} + \dots + B_{v_\ell} \le t \,]$$

i.e.

$$P[B_{v_1} + B_{v_2} + \dots + B_{v_\ell} \le A_j] = \int_0^\infty G_\ell(t) \, a_j^2 \, t \, e^{-a_j t} dt$$

$$= -a_j^2 \frac{d}{ds} G_\ell^*(s) \Big|_{s=a_j}$$

where $G_\ell^*(s)$ is the Laplace transform of $G_\ell(t)$.

Since all B_i's are independent exponential random variables , we have

$$G^*_\ell(s) = \frac{1}{s} \prod_{i=1}^{\ell} \frac{b_{v_i}}{b_{v_i}+s}$$

and consequently

$$- \frac{d}{ds} G^*_\ell(s)\Big|_{s=a_j} = \frac{1}{a_j} [\frac{1}{a_j} + \frac{1}{b_{v_1}+a_j} + \frac{1}{b_{v_2}+a_j} + \dots$$

$$\dots + \frac{1}{b_{v_\ell}+a_j}] \prod_{i=1}^{\ell} \frac{b_{v_i}}{b_{v_i}+a_j} .$$

Therefore

$$P[B_{v_1}+B_{v_2}+\dots+B_{v_\ell} \leq A_j] = a_j[\frac{1}{a_j} + \frac{1}{b_{v_1}+a_j} + \frac{1}{b_{v_2}+a_j} + \dots$$

$$+ \frac{1}{b_{v_\ell}+a_j}] \prod_{i=1}^{\ell} \frac{b_{v_i}}{b_{v_i}+a_j} . \qquad (3.4.8)$$

Using the equations (3.4.8), we can easily calculate the required probabilities.

Remarks 3.4.1 : For the problem P, permutation schedules are the only feasible schedules and the optimal policy (α^*) obtained by dynamic programming technique is better than or at least as good as any permutation schedule. It is easy to see that for / the problem P' obtained from P by relaxing the assumption of no-passing, α^* is a policy which dominates the set of all schedules. In fact, for the problem P' there is no way to minimise the expected makespan even over the set of schedules. It is not known whether α^* is an optimal policy for the problem P' also.

CHAPTER IV

SINGLE-PROCESSOR AND PARALLEL-PROCESSOR STOCHASTIC SCHEDULING PROBLEMS

4.1 Introduction

In this chapter, we first deal with a problem of scheduling tasks with random processing times on parallel processors. The deterministic version of this problem with various objectives has been considered by several authors. For reference,see Mc Naughton (1959), Eastman et. al. (1964) Arthanari and Ramamurthy (1970), Nabeshima (1971) etc. The parallel-processor scheduling problems are very common in practical life. In deterministic case, most of the problems with objectives like minimisation of makespan, flow time, weighted sum of completion times, maximum tardiness etc. are shown to be NP-complete and algorithms are being developed to solve those problems whereas in stochastic case the attention is focussed on deriving optimality criteria for minimising various cost functions. We briefly discuss the work that has been done on stochastic scheduling.

Weber and Nash (1979) have considered the following stochastic scheduling problem. There are n identical spares with random life time for maintenance of a series reliability system. The distribution of life time is assumed to have monotone hazard rate. Each component requires only one spare at a time and the number of components needed to operate the system is a non-decreasing function of time. At time t = 0 a spare may have some age, that is, it might have been used for some time before the start of processing. The objective of this problem is to maximise the expected life of the system. The authors have solved this problem by obtaining a policy of assigning spares to components, which maximises the reliability of the system at each time point. Glazebrook (1979) has considered a problem of scheduling tasks with random processing times on identical parallel processors. In this problem the number of processors available is a non decreasing function of time and processing times of tasks follow geometric distributions. Glazebrook has minimised the expected flow time by scheduling the tasks in non-increasing order of parameters (of the distributions of processing times) and has obtained an optimality criteria for continuous case with exponential processing times when same number of processors are available at any time. Gittins (1981) has considered a problem more general than that of Glazebrook (1979) in which the distributions of processing times have non-decreasing hazard rates. Under certain conditions, Gittins(1981) has shown that the expected flow time would be minimised by scheduling the tasks always in the order of highest hazard rates (non-increasing hazard rates). When the tasks are identical, this result holds even if some of the tasks arrive after the start of processing.

Bruno and Downey (1977) have considered a problem of scheduling tasks with exponential processing times on two identical processors. The objectives considered by the authors are minimisation of expected makespan and expected flow time. No pre-emption is allowed. Formulating this problem as a dynamic programming problem,the authors have shown that the selection of tasks (for processing) in non-decreasing order of parameters (of the distributions of processing times) minimises the expected makespan and the selection of tasks in non-increasing order of parameters minimises the expected makespan. Pinedo and Weiss (1979) have considered the same problem and gave easy and elegant proofs of the above two results. The authors have further obtained optimality criteria for both the objectives when the processing times follow hyper exponential distributions which are mixtures of two fixed exponentials.

Pinedo (1980) has considered a problem of scheduling n spares with exponential life times in a two component parallel system so as to maximise the expected life of the system. Each spare can be put in either position. The decision maker is allowed to take risk, that is, no immediate replacement of a failed component. Pinedo has obtained optimality criteria for some special cases of this problem. Weiss and Pinedo (1980) have considered a very general parallel-processor stochastic scheduling problem. Formulating it as a semi-Markov decision process, they have obtained optimality criteria for two types of cost functions and solved various scheduling problems using these criteria.

In section 4.2, we briefly describe the work of Weiss and Pinedo (1980) and disprove their conjecture by a numerical counter example. We first consider a two-processor (parallel) stochastic scheduling problem in which the processing times of tasks on one processor say A, follow identical exponential distributions and on the other processor, say B, they follow different exponential distributions and obtain optimal policies for minimising expected flow time. Later, we consider a problem which involves 'a' processors of type A and 'b' processors of type B and obtain optimal policies for four cases. Finally, in the last section 4.3 we consider a single-processor stochastic scheduling problem in which the processor is subject to breakdowns and repairs, with the objective of minimising the expected weighted sum of task completion times. Formulating this problem as a semi-Markov decision process, we obtain an optimal policy that minimises the objective under consideration.

4.2 Non-Identical Parallel-Processor Stochastic Scheduling Problems

We first give a brief account of the work of Weiss and Pinedo (1980) without quoting any theorems.

The problem considered by Weiss and Pinedo (1980) can be defined through the following assumptions:

(1) There are n tasks $1,2,\ldots,n$ to be processed and m processors $1,2,\ldots,m$ available for processing at time $t = 0$.

(2) Any processor can process any task but one (task) at a time and any task can be processed by only one processor at a time.

(3) Interruptions in processing and switches of a task from one processor to another are allowed.

(4) The amount of time X_{ij} required by processor i, $1 \leq i \leq m$, to process task j (till completion), $1 \leq j \leq n$, is an exponential random variable with parameter α_{ij}.

(5) There exist (n+m) non negative values $\lambda_1, \lambda_2, \ldots, \lambda_n$ and $s_1, s_2, \ldots, s_m$ such that $\alpha_{ij} = s_i \lambda_j$.

(6) All X_{ij}'s are independent of each other and of the manner in which tasks are assigned to processors.

(7) An uncompleted task j at time t still requires the amount of processing time X_{ij} if it is to be processed on machine i continuously from time t onwards.

(8) The rate of cost at time t is $g(U)$ where U is the set of uncompleted tasks at time t and g is a real valued set function defined on the class of all subsets of the set $\{1,2,\ldots,n\}$.

The objective of this problem is to find a policy that minimises the total expected cost over $[0,\infty)$.

The condition that there exist (n+m) non-negative values $\lambda_1, \lambda_2, \ldots, \lambda_n$ and $s_1, s_2, \ldots, s_m$ such that $\alpha_{ij} = s_i \lambda_j$ for $1 \leq i \leq m$ and $1 \leq j \leq n$ is called separability condition on α_{ij}'s. Under this condition, α_{ij}'s are said to be separable. For any separable α_{ij}'s, we can easily find corresponding λ_j's and s_i's (not uniquely). We call λ_j, $1 \leq j \leq n$, the completion rate of task j and s_i, $1 \leq i \leq m$, the speed of processor i. In the above problem, Weiss and Pinedo (1980) have considered two important policies called SEPT and LEPT. SEPT (LEPT) is a policy which assigns at any moment the task with largest (shortest) completion rate to the processor with highest speed, the task with second largest (shortest) completion rate to the processor with second highest speed and so on.

For the problem defined above, Weiss and Pinedo have mainly considered the following two types of cost function g.

Type I : For every $U \subseteq N$ where $N = 1,2,\ldots,n\}$,

(1) $g(U) \geq 0$ ($g(U) = 0$ for $U = \phi$),

(2) $g(U-\{\ell\}) \geq g(U-\{k\})$ for $k,\ell \in U$ when $\lambda_k \geq \lambda_\ell$ and

(3) $g(U)-g(U-\{k\})-g(U-\{\ell\}) + g(U-\{k,\ell\}) \geq 0$ for $k,\ell \in U$.

Type II: For any $U \subseteq N$, $g(U) = h_{|U|}$ where h_r's $(0 \leq r \leq n)$ satisfy $0 = h_0 \leq h_1 \leq h_2 \ldots \leq h_n$ and

$$\frac{h_1-h_0}{s_1} \geq \frac{h_2-h_1}{s_2} \geq \frac{h_3-h_2}{s_3} \geq \cdots \geq \frac{h_n-h_{n-1}}{s_n}$$

Formulating the above problem as a semi-Markov decision process, the authors have shown that the policy SEPT (LEPT) is optimal for a cost function of type I (II) and/have given seven interesting applications of these optimal policies. Finally, the authors have considered another type of cost function which satisfy, for any $U \subseteq N$, $|U| \geq 2$,

$$\frac{g(U)-g(U-\{k\})}{s_r} \leq \frac{g(U-\{\ell\})-g(U-\{k,\ell\})}{s_{r-1}}$$

where $r = |U|$ and conjectured that for any cost function of this type the policy LEPT is optimal.

We now construct a numerical counterexample to show that the above conjecture does not hold.

Let the set of processors be {1,2} and $N = \{1,2\}$ and let $(\lambda_1,\lambda_2) = (2,2.1)$, $(s_1,s_2) = (11,10)$ and $g(\{1,2\}) = 10$, $g(\{1\}) = 8$, $g(\{2\}) = 9$, $g(\phi) = 0$. We can easily verify that this cost function g is of the last type. The total expected costs T_L and T_S corresponding to the policies LEPT and SEPT are given by

$$T_L = \frac{g(\{1,2\})}{\lambda_1 s_1+\lambda_2 s_2} + [\frac{\lambda_1 s_1}{\lambda_1 s_1+\lambda_2 s_2}] \frac{g(\{2\})}{\lambda_2 s_1} + [\frac{\lambda_2 s_2}{\lambda_1 s_1+\lambda_2 s_2}] \frac{g(\{1\})}{\lambda_1 s_1}$$

$$T_S = \frac{g(\{1,2\})}{\lambda_1 s_2+\lambda_2 s_1} + [\frac{\lambda_2 s_1}{\lambda_1 s_2+\lambda_2 s_1}] \frac{g(\{1\})}{\lambda_1 s_1} + [\frac{\lambda_1 s_2}{\lambda_1 s_2+\lambda_2 s_1}] \frac{g(\{2\})}{\lambda_2 s_1}$$

since $\lambda_1 < \lambda_2$ and $s_1 > s_2$

On substituting the values of λ_1,λ_2, $s_1,s_2,g\{1\}$, $g\{2\}$ and $g\{1,2\}$, we get

$$T_L = 0.6095 \quad \text{and} \quad T_S = 0.6077$$

which imply SEPT is better than LEPT .

Scheduling on Two Non-Identical Processors

In the stochastic scheduling problem considered by Weiss and Pinedo (1980), it has been assumed that the time required by processor i, $1 \leq i \leq m$ to process task j, $1 \leq j \leq n$, is an exponential random variable X_{ij} with parameter $\alpha_{ij} = s_i \lambda_j$ where s_i's and λ_j's are non-negative values. The problem is very difficult to solve if the separability condition, $\alpha_{ij} = s_i \lambda_j$ for $1 \leq i \leq m$ and $1 \leq j \leq n$, is relaxed. For example, consider the case where the processing times satisfy

(1) for any k,ℓ $(1 \leq k,\ell \leq n)$, $\alpha_{ik} \geq \alpha_{i\ell}$ for $1 \leq i \leq m$ if there exists a u, $1 \leq u \leq m$, such that $\alpha_{uk} > \alpha_{u\ell}$,

(2) for any r,s $(1 \leq r,s \leq m)$, $\alpha_{rj} \geq \alpha_{sj}$ for $1 \leq j \leq n$ if there exists a v, $1 \leq v \leq n$, such that $\alpha_{rv} > \alpha_{sv}$.

We have not yet been able to solve this case even for $m = 2$.

In this section, we consider a two-processor (parallel) scheduling problem with the same assumptions as in the problem of Weiss and Pinedo (1980) except the assumption (5) which is replaced by an assumption that $\alpha_{1j} = \mu$ and $\alpha_{2j} = \lambda_j$ for $1 \leq j \leq n$. This assumption means that the processing times of all tasks on one processor, say A, are identically distributed exponential random variables with parameter μ and those on the other processor, say B, are exponential random variables with parameters λ_j's. We have assumed in our problem that interruptions and switches of tasks from one processor to another are allowed at any time. We first assume that they are allowed only at task completion times and formulate the problem as a semi-Markov decision process. Following an approach which is similar to that of Weiss and Pinedo (1980), we obtain (1) a policy that minimises the expected flow time.

Finally, using Markov decision arguments we claim that the optimality of the above policy holds even when interruptions and switches are allowed at any time.

Formulation of the Problem

We formulate the problem as semi-Markov decision process as follows (For reference to semi-Markov decision process, see Ross (1970), Chapter 7).

The state of the system at time t is the set of uncompleted tasks at that time. The set of actions J(U) associated with a state U is given by

$$J(U) = \{f/f = (\alpha,\beta),\ \alpha,\beta \in U,\ \alpha \neq \beta\} \bigcup \{f/f = (0,\beta), \beta \in U\} \bigcup$$
$$\{f/f = (\alpha,0),\ \alpha \in U\} \bigcup \{(0,0)\} .$$

An action $f = (\alpha,\beta)$ is an assignment of tasks α and β to the processors A and B respectively. The 0 in first (second) position indicates that the processor A(B) is not assigned any task. The decision moments are time $t = 0$ and task completion times. The sojourn time in a state U under an action $f = (\alpha,\beta)$ is an exponential random variable with parameter (i) $(\mu + \lambda_\beta)$ if $\alpha,\beta \neq 0$, (ii) λ_β if $\alpha = 0$, $\beta \neq 0$, (iii) μ if $\alpha \neq 0$, $\beta = 0$, (iv) 0 if $\alpha = 0 = \beta$. The probability of transition from state U to state $U - \{\alpha\}$ $(U-\{\beta\})$ under an action $f = (\alpha,\beta)$ is $\mu/(\mu+\lambda_\beta)$ $(\lambda_\beta/(\mu+\lambda_\beta))$. The rate of cost during the sojourn time in state U is $g(U)$. We assume that $g(U) > 0$ for $U \neq \phi$ and $g(\phi) = 0$. The discount factor is zero.

For this semi-Markov decision process, it is enough to consider only stationary policies in order to minimise the total expected cost over $[0,\infty)$. For reference, see Ross (1970). A stationary policy is a policy in which, for any state U, a particular action $f \in J(U)$ will be taken whenever the system visits the state U. Since the process is time homogeneous under a stationary policy π the total expected cost over time $[t,\infty)$ under π can be denoted by $G_\pi(U)$, where U is the state of the system at time t. Let $G_{f'/\pi}(U)$ represent the expected cost over time $[t,\infty)$ when action f' is taken in state U at time t and π is followed later on. For this semi-Markov decision process, the state space is finite and under any policy with finite total expected cost, the system would be absorbed in the state ϕ with probability one and time till absorption has finite expectation. So the following theorem of Weiss and Pinedo (1980), which is originally due to Strauch (1966), holds for our process also.

Theorem 4.2.1 (Weiss and Pinedo (1980) : (i) There exists a stationary optimal policy π. (ii) A stationary policy π is optimal if for all U N,

$$G_\pi(U) = \min_{f' \in J(U)} G_{f'/\pi}(U) \quad . \qquad (4.2.1)$$

Below, we consider a special case with the cost structure $g(U) = |U|$ for $U \subseteq N$ and we obtain an optimal policy that minimises the total expected cost. Without loss of generality we assume that $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

Optimal policy when $g(U) = |U|$ for $U \subseteq N$.

Let π be a stationary policy in which actions are taken at decision moments as follows:

(i) When there are atleast two uncompleted tasks, the task with smallest λ and the task with largest λ are assigned to processors A and B respectively.

(ii) When there is only one uncompleted task, say j, it is assigned to processor A(B) if $\mu > \lambda_j (\mu \leq \lambda_j)$.

For an arbitrary fixed subset U of N, we use the following notation: $G = G_\pi(U)$, $G_k = G_\pi(U-\{k\})$, $G_{k\ell} = G_\pi(U-\{k,\ell\})$ and we rename the tasks in U as $1,2,\ldots,r$ $(= |U|)$ such that $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_r$. We can write

$$G = \frac{r+\mu G_1+\lambda_r G_r}{\mu + \lambda_r} , \qquad (4.2.2)$$

$$G_k = \frac{(r-1)+\mu G_{k1}+\lambda_r G_{kr}}{\mu + \lambda_r} \quad \text{for} \quad k \neq 1,r, \qquad (4.2.3)$$

$$G_1 = \frac{(r-1)+\mu G_{21}+\lambda_r G_{r1}}{\mu + \lambda_r} \qquad (4.2.4)$$

and

$$G_r = \frac{(r-1)+\mu G_{r1}+\lambda_{r-1} G_{rr-1}}{\mu + \lambda_{r-1}} \qquad (4.2.5)$$

Our aim is to show that the above policy π minimises the total expected cost. For this purpose, we first prove the following theorem.

<u>Theorem 4.2.2</u> : π satisfies, for an arbitrary fixed $U \subseteq N$,

$$G-G_j \geq 0 \quad , \tag{4.2.6}$$

$$G_j \geq G_1 \tag{4.2.7}$$

and

$$\lambda_r(G-G_r) \geq \lambda_j(G-G_j) \geq \lambda_1(G-G_1) \tag{4.2.8}$$

for $1 \leq j \leq r$.

<u>Proof</u> : We prove these inequalities by induction on r, the size of U. For r = 2, we have $U = \{1,2\}$,

$$G = (2+\mu G_1+\lambda_2 G_2)/(\mu+\lambda_2) \ ,$$

$$G_1 = 1/\max(\mu, \lambda_2) \text{ and } G_2 = 1/\max(\mu, \lambda_1) .$$

We can easily verify that $G-G_j \geq 0$ for $j = 1,2$ and $G_2 \geq G_1$ and

$$\lambda_2(G-G_2)-\lambda_1(G-G_1) \geq 0 \ .$$

Assume that the inequalities (4.2.6), (4.2.7) and (4.2.8) hold for $r \leq k-1$. We now show that they hold for $r = k$ also. We have

$$G_r-G_1 = \frac{(r-1)+\mu G_{1r}+\lambda_{r-1}G_{r-1r}}{\mu + \lambda_{r-1}} - \frac{(r-1)+\mu G_{12}+\lambda_r G_{1r}}{\mu + \lambda_r}$$

$$= [(\lambda_r-\lambda_{r-1})(r-1)+\mu(\mu+\lambda_r)G_{1r}+\lambda_{r-1}(\mu+\lambda_r)G_{r-1r}$$
$$- \mu(\mu+\lambda_{r-1})G_{12}-\lambda_r(\mu+\lambda_{r-1})G_{1r}]/(\mu+\lambda_r)(\mu+\lambda_{r-1})$$

$$\geq [(\lambda_r-\lambda_{r-1})(r-1)+\mu(\mu+\lambda_r)G_{1r}+\lambda_{r-1}(\mu+\lambda_r)G_{1r}$$
$$- \mu(\mu+\lambda_{r-1})G_{1r}-\lambda_r(\mu+\lambda_{r-1})G_{1r}]/(\mu+\lambda_r)(\mu+\lambda_{r-1})$$

(since $G_{12} \leq G_{1r} \leq G_{r-1r}$ by induction hypothesis)

$$\geq \ 0 \ .$$

For $2 \leq j \leq r-1$,

$$\begin{aligned} G_j - G_1 &= \{(r-1)+\mu G_{1j}+\lambda_r G_{jr} - ((r-1)+\mu G_{12}+\lambda_r G_{1r})\}/(\mu+\lambda_r) \\ &= \{\mu(G_{1j}-G_{12}) + \lambda_r(G_{jr}-G_{1r})\}/(\mu+\lambda_r) \\ &\geq 0 \quad \text{by induction hypothesis.} \end{aligned}$$

Therefore $G_j \geq G_1$ for $1 \leq j \leq r$.

Now, we show that $\lambda_r(G-G_r) \geq \lambda_j(G-G_j) \geq \lambda_1(G-G_1)$ for $1 \leq j \leq r$.

For $1 < j < r$, we have

$$\begin{aligned} (\mu+\lambda_r)[\lambda_r(G-G_r)-\lambda_j(G-G_j)] &= (\lambda_r-\lambda_j)(\mu+\lambda_r)G-\lambda_r(\overline{\lambda_r-\lambda_{r-1}} + \overline{\lambda_{r-1}+\mu})G_r + \lambda_j(\mu+\lambda_r)G_j \\ &= (\lambda_r-\lambda_j)[r+\mu G_1+\lambda_r G_r] - \lambda_r(\lambda_r-\lambda_{r-1})G_r \\ &- \lambda_r(\overline{r-1} +\mu G_{1r}+\lambda_{r-1}G_{r-1r}) +\lambda_j(r-1+\mu G_{1j}+\lambda_r G_{jr}) \\ &= \lambda_r-\lambda_j+\lambda_r[\lambda_{r-1}(G_r-G_{rr-1})-\lambda_j(G_r-G_{rj}) \\ &\qquad + \mu[\lambda_r(G_1-G_{1r})-\lambda_j(G_1-G_{1j})] \\ &\geq 0 \text{ by induction hypothesis.} \end{aligned}$$

Similarly, for $1 < j < r$

$$\begin{aligned} (\mu+\lambda_r)[\lambda_j(G-G_j)-\lambda_1(G-G_1)] &= (\lambda_j-\lambda_1)[r+\mu G_1+\lambda_r\ G_r]-\lambda_j[\overline{r-1} +\mu G_{1j} +\lambda_r\ G_{jr}] \\ &\qquad + \lambda_1[\overline{r-1} +\mu G_{12} +\lambda_r\ G_{1r}] \\ &= (\lambda_j-\lambda_1) + \lambda_r[\lambda_j(G_r-G_{rj})-\lambda_1(G_r-G_{r1})] \\ &\qquad + \mu[\lambda_j(G_1-G_{1j}) - \lambda_1(G_1-G_{12})] \\ &\geq (\lambda_j-\lambda_1) + \lambda_r[\lambda_j(G_r-G_{rj})-\lambda_1(G_r-G_{r1})] \\ &\qquad + \mu[\lambda_j(G_1-G_{1j})-\lambda_2(G_1-G_{12})] \end{aligned}$$

(since $-\lambda_1 \geq -\lambda_2$ and $G_1 \geq G_{12}$ by induction hypothesis)

$\geq$ 0 by induction hypothesis .

Therefore $\lambda_r(G-G_r) \geq \lambda_j(G-G_j) \geq \lambda_1(G-G_1)$ for $1 \leq j \leq r$.

Now, we show that $G-G_j \geq 0$ for $1 \leq j \leq r$.

For $1 < j < r$

$$G-G_j = \frac{r+\mu G_1+\lambda_r G_r}{\mu+\lambda_r} - \frac{(r-1)+\mu G_{1j}+\lambda_r G_{jr}}{\mu+\lambda_r}$$

$$= [1+\mu(G_1-G_{1j}) + \lambda_r(G_r-G_{rj})]/(\mu+\lambda_r)$$

$$> 0 \text{ by induction hypothesis.}$$

We have $\quad (\mu+\lambda_r)G = r+\mu G_1+\lambda_r G_r$

and $\quad (\mu+\lambda_{r-1})G_r = (r-1) + \mu G_{1r} + \lambda_{r-1} G_{r-1\,r}$.

Subtracting the latter equation from the former one, we get

$$(\mu+\lambda_r)(G-G_r) = -(\lambda_r-\lambda_{r-1})G_r+1 +\mu(G_1-G_{1r})+\lambda_r G_r-\lambda_{r-1}G_{r-1r}$$

$$= \lambda_{r-1}(G_r-G_{r-1r}) + 1+\mu(G_1-G_{1r})$$

$$> 0 \text{ by induction hypothesis .}$$

Therefore $G-G_j \geq 0$ for $1 \leq j \leq r$.

Using theorems 4.2.1 and 4.2.2, we show below that the policy π is optimal.

Theorem 4.2.3 : The policy π minimises the total expected cost.

Proof : Consider an arbitrary fixed subset U of N. In order to prove the equation (4.2.1) for this U, we compare all the actions in J(U) with the action $f = (1,r)$ choosen by π in the state U. First we consider the actions of type $f' = (\alpha,\beta)$ $\alpha,\beta \neq 0$. We represent $G_{f'/\pi}(U)$ by G'. Then we can write

$$G' = (r+\mu G_\alpha+\lambda_\beta G_\beta)/(\mu+\lambda_\beta)$$

and $\quad G'-G = [r-\{\mu(G-G_\alpha)+\lambda_\beta(G-G_\beta)\}]/(\mu+\lambda_\beta)$

$$\geq [r-\{\mu(G-G_1) + \lambda_r(G-G_r)\}]/(\mu+\lambda_\beta)$$

$$= 0$$

since the above inequality holds by theorem 4.2.2 and the equality holds because $(\mu+\lambda_r)G = r + \mu G_1 + \lambda_r G_r$. Thus $G' \geq G$ for $f' = (\alpha,\beta)$.

Similarly $G' \geq G$ for $f' = (0,\beta)$ or $(\alpha,0) \in J(U)$. Therefore $G' \geq G$ for $f' \in J(U)$. Hence the theorem.

It is obvious that the total expected cost gives the expected flow time. Therefore the policy π minimises the expected flow time. In the above semi-Markov decision process we allow decisions at times $0 < T_1 < T_2 \ldots$ where T_i's are transition epochs (job completion times in other words). If we allow decisions at times $0 < \delta < 2\delta < \ldots \leq T_1 < (T_1 + \delta) < (T_1 + 2\delta) < \ldots \leq T_2 < (T_2 + \delta) < (T_2 + 2\delta) < \ldots$, however small δ is, the process is still a semi-Markov decision process for which

(1) the set of stationary policies is the same as above,

(2) the theorem 4.2.1 holds

(3) the policy π minimises the total expected cost. This means that even if we allow interruptions and switches of tasks (from one processor to another) at times $0,\delta,2\delta,\ldots, T_1, T_1 + \delta, T_1+2\delta,\ldots,T_2,T_2+\delta,\ldots$ the policy π minimises the expected flow time.

$\geq [r-$~~$\{\mu(G-G_1)+\lambda_r(G-G_r)\}]/(\mu+$~~$\lambda_\beta)$

$= \;\; 0$

since the above inequality holds by theorem 4.2.2 and the equality holds because $(\mu+\lambda_r)G = r + \mu G_1 + \lambda_r G_r$. Thus $G' \geq G$ for $f' = (\alpha,\beta)$.

Similarly $G' \geq G$ for $f' = (0,\beta)$ or $(\alpha,0) \in J(U)$. Therefore $G' \geq G$ ~~for $f' \in J(U)$. Hence the theorem.~~

<u>Problem involving two or more processors under certain conditions</u>

We shall now consider a problem P' which is more general than the above two non-identical processors problem in the sense that there are'a'processors of type A and'b'processors of type B instead of one of each type. Two objectives (1) minimisation of expected flowtime and (2) minimisation expected makespan are considered for the problem. We shall obtain optimal policies for the following four cases.

Case	a	b	objective	condition
I	2	1	to minimise EFT	$\mu \leq \min \lambda_j$
II	m-1	1	to minimise EMS	$\mu \geq \max \lambda_j$
III	1	2	to minimise EMS	$\mu \geq \max \lambda_j$
IV	1	m-1	to minimise EFT	$\mu \leq \min \lambda_j$

EFT - Expected Flowtime, EMS-Expected Makespan

First we assume that pre-emptions and switches are allowed only at task completion times and obtain optimal policies for each of the above four cases. The decision moments are $t = 0$ and the task completion times.

<u>Case I</u> : $a = 2$, $b = 1$, $\mu \leq \min \lambda_j$ and the objective is to minimise the expected flowtime.

Let π_1 be a policy in which at each decision moment

(i) when there are atleast three uncompleted tasks the first two tasks in the nondecreasing order of λ are assigned to type A processors and the last task in the same order assigned to type B processor.

(ii) When there are only two uncompleted tasks, the task with smaller value of λ is assigned to one of the type A processors and the task with larger value of λ assigned to the type B processor.

(iii) When there is a single uncompleted task, it is assigned to the type B processor.

Let $G_{\pi_1}(U)$ represent the expected flowtime under the policy π_1 from time t onwards when U is the set of uncompleted tasks at time t (we have this representation since the processing times are exponential). Let $G = G_{\pi_1}(N)$, $G_k = G_{\pi_1}(N-\{k\})$ and $G_{k\ell} = G_{\pi_1}(N-\{k,\ell\})$. Then we have

$$G = \frac{n+\mu(G_1+G_2) + \lambda_n G_n}{2\mu+\lambda_n} \qquad 4.2.9$$

or equivalently

$$\mu(G-G_1)+\mu(G-G_2)+\lambda_n(G-G_n) = n \qquad 4.2.10$$

for $n \geq 3$,

$$G = \frac{2+\mu G_1+\lambda_2 G_2}{\mu+\lambda_2} \quad \text{for} \quad n = 2 \qquad 4.2.11$$

and

$$G = 1/\lambda_1 \quad \text{for} \quad n = 1 \qquad 4.2.12$$

Theorem 4.2.4

$$G \geq G_j \quad \text{for} \quad j \in N \qquad 4.2.13$$

$$G_j \geq G_i \quad \text{for} \quad i,j \in N,\ i < j \qquad 4.2.14$$

and

$$\lambda_n(G-G_n) \geq \lambda_j(G-G_j) \quad \text{for} \quad j \in N \qquad 4.2.15$$

Proof : We prove this by induction on n. For n = 2, the inequalities hold by Theorem 4.2.2. Let n = 3. Then we have

$$(2\mu+\lambda_3)[\lambda_3(G-G_3)-\lambda_2(G-G_2)]$$

$$= (2\mu+\lambda_3)(\lambda_3-\lambda_2)G-\lambda_3(\mu+\lambda_2)G_3-\lambda_3(\mu+\lambda_3-\lambda_2)G_{\,}+\lambda_2(\mu+\lambda_3)G_2+\mu\lambda_2G_2$$

$$= (\lambda_3-\lambda_2)[3+\mu G_1+\mu G_2+\lambda_3G_3]-\lambda_3[2+\mu G_{31}+\lambda_2G_{32}]$$

$$+ \lambda_2[2+\mu G_{21}+\lambda_3G_{23}]-\lambda_3(\mu+\lambda_3-\lambda_2)G_3+\mu\lambda_2G_2$$

$$= (\lambda_3-\lambda_2) +\mu[(\lambda_3-\lambda_2)G_1-\lambda_3G_{13}+\lambda_2G_{12}]-\mu\lambda_3(G_3-G_2)$$

$$\geq (\lambda_3-\lambda_2) - \mu\lambda_3(G_3-G_2)$$

(since the inequality 4.2.15 holds for n = 2)

$$\geq 0$$

Proof of the inequality $(\lambda_3-\lambda_2)-\mu\lambda_3(G_3-G_2) \geq 0$ is given Appendix A.

Thus $\lambda_3(G-G_3) \geq \lambda_2(G-G_2)$.

To show the inequality $\lambda_2(G-G_2) \geq \lambda_1(G-G_1)$, we have

$$(2\mu+\lambda_3)[\lambda_2(G-G_2)-\lambda_1(G-G_1)]$$

$$= (\lambda_2-\lambda_1)(2\mu+\lambda_3)G-\lambda_2(\mu+\lambda_3)G_2-\mu\lambda_2G_2+\lambda_1(\mu+\lambda_3)G_1+\mu\lambda_1G_1$$

$$= (\lambda_2-\lambda_1)[3+\mu G_1+\mu G_2+\lambda_3G_3]-\lambda_2[2+\mu G_{21}+\lambda_3G_{23}]$$

$$+ \lambda_1[2+\mu G_{12}+\lambda_3G_{13}]-\mu\lambda_2G_2+\mu\lambda_1G_1$$

$$= (\lambda_2-\lambda_1)+\lambda_3[(\lambda_2-\lambda_1)G_3-\lambda_2G_{32}+\lambda_1G_{31}]$$

$$+ \mu[(\lambda_2-\lambda_1)(G_1+G_2)-\lambda_2G_2+\lambda_1G_1-\lambda_2G_{21}+\lambda_1G_{12}]$$

$$\geq (\lambda_2-\lambda_1) + \mu[\lambda_2G_1-\lambda_1G_2-\lambda_2G_{21}+\lambda_1G_{12}]$$

(since the inequality 4.2.15 holds for n = 2)

$$\geq \mu[\lambda_2(G_1-G_{12})-\lambda_1(G_2-G_{12})]$$

$$\geq 0.$$

Proof of the inequality $\lambda_2(G_1-G_{12})-\lambda_1(G_2-G_{12}) \geq 0$ is shown in Appendix B. Therefore the inequality 4.2.15 holds for $n = 3$. To show $G_3 \geq G_2$. Consider the following two equations

$$(\mu+\lambda_2)G_3 = 2 + \mu G_{31} + \lambda_2 G_{32}$$

$$(\mu+\lambda_3)G_2 = 2 + \mu G_{21} + \lambda_3 G_{23}\ .$$

From these equations, we get

$$\begin{aligned}(\mu+\lambda_2)(G_3-G_2) &= (\lambda_3-\lambda_2)G_2-(\lambda_3-\lambda_2)G_{32}+\mu(G_{31}-G_{21}) \\ &= (\lambda_3-\lambda_2)(G_2-G_{32})+\mu(G_{31}-G_{21}) \\ &\geq 0\end{aligned}$$

since the inequalities 4.2.13 and 4.2.14 hold for $n = 2$. Thus $G_3 \geq G_2$. Similarly $G_2 \geq G_1$. Therefore the inequalities (6) hold for $n = 3$. We have

$$\begin{aligned}G-G_1 &= 3 + \mu(G_2-G_1) + \lambda_3(G_3-G_1) \\ &> 0\ .\end{aligned}$$

Now it easily follows by 4.2.15 that $G-G_3 \geq 0$ which implies $G-G_2 \geq 0$ by (6). Therefore the inequalities 4.2.13, 4.2.14 and 4.2.15 hold for $n = 3$.

Assume that the inequalities 4.2.13, 4.2.14 and 4.2.15 hold for $n = k (\geq 3)$. We now show that it is true for $n = k+1$ also. Let $n = k+1$. First we prove the inequalities 4.2.14. We have for $3 \leq i < n$

$$(2\mu+\lambda_{n-1})G_n = (n-1)+\mu G_{n1}+\mu G_{n2} + \lambda_{n-1} G_{nn-1}$$

$$(2\mu+\lambda_n)G_i = (n-1) + \mu G_{i1} + \mu G_{i2} + \lambda_n G_{in}.$$

From these equations, we can write

$$\begin{aligned}(2\mu+\lambda_{n-1})(G_n-G_i) &= (\lambda_n-\lambda_{n-1})G_i + \mu(G_{n1}-G_{i1}) \\ &\quad + \mu(G_{n2}-G_{i2})+\lambda_{n-1}G_{nn-1}-\lambda_n G_{in} \\ &\geq (\lambda_n-\lambda_{n-1})G_i + \lambda_{n-1}G_{nn-1}-\lambda_n G_{in} \\ &\geq (\lambda_n-\lambda_{n-1})G_i + \lambda_{n-1}G_{i\,n-1} - \cdot\ \lambda_n G_{in} \\ &\geq 0 .\end{aligned}$$

The above three inequalities follow from induction hypothesis. Thus $G_n \geq G_i$ for $i \geq 3$. On the same lines, we can show that $G_j \geq G_i$ for $3 \leq i < j < n$, $G_j \geq G_2$ for $j \geq 3$ and $G_2 \geq G_1$. Therefore the inequalities 4.2.14 hold. For $3 \leq j < n$, we have

$$\begin{aligned}&(2\mu+\lambda_n)[\lambda_n(G-G_n)-\lambda_j(G-G_j)]= \\ &= (\lambda_n-\lambda_j)(2\mu+\lambda_n)G-\lambda_n(2\mu+\lambda_{n-1})G_n+\lambda_j(2\mu+\lambda_n)G_j-\lambda_n(\lambda_n-\lambda_{n-1})G_n \\ &= (\lambda_n-\lambda_j)(n+\mu G_1+\mu G_2+\lambda_n G_n)-\lambda_n(n-1+\mu G_{n1}+\mu G_{n2} \\ &\quad + \lambda_{n-1}G_{nn-1}) + \lambda_j(\overline{n-1} + \mu G_{j1}+\mu G_{j2}+\lambda_n G_{jn})-\lambda_n(\lambda_n-\lambda_{n-1})G_n \\ &= (\lambda_n-\lambda_j)+\mu[(\lambda_n-\lambda_j)G_1-\lambda_n G_{1n}+\lambda_j G_{ij}] + \mu[(\lambda_n-\lambda_j)G_2 \\ &\quad - \lambda_n G_{2n}+\lambda_j G_{2j}] + \lambda_n[(\lambda_{n-1}-\lambda_j)G_n-\lambda_{n-1}G_{nn-1}+\lambda_j G_{nj}] \\ &\geq 0\end{aligned}$$

by induction hypothesis.

Similarly

$$(2\mu+\lambda_n)[\lambda_n(G-G_n)-\lambda_2(G-G_2)]$$

$$= (\lambda_n-\lambda_2)+\mu[(\lambda_n-\lambda_2)G_1-\lambda_n G_{1n}+\lambda_2 G_{12}]+\lambda_n[(\lambda_{n-1}-\lambda_2)G_n$$

$$-\lambda_{n-1}G_{nn-1}+\lambda_2 G_{n2}]+\mu[(\lambda_n-\lambda_2)G_2-\lambda_n G_{2n}+\lambda_2 G_{23}]$$

$$\geq \mu[(\lambda_n-\lambda_2)G_2-\lambda_n G_{2n}+\lambda_2 G_{23}]$$

(by induction hypothesis)

$$= \mu[\lambda_n(G_2-G_{2n})-\lambda_2(G_2-G_{23})]$$

$$\geq \mu[\lambda_n(G_2-G_{2n})-\lambda_3(G_2-G_{23})]$$

$$\geq 0$$

by induction hypothesis

On exactly the same lines as above, we can see that $\lambda_n(G-G_n) \geq \lambda_1(G-G_1)$. Therefore the inequalities 4.2.15 hold. We have

$$G-G_1 = [n+\mu(G_2-G_1) + \lambda_n(G_n-G_1)]/(2\mu+\lambda_n) \geq 0$$

since G_n, $G_2 \geq G_1$. Now $G-G_n \geq 0$ by inequalities (7). Since $G_n \geq G_j$ for all j, it follows that $G-G_j \geq 0$ for all j. Hence the theorem.

Using Theorem 4.2.4 we shall now show that the policy π_1 minimises the expected flowtime.

Theorem 4.2.5: The policy π_1 minimises the expected flowtime.

Proof : We prove this theorem also by induction on n. For n = 1, the policy π_1 trivially minimises the expected flowtime. For n = 2, we have initially the following set (J) of actions to choose. $J=\{(V,j)\,|\,j\in N, V\subsetneq N-\{j\}\}\cup\{(V,0)\,|\,V\subseteq N, V \neq \phi\}$. An action (V,j) is an assignment of task j to the type B processor and the task in V to one of the type A processors. In an action (V,0), tasks in V are assigned to the type A [illegible] type B processor.

Consider an action $f' = (V,j)$. Let G' represent the expected flowtime when the action f' is chosen at time $t = 0$ and the policy π_1 is followed later on. Then we have

$$G' = \frac{2 + \mu \sum_{i \epsilon V} G_i + \lambda_j G_j}{|V|\mu + \lambda_j} ,$$

and

$$G'-G = \frac{2-\{\mu \sum_{i \epsilon V} (G-G_i) + \lambda_j (G-G_j)\}}{|V|\mu + \lambda_j}$$

$$\geq \frac{2-\{\mu(G-G_1)+\lambda_2(G-G_2)\}}{|V|\mu + \lambda_j}$$

(by inequalities 4·2·13, 4·2·14 and 4·2·15)

$$= 0$$

since $(\mu+\lambda_2)G = 2+\mu G_1+\lambda_2 G_2$. Thus $G' \geq G$ for $f' = (V,j)$. Similarly $G' \geq G$ for $f' = (V,0)$. Therefore $G' \geq G$ for any $f' \epsilon J$, that is, the policy π_1 is optimal.

Suppose π_1 is optimal for $n \leq k$ where k is an integer greater than or equal to 2. We now show that π_1 is optimal for $n = k+1$ also. Let $n = k+1$. The set of actions that can be taken at first decision moment $t = 0$ is

$$J = \{(V,j) | j \epsilon N, V \subseteq N-\{j\}, |V| \leq 2\}$$

$$\bigcup \{(V,0) | V \subseteq N, 1 \leq |V| \leq 2\}.$$

Since π_1 is optimal for $n \leq k$, we follow π_1 from second decision moment onwards, whatever action is chosen at time $t = 0$, in order to minimise the expected flowtime. So, it is now enough to show that π_1 is better than (atleast as good as) any policy that coincides with π_1 from second decision moment onwards. Consider a policy π' which chooses an action $f' = (V,j)$ in

J at first decision moment and coincides with π_1 later on. Let G' represent the expected flowtime for the policy π'. Then we have

$$G' = \frac{n + \mu \sum_{i \in V} G_i + \lambda_j G_j}{|V|\mu + \lambda_j}$$

and

$$G'-G = \frac{n - \{\mu \sum_{i \in V} (G-G_i) + \lambda_j (G-G_j)\}}{|V|\mu + \lambda_j}$$

$$\geq \frac{n - \{\mu(G-G_1) + \mu(G-G_2) + \lambda_n (G-G_n)\}}{|V|\mu + \lambda_j}$$

$$= 0.$$

The above inequality holds by Theorem ~~1~~ 4.2.4 and the last equality holds by 4.2.10. Thus $G' \geq G$. Similarly we can see that $G' \geq G$ when $f' = (V,0)$. Therefore π_1 is atleast as good as any policy that coincides with π_1 from second decision onwards. Hence π_1 is optimal.

<u>Case II</u> : $a = m-1$, $b = 1$, $\mu \geq \max_j \lambda_j$ and the objective is to minimise the expected makespan.

Let π_2 be a policy in which at each decision moment we assign, among the set (U) of uncompleted tasks,

(i) the first (m-1) tasks in the nondecreasing order of λ to A type processors and the last task in the same order to the B type processor when $|U| \geq m$

(ii) all the tasks to A type processors when $|U| \leq m-1$.

Let G, G_k and $G_{k\ell}$ represent same terms as earlier with the exception that π_1 is replaced by π_2 and flowtime by makespan.

$$G = [1+\mu \sum_{i=1}^{m-1} G_i + \lambda_n G_n]/[(m-1)\mu+\lambda_n] \text{ if } n \geq m$$

and

$$G = \frac{1}{\mu}\left[\frac{1}{n} + \frac{1}{n-1} + \dots + \frac{1}{2} + 1\right] \text{ otherwise.}$$

Theorem 4.2.6: $G \geq G_j$ for $1 \leq j \leq n$ 4.2.16

$G_j \geq G_i$ for $1 \leq i \leq m-1 < j \leq n$ if $n \geq m$ 4.2.17

$\lambda_n(G-G_n) \geq \lambda_j(G-G_j)$ for $1 \leq j < n$ 4.2.18

Proof : For $1 \leq n \leq m-1$, it is easy to verify $G_j = G_i$ for $i \neq j$, $G-G_j = \frac{1}{\mu n}$ for $1 \leq j \leq n$ and $\lambda_n(G-G_n) \geq \lambda_j(G-G_j)$ for $1 \leq j < n$. Let $n = m$. Then we can easily see that $G_j = G_i$ for $i \neq j$, $G-G_j = 1/\{\lambda_n+(n-1)\mu\}$ for $1 \leq j \leq n$ and consequently $\lambda_n(G-G_n) \geq \lambda_j(G-G_j)$ for $1 \leq j < n$. Thus the inequalities 4.2.16, 4.2.17 and 4.2.18 hold for $n \leq m$.

By induction on n, we shall show that the inequalities hold for $n \geq m$. Suppose the inequalities hold for $n = k$ where k is an integer greater than or equal to m. We now show that it is true for $n = k+1$ also. Let $n = k+1$. Then we can write for $1 \leq j \leq m-1$

$$[(m-1)\mu+\lambda_n][(\lambda_n-\lambda_j)G-\lambda_nG_n + \lambda_jG_j]$$

$$= (\lambda_n-\lambda_j)[(m-1)\mu+\lambda_n]G-\lambda_n[(m-1)\mu+\lambda_{n-1}]G_n-\lambda_n(\lambda_n-\lambda_{n-1})G_n+\lambda_j[(m-1)\mu+\lambda_n]G_j$$

$$= (\lambda_n-\lambda_j)[1 + \mu\sum_{i=1}^{m-1} G_i + \lambda_nG_n]-\lambda_n[1+\mu \sum_{i=1}^{m-1} G_{ni} + \lambda_{n-1}G_{nn-1}]$$

$$+ \lambda_j[1+\mu \sum_{\substack{i=1\\i\neq j}}^{m} G_{ji} + \lambda_n G_{jn}]- \lambda_n(\lambda_n-\lambda_{n-1})G_n$$

$$= \mu\sum_{\substack{i=1\\i\neq j}}^{m-1} [(\lambda_n-\lambda_j)G_i-\lambda_nG_{in}+\lambda_jG_{ij} +\lambda_n[(\lambda_{n-1}-\lambda_j)G_n-\lambda_{n-1}G_{nn-1}$$

$$+ \lambda_jG_{nj}] + \mu[(\lambda_n-\lambda_j)G_j-\lambda_nG_{jn} + \lambda_j G_{jm}]$$

$$\geq \mu[(\lambda_n-\lambda_j)G_j-\lambda_nG_{jn} + \lambda_jG_{jm}]$$

$$= \mu[\lambda_n(G_j-G_{jn})-\lambda_j(G_j-G_{jm})]$$

$$\geq \mu[\lambda_n(G_j-G_{jn})-\lambda_m(G_j-G_{jm})]$$

$$\geq 0.$$

The last three inequalities hold by induction hypothesis. Thus $\lambda_n(G-G_n) \geq \lambda_j(G-G_j)$ for $1 \leq j \leq m-1$. Following the above approach, we can easily see that $\lambda_n(G-G_n) \geq \lambda_j(G-G_j)$ for $m \leq j < n$.

For $1 \leq j \leq m-1$, we have

$$[(m-1)\mu+\lambda_{n-1}]G_n = 1 + \mu \sum_{i=1}^{m-1} G_{ni} + \lambda_{n-1}G_{nn-1}$$

$$[(m-1)\mu+\lambda_n]G_j = 1 + \mu \sum_{\substack{i=1 \\ i\neq j}}^{m} G_{ji} + \lambda_n G_{jn} .$$

From these equations, we get

$$[(m-1)\mu+\lambda_{n-1}](G_n-G_j) = (\lambda_n-\lambda_{n-1})G_j-\lambda_n G_{jn}+\lambda_{n-1}G_{nn-1}+\mu[\sum_{\substack{i=1 \\ i\neq j}}^{m-1}(G_{ni}-G_{ji})+(G_{jn}-G_{jm})$$

$$\geq 0$$

by induction hypothesis.

On the same lines we can easily see that

$$G_n \geq G_j \qquad \text{for} \qquad m \leq j < n \qquad\qquad 4.2.19$$

and $\qquad G_j \geq G_i \qquad$ for $\qquad 1 \leq i \leq m-1 < j < n.$

We have

$$G-G_1 = \frac{1+\mu \sum_{i=1}^{m-1} G_i + \lambda_n G_n - \{1+\mu \sum_{i=2}^{m} G_{1i} + \lambda_n G_{1n}\}}{(m-1)\mu + \lambda_n}$$

$$\geq 0$$

by induction hypothesis. Now $G-G_n \geq 0$ by 4.2.18 and consequently $G-G_j \geq 0$ for $1 \leq j < n$ by 4.2.17 and 4.2.19. Hence the theorem.

Theorem 4.2.7: The policy π_2 minimises the expected makespan.

Proof is similar to that of Theorem 4.2.11 (given at the end). In this case the set of actions that can be taken at time $t = 0$ is given by

$$J = \{(V,j) \mid j \in N,\ V \subseteq N-\{j\},\ |V| \leq \min(m-1, |N-\{j\}|)\}$$

$$\cup \{(V,0) \mid V \subseteq N,\ 1 \leq |V| \leq \min(|N|, m-1)\} .$$

Case III : $a = 1$, $b = 2$, $\mu \geq \max \lambda_j$ and the objective is to minimise the expected makespan.

Let π_3 be a policy in which at each decision moment we assign, among the set (U) of uncompleted tasks,

(i) the first task in the ~~nondecreasing~~ order of λ to A type processor and the last two tasks in the same order to B type processors when $|U| \geq 3$

(ii) the task with smaller value of λ to A type processor and the task with larger value of λ to one of B type processors when $|U| = 2$.

(iii) the task to A type processor when $|U| = 1$.

Let G, G_k and $G_{k\ell}$ represent the same terms as in case II with the exception that π_2 is replaced by π_3.

Here, we have

$$G = (1+\mu G_1+\lambda_{n-1}G_{n-1}+\lambda_n G_n)/(\mu+\lambda_{n-1}+\lambda_n)$$

or equivalently

$$\mu(G-G_1) + \lambda_{n-1}(G-G_{n-1}) + \lambda_n(G-G_n) = 1$$

for $n \geq 3$,

$$G = (1+\mu G_1+\lambda_2 G_2)/(\mu+\lambda_2)$$

for $n = 2$ and $G = 1/\mu$ for $n = 1$.

Theorem 4.28 :

$$G \geq G_j \quad \text{for} \quad 1 \leq j \leq n \qquad 4.2.20$$

$$G_j \geq G_1 \quad \text{for} \quad 1 \leq j \leq n \qquad 4.2.21$$

$$\min\{\lambda_n(G-G_n),\ \lambda_{n-1}(G-G_{n-1})\} \geq \lambda_j(G-G_j) \quad \text{for} \quad 1 \leq j < n-1$$

$$\text{if} \quad n \geq 3 \quad \text{and} \quad \lambda_2(G-G_2) \geq \lambda_1(G-G_1) \quad \text{if} \quad n = 2 \qquad 4.2.22$$

Proof : We prove the theorem by ~~induction on n.~~ for $n = 2$ the above inequalities can be easily verified.

Let $n = 3$,. Then we have

$$G = (1+\mu G_1+\lambda_2 G_2+\lambda_3 G_3)/(\mu+\lambda_2+\lambda_3)$$

$$\begin{aligned} G_3-G_1 &= \frac{1+\mu G_{31}+\lambda_2 G_{32}}{\mu+\lambda_2} - \frac{1+\mu G_{12}+\lambda_3 G_{13}}{\mu+\lambda_3} \\ &= \frac{1+1+\lambda_2/\mu}{\mu+\lambda_2} - \frac{1+1+\lambda_3/\mu}{\mu+\lambda_3} \\ &= (\lambda_3-\lambda_2)/(\mu+\lambda_2)(\mu+\lambda_3) \\ &\geq 0 . \end{aligned}$$

Similarly, $G_3-G_2 = (\lambda_3-\lambda_2)/(\mu+\lambda_2)(\mu+\lambda_3) \geq 0$ and $G_2-G_1 = (G_3-G_1)-(G_3-G_2) = 0$.

Thus the inequalities 4.2.21 hold for $n = 3$. We have

$$\begin{aligned} &\lambda_3(G-G_3)-\lambda_2(G-G_2) \\ &= \frac{\lambda_3}{\mu+\lambda_2+\lambda_3}[1-(\mu+\lambda_2)(G_3-G_2)] - \frac{\lambda_2}{\mu+\lambda_2+\lambda_3}[1+\lambda_3(G_3-G_2)] \end{aligned}$$

since $G_2 = G_1$. Now we can write

$$
\begin{aligned}
(\mu+\lambda_2+\lambda_3)[\lambda_3(G-G_3)-\lambda_2(G-G_2)] &= (\lambda_3-\lambda_2)-\lambda_3(\mu+2\lambda_2)(G_3-G_2) \\
&= (\lambda_3-\lambda_2)[1-\lambda_3(\mu+2\lambda_2)/(\mu+\lambda_2)(\mu+\lambda_3)] \\
&\geq 0.
\end{aligned}
$$

We have $\lambda_2(G-G_2) \geq \lambda_1(G-G_1)$ since $\lambda_2 \geq \lambda_1$ and $G_1 = G_2$. Thus the inequalities 4.2.22 hold for $n = 3$. Now we can easily see that $G-G_j \geq 0$ for $1 \leq j \leq 3$.

Suppose the inequalities 4.2.20, 4.2.21 and 4.2.22 hold for $n = k(\geq 3)$. We shall now show that it is true for $n = k+1$ also. Let $n = k+1$. Then we have for $1 < j \leq n-2$

$$
\begin{aligned}
&(\mu+\lambda_{n-1}+\lambda_n)[\lambda_n(G-G_n)-\lambda_j(G-G_j)] \\
&\quad = (\lambda_n-\lambda_j)(\mu+\lambda_{n-1}+\lambda_n)G-\lambda_n(\mu+\lambda_{n-2}+\lambda_{n-1})G_n-\lambda_n(\lambda_n-\lambda_{n-2})G_n+\lambda_j(\mu+\lambda_{n-1}-\lambda_n)G_j
\end{aligned}
$$

expanding $(\mu+\lambda_{n-1}+\lambda_n)G$ and rearranging all the terms on the right hand side, we can write

$$
\begin{aligned}
&(\mu+\lambda_{n-1}+\lambda_n)[\lambda_n(G-G_n)-\lambda_j(G-G_j)] \\
&\quad = \mu[(\lambda_n-\lambda_j)G_1-\lambda_n G_{1n}+\lambda_j G_{1j}] + \lambda_{n-1}[(\lambda_n-\lambda_j)G_{n-1}-\lambda_n G_{n-1\,n} \\
&\qquad + \lambda_j G_{n-1\,j}] + \lambda_n[(\lambda_{n-2}-\lambda_j)G_n-\lambda_{n-2} G_{n\,n-2} + \lambda_j G_{nj}] \\
&\quad \geq 0
\end{aligned}
$$

by induction hypothesis, that is, $\lambda_n(G-G_n) \geq \lambda_j(G-G_j)$ for $1 < j \leq n-2$.

Similarly,

$$(\mu+\lambda_{n-1}+\lambda_n)[\lambda_n(G-G_n)-\lambda_1(G-G_1)]$$

$$= \mu[(\lambda_n-\lambda_1)G_1-\lambda_nG_{1n}+\lambda_1G_{12}]+\lambda_{n-1}[(\lambda_n-\lambda_1)G_{n-1}-\lambda_nG_{n-1n}$$

$$+\lambda_1G_{n-11}]+\lambda_n[(\lambda_{n-2}-\lambda_1)G_n-\lambda_{n-2}G_{nn-2}+\lambda_1G_{n1}]$$

$$\geq \mu[(\lambda_n-\lambda_1)G_1-\lambda_nG_{1n}+\lambda_1G_{12}]$$

(by induction hypothesis)

$$\geq \mu[(\lambda_n-\lambda_2)G_1-\lambda_nG_{1n}+\lambda_2G_{12}]$$

$$\geq 0.$$

The last inequality holds by induction hypothesis and the previous one holds since $-\lambda_1 \geq -\lambda_2$ and $G_1-G_{12} \geq 0$. Following the same lines as above, we can show that $\lambda_{n-1}(G-G_{n-1}) \geq \lambda_j(G-G_j)$ for $1 \leq j \leq n-2$ and $\lambda_{n-1}(G-G_{n-1}) \geq \lambda_1(G-G_1)$. We can write

$$(\mu+\lambda_{n-2}+\lambda_{n-1})G_n = 1+\mu G_{n1}+\lambda_{n-2}G_{nn-2}+\lambda_{n-1}G_{nn-1}$$

$$(\mu+\lambda_{n-1}+\lambda_n)G_1 = 1+\mu G_{12}+\lambda_{n-1}G_{1n-1}+\lambda_nG_{1n}$$

From these equations, we get

$$(\mu+\lambda_{n-2}+\lambda_{n-1})(G_n-G_1)$$

$$= [(\lambda_n-\lambda_{n-2})G_1-\lambda_nG_{1n}+\lambda_{n-2}G_{nn-2}]+\lambda_{n-1}(G_{nn-1}-G_{1n-1})+\mu(G_{n1}-G_{12})$$

$$\geq (\lambda_n-\lambda_{n-2})G_1-\lambda_nG_{1n}+\lambda_{n-2}G_{nn-2}$$

$$\geq (\lambda_n-\lambda_{n-2})G_1-\lambda_nG_{1n}+\lambda_{n-2}G_{1n-2}$$

by induction hypothesis. So $G_n \geq G_1$. On the same lines we can easily see that

$$G_{n-1} \geq G_1, \quad G_j \geq G_1 \quad \text{for} \quad 1 \leq j \leq n-2$$

and

$$G_n \geq G_j \quad \text{for} \quad 1 < j \leq n-2 . \qquad 4.2.23$$

Now

$$G-G_1 = \{1 + \lambda_{n-1}(G_{n-1}-G_1)+\lambda_n(G_n-G_1)\}/(\mu+\lambda_{n-1}+\lambda_n)$$

$$\geq 0$$

and $G-G_n \geq 0$ and $G-G_{n-1} \geq 0$ since $\min\{\lambda_n(G-G_n), \lambda_{n-1}(G-G_{n-1})\} \geq \lambda_1(G-G_1)$
It now follows from 4.2.23 that $G-G_j \geq 0$ for $1 < j \leq n-2$. Hence the theorem.

<u>Theorem 4.2.9</u>: The policy π_3 minimises the expected makespan.

Proof is similar to that of Theorem 4.2.5. The set of actions that can be taken initially at time $t = 0$ is given by

$$J = \{(j,V)|j \in N, \; V \subseteq N-\{j\}, \; |V| \leq 2\} \bigcup \{(0,V)|V \subseteq N, \; 1 \leq |V| \leq 2\}$$

In an action (j,V), we assign take j to A type processor and tasks in V to B type processors. Zero indicates no assignment of any task to A type processor. For $n = 1,2$ and 3, we can easily verify, as in Theorem 4.2.5 that the policy π_3 is optimal. For $n > 3$ we use induction method as in Theorem 4.2.5.

<u>Case IV</u> : $a = 1$, $b = m-1$, $\mu \leq \min_j \lambda_j$ and the objective is to minimise the expected flowtime.

Let π_4 be a policy in which at each decision moment. We assign, among the set (U) of uncompleted tasks,

(i) the first task in the non-decreasing order of λ to A type processor and the last $(m-1)$ tasks in the same order to B type processors when $|U| \geq m$

(ii) all the tasks to B type processors when $|U| \leq m-1$.

Let G, G_k and $G_{k\ell}$ represent the same terms as in Case I with the exception that π_1 is replaced by π_4. Now, we have

$$G = \frac{n + \mu G_1 + \sum_{i=h}^{n} \lambda_i G_i}{\mu + \sum_{i=h}^{n} \lambda_i} \quad \text{for } n \geq m$$

where $h = n-m+2$ and $G = \sum_{i=1}^{n} 1/\lambda_i$ for $n \leq m-1$.

Theorem 4.2.10

$$G \geq G_j \quad \text{for} \quad 1 \leq j \leq n \tag{4.2.24}$$

$$G_j \geq G_1 \quad \text{for} \quad 1 < j \leq n \tag{4.2.25}$$

$$\lambda_k(G-G_k) \geq \lambda_j(G-G_j) \quad \text{for} \quad 1 \leq j < k \leq n \quad \text{if} \quad n \leq m$$

and

$$\lambda_k(G-G_k) \geq \lambda_j(G-G_j) \quad \text{for} \quad 1 \leq j < n-m+2 \leq k \leq n \tag{4.2.26}$$

otherwise.

Proof : We prove the theorem by induction on n. It is easy to verify the above inequalities hold for $n \leq m-1$. Let $n = m$. Then we have

$$G = (n+\mu G_1 + \sum_{i=2}^{n} \lambda_i G_i)/(\mu + \sum_{i=2}^{n} \lambda_i)$$

and

$$G_j = \sum_{i=1}^{n} 1/\lambda_i - 1/\lambda_j \quad \text{for} \quad 1 \leq j \leq n.$$

For $1 \leq j < k \leq n$, we have

$$\lambda_k(G-G_k)-\lambda_j(G-G_j)$$

$$= \frac{1}{\mu + \sum_{i=2}^{n} \lambda_i} [n\lambda_k+\mu\lambda_k(G_1-G_k)+\lambda_k \sum_{i=2}^{n} \lambda_i(G_i-G_k)-\{n\lambda_j+\mu\lambda_j(G_1-G_j)$$

$$+ \lambda_j \sum_{i=2}^{n} \lambda_i(G_i-G_j)\}]$$

$$= \frac{1}{\mu + \sum_{i=2}^{n} \lambda_i} [n(\lambda_k-\lambda_j) + \frac{\mu}{\lambda_1} (\lambda_j-\lambda_k) + \sum_{i=2}^{n} (\lambda_j-\lambda_k)]$$

(by substituting $G_\ell = \sum_{i=1}^{n} 1/\lambda_i - 1/\lambda_\ell$ for $1 \leq \ell \leq n$)

$$= \frac{(\lambda_k-\lambda_j)}{\mu + \sum_{i=2}^{n} \lambda_i} [n-\mu/\lambda_1 - (n-1)]$$

$$\geq 0$$

since $\mu \leq \lambda_1$. Thus $\lambda_k(G-G_k) \geq \lambda_j(G-G_j)$ for $1 \leq j < k \leq n$. It is obvious that $G_j \geq G_1$ for $1 < j \leq n$ and consequently $G-G_1 \geq 0$. Since $\lambda_k(G-G_k) \geq \lambda_j(G-G_j)$ for $j < k$ it now follows that $G-G_j \geq 0$ for $1 \leq j \leq n$.

Suppose the inequalities 4.2.24, 4.2.25 and 4.2.26 hold for $n = k (\geq m)$. We shall now show that it is true for $n = k+1$ also. Let $n = k+1$. Then we have for $1 \leq j < h \leq k \leq n$

$$(\mu + \sum_{i=h}^{n} \lambda_i)[\lambda_k(G-G_k)-\lambda_j(G-G_j)]$$

$$= (\lambda_k-\lambda_j)(\mu + \sum_{i=h}^{n} \lambda_i)G-\lambda_k(\mu + \sum_{\substack{i=h-1\\ i\neq k}}^{n}\lambda_i)G_k-\lambda_k(\lambda_k-\lambda_{h-1})G_k$$

$$+ \lambda_j(\mu + \sum_{i=h}^{n} \lambda_i)G_j$$

$$= (\lambda_k-\lambda_j)[n+\mu G_1+ \sum_{i=h}^{n} \lambda_i G_i] - \lambda_k[\overline{n-1} + \mu G_{k1} + \sum_{\substack{i=h-1\\ i\neq k}}^{n}\lambda_i G_{k_i}]$$

$$+ \lambda_j[\overline{n-1} +\mu G_{j1} + \sum_{i=h}^{n} \lambda_i G_{ji}] - \lambda_k(\lambda_k-\lambda_{h-1})G_k$$

$$= (\lambda_k-\lambda_j) + \mu[(\lambda_k-\lambda_j)G_1-\lambda_k G_{1k}+\lambda_j G_{ij}] \sum_{\substack{i=h\\ i\neq k}}^{n} \lambda_i[(\lambda_k-\lambda_j)G_i$$

$$- \lambda_k G_{ik} + \lambda_j G_{ij}] + \lambda_k[(\lambda_{h-1}-\lambda_j)G_k-\lambda_{h-1} G_{kh-1} + \lambda_j G_{kj}]$$

$$\geq 0$$

by induction hypothesis.

To show $G_j \geq G_1$ for $h \leq j \leq n$, consider the equations

$$(\mu + \sum_{\substack{i=h-1\\ i\neq j}}^{n} \lambda_i)G_j = (n-1) + \mu G_{j1} + \sum_{\substack{i=h-1\\ i\neq j}}^{n} \lambda_i G_{ji}$$

$$(\mu + \sum_{i=h}^{n} \lambda_i)G_1 = (n-1) + \mu G_{12} + \sum_{i=h}^{n} \lambda_i G_{1i}$$

Substracting the later equation from the former one and applying induction hypothesis, we get

$$\begin{aligned}(\mu + \sum_{\substack{i=h-1\\ i\neq j}}^{n} \lambda_i)(G_j - G_1) &= (\lambda_j - \lambda_{h-1})G_1 - \lambda_j G_{1j} + \lambda_{h-1} G_{jh-1} \\ &\geq (\lambda_j - \lambda_{h-1})G_1 - \lambda_j G_{ij} + \lambda_{h-1} G_{1h-1} \\ &\geq 0\end{aligned}$$

by induction hypothesis. Similarly, we can show $G_j \geq G_1$ for $1 < j < h$ and

$$G_n \geq G_j \quad \text{for} \quad 1 \leq j < h . \qquad 4.2.27$$

It is easy to see that $G - G_1 \geq 0$ and consequently $G - G_j \geq 0$ for $h \leq j \leq n$ by inequalities 4.2.26. $G - G_j \geq 0$ for $1 < j < h$ since $G - G_n \geq 0$ and $G_n \geq G_j$ for $1 < j < h$. Hence the theorem.

Theorem 4.2.11: The policy π_4 minimises the expected flowtime.

Proof : We prove the theorem by induction on n. It is obvious that π_4 is optimal for n = 1. Suppose it is true for n = k. We shall now show that it is true for n = k+1 also. Let n = k+1. The set of actions that can be taken at time t = 0 is given by

$$J = \{(j,V) \mid j \in N,\ V \subsetneq N - \{j\},\ |V| \leq \min(|N-\{j\}|, m-1)\}$$

$$\bigcup \{(0,V) \mid V \subseteq N, 1 \leq |V| \leq \min(|N|, m-1)\}$$

Let G' represent the expected flowtime when an action f' = (j,V) in J is taken at time t = 0 and π_4 is followed later on. We can write

$$G' = (n + \mu G_j + \sum_{i \in V} \lambda_i G_i)/(\mu + \sum_{i \in V} \lambda_i)$$

and

$$(\mu + \sum_{i \in V} \lambda_i)(G' - G) = n - \{\mu(G - G_j) + \sum_{i \in V} \lambda_i (G - G_i)\} .$$

If $n \leq m-1$, then

$$(\mu + \sum_{i\varepsilon V}\lambda_i)(G'-G) = n-\{\mu/\lambda_j + |V|\}$$
$$\geq 1-\mu/\lambda_j$$
$$\geq 0.$$

If $n \geq m$, then

$$(\mu + \sum_{i\varepsilon V}\lambda_i)(G'-G) \geq n-\{\mu(G-G_1) + \sum_{i=h}^{n}\lambda_i(G-G_i)\}$$
$$= 0$$

The above inequality holds by Theorem 4.2.10 / and the equality holds since

$(\mu + \sum_{i=h}^{n} \lambda_i)G = n + \mu G_1 + \sum_{i=h}^{n} \lambda_i G_i$.

Similarly, we can see that $G' \geq G$ when $f' = (0,V)$. Therefore π_4 is optimal for $n = k+1$ also. Hence π_4 is optimal for any n.

So far, we have shown that the policies π_1, π_2, π_3 and π_4 are optimal for the cases I, II, III and IV of the problem P' respectively when the pre-emptions and switches are allowed only at task completion times. Using semi-Markov decision arguments as earlier, we can easily see that the above policies are optimal for the respective cases even when pre-emptions and switches are allowed at times

$\delta < 2\delta < \ldots < T_1 < T_1 + \delta < T_1 + 2\delta < \ldots < T_2 < T_2 + \delta < \ldots$ where $T_1, T_2, \ldots$ are task completion times.

Remarks 4.2.12 :

In view of the above results we conjecture that for the problem with 'a' processors of type A and 'b' processors of type B, the expected flowtime (expected makespan) is minimised by the policies $\beta_1(\beta_2)$ when $\min \lambda_j \geq \mu (\max \lambda_j \leq \mu)$, where β_1 and β_2 are as described below:

β_1 is a policy in which at any decision moment

(i) when the number (r) of uncompleted tasks is atleast a+b, last b tasks in the non-decreasing order of λ are assigned to processors of type B and first a tasks in the same order assigned to processors of type A

(ii) when $b \leq r < a+b$, the last b tasks in the non decreasing order of λ are assigned to processors of type B and the remaining tasks assigned to processors of type A

(iii) when $r < b$, all tasks are assigned to processors of type B.

β_2 is a policy in which at any decision moment

(i) when the number (r) of uncompleted tasks is atleast a + b, first a tasks in the non-decreasing order of λ an assigned to processors of type A and the last b tasks in the same order assigned to processors of type B

(ii) when $a \leq r < a + b$, first a tasks in the non-decreasing order of λ are assigned to processors of type A and the remaining tasks assigned to type B

(iii) when $r < a$, all tasks are assigned to processors of type A.

~~It is obvious that the total expected cost gives the expected flow time. Therefore the policy π minimises the expected flow time. In the above semi-Markov decision process we allow decisions at times $0 < T_1 < T_2 \ldots$ where T_i's are transition epochs (job completion times in other words). If we allow decisions at times $0 < \delta < 2\delta < \ldots \leq T_1 < (T_1 + \delta) < (T_1 + 2\delta) < \ldots \leq T_2 < (T_2 + \delta) < (T_2 + 2\delta) < \ldots$, however small δ is, the process is still a semi-Markov decision process for which~~

~~(1) the set of stationary policies is the same as above,~~

~~(2) the theorem 4.2.1 holds~~

~~(3) the policy π minimises the total expected cost. This means that even if we allow interruptions and switches of tasks (from one processor to another) at times $0, \delta, 2\delta, \ldots, T_1, T_1 + \delta, T_1 + 2\delta, \ldots, T_2, T_2 + \delta, \ldots$ the policy π minimises the expected flow time.~~

4.3 Single-Processor Stochastic Scheduling Problem

Introduction

In this section, we consider a single-processor stochastic scheduling problem in which the processor is subject to breakdowns and repairs. It is generally assumed in scheduling theory that the processor never breaks down. However, in many practical situations it is quite common that the processor breakdown occasionally while operating and its repair requires certain amount of time. The continuous operating time and the repair time of the processor and processing times of the tasks may all be non-deterministic.

Our problem is to find, under this environment, a policy that minimises the expected weighted sum of task completion times. We define the problem through the following assumptions.

(1) There are n tasks $1,2,\dots,n$ to be processed on a single processor.

(2) Only one task can be processed at a time.

(3a) The processor is subject to breakdowns and repairs.

(3b) Initially at time $t = 0$ the processor is in operating condition.

(3c) The length Y_i of i th period over which the processor operates continuously follows exponential distribution with parameter μ for $i \geq 1$, i.e., all Y_i's are identically distributed exponential random variables.

(3d) The length Z_i of i th repair period, $i \geq 1$, is a r.v. with expectation $\theta(<\infty)$ and all Z_i's are identically distributed.

(4) The amount of processing time X_j that a job j, $1 \leq j \leq n$, requires is an exponential random variable with parameter λ_j.

(5) Set-up times are assumed to be zero.

(6) All the random variables described above are mutually independent.

(7) Pre-emptions are allowed only at the time of completion of repair.

(8) Cost c_j is incurred on task j $(1 \leq j \leq n)$ per unit time till the task j is completed.

The objective of this problem is to minimise the total expected cost. By assumption (7) we allow decisions as to assign a task to the processor at task completion times and also whenever the repair of the processor is completed. In order to formulate the problem as a semi-Markov decision process, we allow decisions at failure times also which are immaterial in view of decisions at repair completion times.

. Formulation as a Semi-Markov Decision Process

The state of the system at time t is (U,δ) where U is the set of uncompleted tasks at time t and $\delta = 0$ if the processor is being repaired at time t and $\delta = 1$ otherwise. The state at time $t = 0$ is $(N,1)$ where $N = \{1,2,\ldots,n\}$. The set of actions $J(U,\delta)$ associated with a state (U,δ), $U = \{i_1,i_2,\ldots,i_r\}$, is defined as $J(U,\delta) = \{0,i_1,i_2,\ldots,i_r\}$ i.e., $J(U,\delta) = U \bigcup \{0\}$. An action $k \in J(U,\delta)$, $k \neq 0$; is an assignment of task $k \in U$ to the processor. The action $k = 0$ means no assignment of tasks to the processor.

Let $P^k[(U,\delta),\ (U',\delta')]$ represent the probability of transition from a state (U,δ) to a state (U',δ') under the action k. These transition probabilities **are given as**

$$P^k[(U,0),\ (U,1)] = 1 \quad \text{for} \quad k \in J(U,0)$$

$$P^k[(U,1),\ (U,0)] = \frac{\mu}{\mu+\lambda_k}$$

and
$$P^k[(U,1),\ (U-\{k\},1)] = \frac{\lambda_k}{\mu+\lambda_k} \quad \text{for} \quad k \in J(U,1)$$

where
$$\lambda_0 = 0.$$

Let $T^k(U,\delta)$ represent the sojourn time in a state (U,δ) under the action k. These sojourn times $T^k(U,\delta)$'s are given as

$$T^k(U,0) = Z \quad \text{for} \quad k \in J(U,0)$$

$$T^0(U,1) = Y$$

$$T^k(U,1) = \min(X_k,Y) \text{ for } k \in J(U,1) \text{ and } k \neq 0$$

where Y is an exponential random variable with parameter μ and Z a r.v. following the distribution of Z_i's. Cost $\sum_{i \in U} c_i$ is incurred per unit time during the sojourn time $T^k(U,\delta)$ and the discount factor α is zero.

For a semi-Markov decision process, it is enough to consider the stationary policies in order to minimise the expected cost. For reference, see Ross (1970), Chapter 7. Therefore, we restrict our attention to the set (C_s) of all stationary policies. Since the actions taken in the states of the type $(U,0)$ do not effect the total expected cost we consider only the stationary policies in which the actions choosen in the states $(U,0)$ and $(U,1)$ are same for each $U \subseteq N$. Let C_{ss} represent the set of all such stationary policies. It is obvious that a stationary policy in C_{ss} which takes the action 0 in states $(U,0)$ and $(U,1)$ for a particular $U \neq \phi$ gives total expected cost ∞. Therefore, we consider only the stationary policies of C_{ss} which do not take the action 0 in any state (U,δ), $U \neq \phi$. Let us represent the set of such stationary policies by C^*_{ss}.

A policy f of C^*_{ss} takes same action (non-zero) in both the states $(U,0)$ and $(U,1)$ for each $U \neq \phi$, that is, the policy f takes an action corresponding to the set U only. The policies of C^*_{ss} can be divided into two kinds (i) permutation policies and (ii) non-permutation policies. A permutation policy f is a stationary policy to which there corresponds a permutation $\pi = (\pi_1,\pi_2,\ldots,\pi_n)$ of the elements of N such that when the elements $\pi_1,\pi_2,\ldots,\pi_n$ are renumbered as $1,2,\ldots,n$ respectively the policy f takes an action $k = \min_{j \in U} j$ in a state (U,δ), $U \neq \phi$. The permutation policy f can be simply represented by the corresponding permutation π. The policies other than the permutation policies in the set C^*_{ss} are called non-permutation policies. Below, we illustrate permutation and non-permutation policies for $N = \{1,2,3\}$.

Permutation policy : Let k represent the action choosen corresponding to U. A policy (of C^*_{ss}) in which

k	=	2	for	U	=	{1,2,3}
k	=	2	for	U	=	{1,2}
k	=	2	for	U	=	{2,3}
k	=	3	for	U	=	{1,3}

is a permutation policy. The underlying permutation of the policy is (2,3,1).

Non-permutation policy : A policy (of C^*_{ss}) in which

k	=	2	for	U	=	{1,2,3}
k	=	1	for	U	=	{1,2}
k	=	3	for	U	=	{2,3}
k	=	3	for	U	=	{1,3}

is a non-permutation policy.

Under a policy f of C^*_{ss} the system moves from a given set U of cardinality $r(\geq 1)$ to a set U' of cardinality r-1 uniquely. So, for any non-permutation policy f, we can find a corresponding permutation policy f' such that the processes under f and f' are identical. We can find such a correspondence in the above given illustrations. Therefore we consider only the permutation policies in order to minimise the total expected cost.

Let $G_\pi(U,1)$ represent the total expected cost over the interval $[t,\infty)$ under a permutation (permutation policy) $\pi = (\pi_1, \pi_2 \ldots, \pi_n)$ when (U,1) is the state of the system at time t. We simply represent $G_\pi(N,1)$ by G_π. In the theorem given below, we obtain an optimal permutation that minimises G_π over the set (S) of all permutations.

Lemma 4.3.1 Suppose we choose a task, say j, for processing initially at time $t = 0$ and process it until its completion without preemption. Let T_j represent the time at which the task j is completed. Then $E[T_j] = \frac{1+\mu\theta}{\lambda_j}$

Proof : Using renewal concept, we can write

$$E[T_j] = E[\min(X_j, Y_1)] + E[Z_1 + T_j]\, P(X_j > Y_1)$$

$$= \frac{1}{\mu+\lambda_j} + \frac{[\theta + E(T_j)]\mu}{\mu+\lambda_j}$$

i.e. $E[T_j] = (1+\mu\theta)/\lambda_j$.

Theorem 4.3.2 : If $c_{p_1}\lambda_{p_1} \geq c_{p_2}\lambda_{p_2} \geq \cdots \geq c_{p_n}\lambda_{p_n}$, for a sequence

$p = p_1, p_2, \ldots, p_n)$, then $G_p = \min_{s \in S} G_s$.

Proof : Consider an arbitrary fixed permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$. Without loss of generality we assume that $\pi = (1,2,\ldots,n)$. We can write

$$G_\pi = [\sum_{i \in N} c_i] E(T_1) + G_\pi(N-\{1\},1)$$

(since the state is $(N-\{1\},1)$ at completion time T_1 of task 1)

$$= [\sum_{i \in N} c_i]\frac{(1+\mu\theta)}{\lambda_1} + G_\pi(N-\{1\},1) \text{ by Lemma 4.3.1}$$

Applying the above arguments repeatedly, we finally get

$$G_\pi = [\sum_{i=1}^{n} c_i]\frac{1+\mu\theta}{\lambda_1} + [\sum_{i=2}^{n} c_i]\frac{1+\mu\theta}{\lambda_2} + \ldots + [\sum_{i=n-1}^{n} c_i]\frac{1+\mu\theta}{\lambda_{n-1}} + (c_n)\frac{1+\mu\theta}{\lambda_n}$$

i.e., $G_\pi = (1+\mu\theta)\,[\frac{c_1}{\lambda_1} + c_2(\frac{1}{\lambda_1} + \frac{1}{\lambda_2}) + \ldots + c_n(\frac{1}{\lambda_1} + \frac{1}{\lambda_2} + \ldots + \frac{1}{\lambda_n})]$.

Therefore, for any arbitrary permutation $s = (s_1, s_2, \ldots, s_n)$ we have

$$G_s = (1+\mu\theta)[\sum_{\ell=1}^{n} c_{s_\ell}\,(\frac{1}{\lambda_{s_1}} + \frac{1}{\lambda_{s_2}} + \ldots + \frac{1}{\lambda_{s_\ell}})] .$$

Given $a_1, a_2, \ldots, a_m$ and $b_1, b_2, \ldots, b_m$, a_i and $b_i \geq 0$ for $1 \leq i \leq m$, a function $h(\pi)$ defined on the set of permutations of $1, 2, \ldots, m$ as

$$h(\pi) = \sum_{\ell=1}^{m} a_{\pi_\ell} \sum_{j=1}^{\ell} b_{\pi_j}$$

where $\pi = (\pi_1, \pi_2, \ldots, \pi_m)$ is minimised by a permutation $v = (v_1, v_2, \ldots, v_n)$ which satisfies $\frac{b_{v_1}}{a_{v_1}} \leq \frac{b_{v_2}}{a_{v_2}} \leq \ldots \leq \frac{b_{v_m}}{a_{v_m}}$. For reference, see Mc Naughton (1959).

Now, we can easily see that G_s is minimised by the permutation p.

Q.E.D.

From the earlier arguments and the above theorem, we conclude that the total expected cost will be minimised if the tasks are processed without pre-emptions in the non-increasing order of $c_j \lambda_j$.

Remark 4.3.4 : The optimal policy that minimises the total expected cost is independent of the nature of repair time and the value of parameter μ. If the tasks are processed without pre-emptions in the non-increasing order of $w_j \lambda_j$, where w_j, $1 \leq j \leq n$, is the weight associated with job i, then the expected weighted sum of completion times would be minimised.

REFERENCES

ACHUTHAN, N.R. and GHOSH, D.T. (1980) :"A note on heuristic rules for tho $(n/3/F/F_{max})$ problom", Opsearch, Vol. 17, pp. 50-56.

ACHUTHAN, N.R. (1980) : Flow-shop scheduling problems, Ph.D. Thesis, Indian Statistical Institute, Calcutta, India.

ARTHANARI, T.S. and RAMAMURTHY, K.G. (1970) : "A branch and bound algorithm for sequencing n jobs on m parallel processors", Opsearch, Vol. 7, no. 3.

ARTHANARI, T.S. and MUKHOPADHYAY, A.C. (1971) : "A note on a paper by W. Szwarc", Nav. Res. Log. Quart., Vol. 18, pp. 135-138.

ARTHANARI, T.S. (1974) : On some problems of sequencing and grouping , Ph.D. Thesis, Indian Statistical Institute, Calcutta, India.

ASHOUR, S. (1970) : "An experimental investigation and comparative evaluation of flow-shop sequencing techniques", Opn. Res., Vol. 18, pp.541-549.

BAGGA, P.C. (1970) : "n-job, 2-machine sequencing problems with stochastic service times", Opsearch, Vol. 7, pp. 184-197.

BAKER, K.R. (1974): "Introduction to sequencing and scheduling", John Wiley & Sons, Inc., N.Y.

BAKER, K.R. (1975) : An elimination method for the flow-shop problem , Opn. Res., Vol. 23, pp. 159-162.

BROWN, A.P.G. and LOMNICKI, Z.A. (1966) : "Some applications of the'branch and bound'algorithm to the machine scheduling problem", Opnl. Res. Quart., Vol. 17, pp. 173-186.

BRUNO, J. and DOWNEY, P. (1977) : "Sequencing tasks with exponential service times on two machines", Technical Report, Department of Computer Sciences University of California, Santa Barbara.

BURDYUK, V.Y. (1969) : "The m-machine problem $(m > 2)$", Kibernetika, Vol. 5, pp. 74-76.

CAMPBELL, H.G., DUDEK, R.A. and SMITH, M.L. (1970) : "A heuristic algorithm for the n-job, m-machine sequencing problom", Man. Sci., Vol. 16, pp. B 630-637.

CONWAY, R.W., MAXWELL, W.L. and MILLER, L.W. (1967) : Theory of Scheduling, Addison-Wesley, Reading, Mass.

CUNNINGHAM, A.A. and DUTTA, S.K. (1973) : "Scheduling jobs, with exponentially distributed processing times on two machines of a flowshop", Nav. Res. Log. Quart., Vol. 20, pp. 69-81.

EASTMAN,W.L., EVEN, S. and ISAACS : I.M. (1964), "Bounds for the optimal scheduling of n jobs on m processors", Man. Sci., Vol. 11, No. 2.

ELMAGHRABY, S.E. (1968) : "The machine sequencing problem-Review and extensions", Nav. Res. Log. Quart., Vol. 15, No. 2.

FREDERICKSON, G.N. (1978) : "Sequencing tasks with exponential service times to minimise the expected flow time or makespan. Department of Computer Sciences,Pennsylvania State University Report CS-78-07.

GAREY, M.R., JOHNSON, D.S. and SETHI, R. (1976) : "The complexity of flow-shop and job-shop scheduling", Math. Opns. Res., Vol. 1, pp. 117-129.

GAREY, M.R. and JOHNSON, D.S. (1979) : Computers and intractibility : a guide to the theory of NP-completeness, Freeman.

GIGLIO, R.J. and WAGNER, H.M. (1964) : Approximate solutions to the three-machine scheduling problem", Opn. Res., Vol. 12, pp.305-324.

GILMORE, P.C. and GOMORY, R.E. (1964) : "Sequencing a one-state variable machine : A solvable case of the travelling sales man problem", Opn. Res., Vol. 12, pp.655-679.

GITTINS, J.C. (1981) : Multi-server scheduling of jobs with increasing completion rates", J. Appl. Prob. Vol. 18, pp. 321-324.

GLAZEBROOK, K.D. (1979) : "Scheduling tasks with exponential service times on parallel processors", J. Appl. Prob. Vol. 16, pp. 685-689.

GRABOWSKI, J. and SYSLO, M.M. (1975) : "New solvable special cases of n-machine and n-element sequencing problems", Zastosowania Matematyki, Applicationes Mathematicae, Vol. 14, pp. 599-606.

GUPTA, J.N.D. (1971) : "An improved combinatorial algorithm for the flowshop scheduling problem", Opn. Res. Vol. 19, pp. 1753-1758.

GUPTA, J.N.D. (1975) : "Optimal schedules for special structure flowshops", Nav. Res. Log. Quart., Vol. 22, 255-269.

GUPTA, J.N.D. (1976) : "A review of flowshop scheduling research", U.S. Postal Service, Washington, DC, 20260.

IGNALL, E. and SCHRAGE, L. (1965) : "Application of the branch-and-bound technique to some flowshop scheduling problems", Opn . Res., Vol. 13, pp. 400-412.

JACKSON, J.R. (1956) : An extension of Johnson's results on job-lot scheduling", Nav. Res. Log. Quart., Vol. 3, No. 3.

JOHNSON, S.M. (1954) : "Optimal two- and three- stage production schedules with set-up times included", Nav. Res. Log. Quart., Vol. 1, pp. 61-68.

LAGEWEG, B.J., LENSTRA, J.K. and RINNOOY KAN, A.H.G. (1978) : "A general bounding scheme for the permutation flowshop problem", Opn .Res., Vol. 26, pp. 53-67.

LOMNICKI, Z.A. (1965) : A branch and bound algorithm for the exact solution of the three machine scheduling problem", Opnl. Res. Quart., Vol. 16, pp. 89-100.

MAKINO, T. (1965) : "On a scheduling problem", J. Opn. Res. Soc. Japan, Vol. 8, pp. 32-44.

Mc MAHON, G.B. and BURTON, P.G. (1967) : "Flowshop scheduling with the branch and bound method", Opn. Res., Vol. 15, pp. 473-481.

Mc MAHON, G.B. (1969) :"Optimal production schedules for flowshops", Canadian Operational Society Journal, Vol. 7, pp. 141-151.

Mc NAUGHTON, R. (1959) : "Scheduling with dead lines and loss functions", Man. Sci., Vol. 6, pp. 1-12.

MUTH, E.J. (1977) : Transform methods with application to engineering and operations research, Prentice-Hall Inc. Englewood Cliffs, New Jersey.

NABESHIMA, I. (1960 & 1961): "The order of n items processed on m machines", J. Opn. Res. Soc. Japan, Vol. 3, pp. 170-175 and Vol. 4, pp. 1-8.

NABESHIMA, I. (1971) : General scheduling algorithm with application to parallel scheduling and multi-project scheduling", J. Opn. Res. Soc. Japan, Vol. 14.

NABESHIMA, I. (1977) : "Notes on the analytical results in flowshop scheduling problem", Parts 1 and 2, Reports of the University of Electro-communications, Vol. 27, pp. 245-252 and 253-257.

OBERHETTINGER, F. and BADII, L. (1973) : Tables of Laplace transforms, Springer-Verlag, New York.

PALMER, D.S. (1965) : "Sequencing jobs through a multi-stage process in the minimum total time : A quick method of obtaining a near optimum", Opnl. Res. Quart., Vol. 16, pp. 101-107.

PANWALKAR, S.S. and KHAN, A.W. (1977) : "A convex property of an ordered flowshop sequencing problem", Nav. Res. Log. Quart., Vol. 24, pp. 159-162.

PANWALKAR, S.S. and WOOLLAM, C.R. (1979) : "Flowshop scheduling problems with no in-process waiting : A special case", Journal of Operational Research Society, Vol. 30, pp. 661-664.

PANWALKAR, S.S., SMITH, M.L. and WOLLAM, C.R. (1981) : "Counter examples to optimal permutation schedules for certain flowshop problems", Nav. Res. Log. Quart., Vol. 28, pp. 339-340.

PINEDO, M. and WEISS, G. (1979) : "Scheduling stochastic tasks on two parallel processors", Nav. Res. Log. Quart., Vol. 26, pp. 527-535.

PINEDO, M. (1980) : Scheduling spares with exponential lifetimes in a two-component parallel system", J. Appl. Prob., Vol. 17, pp.1025-1032.

RAJENDRA PRASAD, V. (1979) : "On flowshop scheduling problem with no in-process waiting", Discussion paper No. 7940, Indian Statistical Institute, Delhi Centre, India.

RAJENDRA PRASAD, V. (1980) : "n x 2 flowshop sequencing problem with random processing times", Opsearch, Vol. 18, pp. 1-14.

RAJENDRA PRASAD, V. (1982) : "Two-machine flowshop sequencing problem with uniformly distributed processing times", presented at the Indian Statistical Institute, Golden Jubilee Celebration Conference on Quality Control Reliability and Operations Research, held in New Delhi, Feb. 18-20, 1982.

RAJENDRA PRASAD, V. (1982) :"A letter to the Editor", to appear in J. Appl. Prob.

RAMAMURTHY, K.G. and RAJENDRA PRASAD, V. (1978) : "Some classes of flowshop problems", Discussion paper No. 7821, Indian Statistical Institute, Delhi Centre, India.

RANA, S.P. and ARORA, R.K. (1980) : "Scheduling in a semi-ordered flowshop without intermediate queues", AIIE Transactions, Vol. 12, pp. 263-272.

REDDY, S.S. and RAMAMURTHY, C.V. (1972) : On the flowshop sequencing problem with no wait in process", Opnl. Res. Quart., Vol. 23, pp. 323-331.

RINNOOY KAN, A.H.G. (1976) : Machine scheduling problems : classification, complexity and computations, Nijhoff, The Hague, Netherlands.

ROSS, S.M. (1970) : Applied probability models with optimisation applications, Holden-Day, San Francisco.

SMITH, R.D. and DUDEK, R.A. (1967) : "A general algorithm for solution of the n-job, M-machine sequencing problem of the flowshop", Opn. Res., Vol. 15, pp. 71-82.

SMITH, M.L., PANWALKAR, S.S. and DUDEK, R.A. (1975) : "Flowshop sequencing with ordered processing time matrices", Man. Sci., Vol. 21, pp. 544-549.

SMITH, M.L., PANWALKAR, S.S. and DUDEK, R.A. (1976) :"Flowshop sequencing problem with ordered processing time matrices: A special case", Nav. Res. Log. Quart., Vol. 23, pp. 481-486.

STRAUCH, R. (1966) : "Negative dynamic programming", Ann. Math. Stat. Vol. 37, pp. 871-890.

SZWARC, W. (1971) : "Optimal elimination methods in the m x n flowshop scheduling problem", Opn. Res., Vol. 21, pp. 1250-1259.

SZWARC, W. (1974) : "Mathematical aspects of the 3 x n job-shop sequencing problem", Nav. Res. Log. Quart., Vol. 21, pp. 145-153.

SZWARC, W. (1974a) : "A note on mathematical aspects of the 3 x n job-shop sequencing problem", Nav. Res. Log. Quart., Vol. 21, pp. 725-726.

SZWARC, W. (1977) : "Special cases of the flowshop problem", Nav. Res. Log. Quart., Vol. 24, pp. 483-492.

SZWARC, W. (1978) : "Permutation flowshop theory revisited", Nav. Res. Log. Quart., Vol. 25, pp. 557-570.

SZWARC, W. (1979) : "The critical path approach in the flow-shop problem", Opsearch, Vol. 16, pp. 98-102.

TALWAR, P.P. (1967) : "A note on sequencing problems with uncertain job times", J. Opn. Res. Soc. Japan, Vol. 9, pp. 93-97.

WEBER, R.R. (1978) : "On the optimal assignment of customers to parallel servers", J. Appl. Prob., Vol. 15, pp. 406-413.

WEBER, R.R. (1982) : "Scheduling jobs with stochastic processing requirements on parallel machines to minimise makespan or flow time", J. Appl. Prob., Vol. 19, pp. 167-182.

WEBER, R.R. and NASH, P. (1979) : "An optimal strategy in multi-server stochastic scheduling", J.R. Stat. Soc., Series B, Vol. 40, pp. 323-328.

WEISS, G. and PINEDO, M. (1980) : "Scheduling tasks with exponential service times on non-identical processors to minimise various cost functions", J. Appl. Prob. Vol. 17, pp. 187-202.

WHITE, D.J. (1978) : Finite dynamic programming, John-Wiley & Sons Ltd.

WISMER, D.A. (1972) : "Solution of the flowshop scheduling problem with no intermediate queues", Opn. Res. Vol. 20, pp. 687-697.

APPENDIX A

$$G_3-G_2 = \frac{2+\mu/\lambda_2+\lambda_2|\lambda_1}{\mu+\lambda_2} - \frac{2+\mu/\lambda_3+\lambda_3|\lambda_1}{\mu+\lambda_3}$$

i.e.,

$$(\mu+\lambda_2)(\mu+\lambda_3)(G_3-G_2)$$

$$= 2(\lambda_3-\lambda_2)+\mu(\mu+\lambda_3)/\lambda_2 + \lambda_2(\mu+\lambda_3)/\lambda_1-\mu(\mu+\lambda_2)/\lambda_3-\lambda_3(\mu+\lambda_2)/\lambda_1$$

$$= (\lambda_3-\lambda_2) + \mu(\mu+\lambda_3)/\lambda_2 - \mu(\mu+\lambda_2)/\lambda_3+(\lambda_3-\lambda_2)(1-\mu/\lambda_1)$$

$$= [(\lambda_3-\lambda_2)\lambda_2\lambda_3 + \mu^2(\lambda_3-\lambda_2) + \mu(\lambda_3^2-\lambda_2^2)]/\lambda_2\lambda_3+ (\lambda_3-\lambda_2)(1-\mu/\lambda_1)$$

$$= (\lambda_3-\lambda_2)[(\mu+\lambda_2)(\mu+\lambda_3)/\lambda_2\lambda_3 + 1-\mu/\lambda_1].$$

Thus, $G_3-G_2 = (\lambda_3-\lambda_2)[\frac{1}{\lambda_2\lambda_3} + \frac{1-\mu/\lambda_1}{(\mu+\lambda_2)\ (\mu+\lambda_3)}]$.

Now, we have

$$(\lambda_3-\lambda_2) -\mu\lambda_3(G_3-G_2)$$

$$= \mu\lambda_3(\lambda_3-\lambda_2)[\frac{1}{\mu\lambda_3} - \{ \frac{1}{\lambda_2\lambda_3} + \frac{1-\mu/\lambda_1}{(\mu+\lambda_2)(\mu+\lambda_3)}\}]$$

$$= \mu\lambda_3(\lambda_3-\lambda_2)[\frac{1-\mu/\lambda_2}{\mu\lambda_3} - \frac{1-\mu/\lambda_1}{(\mu+\lambda_2)(\mu+\lambda_3)}]$$

≥ 0 since $\lambda_2 \geq \lambda_1$ and $1/\mu\lambda_3 \geq 1/(\mu+\lambda_2)(\mu+\lambda_3)$.

APPENDIX B

$$\mu[\lambda_2(G_1-G_{21})-\lambda_1(G_2-G_{12})]$$

$$= \mu[\lambda_2\{\frac{2+\lambda_3(G_{31}-G_{21})}{\mu+\lambda_3}\} - \lambda_1\{\frac{2+\lambda_3(G_{23}-G_{21})}{\mu+\lambda_3}\}]$$

$$= \mu[2\lambda_2 + (\lambda_3-\lambda_2) - 2\lambda_1 - (\lambda_3-\lambda_1)]/(\mu+\lambda_3)$$

$$= \mu(\lambda_2-\lambda_1)/(\mu+\lambda_3)$$

$$\geq 0$$

INDIAN STATISTICAL INSTITUTE LIBRARY

Processing slip

Source............................P/E/G date 23.12.05 Acc. no. T 165

Author Prasad, V. Rajendra.

Title On some deterministic & stochastic scheduling problems.

CHECKING : Call no.

1 In library/earlier ed./later ed. [ ]

2 Earlier volumes/nos. of the series in lib. [ ]

3 New title [✓]

Suggested/Approved by

Note :

[illegible] 23.12.05

Signature & date

CLASSIFICATION :

Class no............................x-ref..

Scrutiny report : Signature & date

Class no............................x-ref..

Note :

Signature & date

CATALOGUING ;

1 Main Card [ ]

2 Supp. Cards (Subject/Title/Series/Added entries/Shelf List) [Total no. of cards]..........................

3 Mechanical processing completed on..............................

4 Cards filed [ ]

Note :

Signature & date