THE UNIVERSITY OF MICHIGAN

INDUSTRY PROGRAM OF THE COLLEGE OF ENGINEERING

SEQUENCING PROBLEMS WITH DUE DATE ORIENTED OBJECTIVE FUNCTIONS

Jon M. Moore

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in the
University of Michigan
Department of Industrial Engineering
1967

December, 1967

IP-796

PREFACE

To Professor Richard C. Wilson, who has been a most stimulating chairman and an outstanding advisor;

To ProfessorsR. M. Thrall, E. L. Lawler and D. H. Wilson for their advice and suggestions;

To the Industry Program of the College of Engineering at the University of Michigan for the final preparation;

To my wife for help and patience in preparing the initial draft;

To my mother and father for years of help and encouragement; Sincere thanks.

J. Michael Moore

TABLE OF CONTENTS

TABLE OF CONTENTS (CONT'D)

LIST OF APPENDICES

LIST OF SYMBOLS

J_i The job associated with the index i in some set of jobs.

t_i The processing time of job J_i .

D_i The due date of job J_i .

r_i The priority or reward associated with job J_i .

f_i The penalty or loss function associated with J_i where $f_i(x)$ represents the cost of being x units of time late in the completion of job J_i .

J Represents an arbitrary set of jobs, $\{J_1 \ldots J_n\}$.

S Represents a schedule or well ordering of an arbitrary set of jobs.

$L(i,k,t,T)$ Represents the loss associated with scheduling a job J_i at time t and another job J_k , T units of time after the completion of J_i .

M_i $= t_i + t - D_i$ is the amount of time by which J_i meets or fails to meet its due date in a schedule of the form given above.

N_k $= t_i + t_k + t + T - D_k$ is the amount of time by which J_k meets or fails to meet its due date in a schedule of the form given above.

$R(t,T)$ The relation defined on a set of jobs, J , where $(J_i,J_k) \in R(t,T)$ iff $L(i,k,t,T) \leq L(k,i,t,T)$ for arbitrary J_i , $J_k \in J$.

$J_i \ll (t,T) J_k$ Indicates that $(J_i,J_k) \in R(t,T)$

$J_i \ll J_k$ Indicates that $(J_i,J_k) \in R(t,T)$ for all $T \geq 0$.

R The relation defined on a set of jobs, J , where $(J_i,J_k) \in R$ iff $J_i \ll J_k$ for arbitrary J_i , $J_k \in J$.

$P(S,t)$ The sum of the lateness penalties incurred over a set of jobs, J , scheduled according to S where the first job commences at time t .

$J_i \ll (a) J_k$ Indicates that $(J_i,J_k) \in R(t,0)$ for all t .

R' The relation defined on a set of jobs, J , where $(J_i,J_k) \in R'$ iff $J_i \ll (a)\ J_k$ for J_i, $J_k \in J$.

D The relation defined on a set of jobs, J , as follows:
(1) If $J_i \in J$, then $(J_i,J_i) \in D$.
(2) If J_i,J_k are time dependently associated, then (J_i,J_k) and $(J_k,J_i) \in D$.

E_i A time dependent equivalence class with index i associated with the partition of a set of jobs, J , by the equivalence relation D .

$\mathcal{T}_{i,k}(T)$ The value of t for a pair of single-time dependent jobs such that
$J_i \ll (t',T)J_k$ when $t' \leq t$ and $J_k \ll (t',T)J_i$
for $t' > t$.

E The set of jobs completed on or before their due dates in some schedule S .

L The set of jobs completed after their due dates in some schedule S .

A The ordered set of jobs defined by ordering the elements of the set E (defined above) according to their order in the schedule S .

R The ordered set of jobs defined by ordering the elements of the set L (defined above) according to their order in the schedule S .

π An arbitrary permutation of the set R .

P The ordered set obtained by applying the permutation π to the set R .

A_D The ordered set obtained by ordering the jobs in the set E (defined above) according to their due dates.

a_j The processing time for a job on the first of two machines in series.

b_j The processing time for a job on the second of two machines in series.

p_j Linear deferral cost coefficient for a job.

CHAPTER 1

INTRODUCTION

1.1 Introduction to Job Shop Scheduling Problems

The problems considered here are in the general area of job shop scheduling. Such problems occur in "job shops," which denote special types of manufacturing or service environments. While there is no generally accepted definition of a job shop, the following definition is given to point out some of its distinctive characteristics:

A job shop consists of a collection of service centers, each composed of a finite set of servers that perform operations or services for a set of entities called jobs. These jobs are unique in the sense that:

(a) They are identified with a given customer;

(b) They cannot generally be stockpiled for sale or service.

The term "job" was used in the above definition and can be thought of as a customer order consisting of a partially ordered set of operations or services to be performed in order to accomplish a well defined objective specified by the customer. In addition to the operations, there may be a delivery commitment (deadline) established by the shop, the customer, or both. Some good examples of job shops are computer or data processing centers, construction companies, repair shops, companies that manufacture primarily to "special order," and hospitals.

The scheduling of a job shop refers to the sequence of decisions that determines the operations on which the servers in each center are to be engaged as a function of time. The result of scheduling a job shop can be represented in the traditional form of a Gantt Chart.

The majority of the job shop scheduling problems considered in the literature concern methods for producing Gantt Charts that optimize a well defined scheduling objective. Some of these objectives are listed below. In each case a rather general statement of the objective is given (one that a shop manager might specify) followed by a series of possible specific mathematical representations of it.

1. Good customer relations should be maintained by seeing that jobs are delivered reasonably on time.

Some traditional representations of this objective are:

(a) Minimize the number of late jobs.

(b) Minimize the total lateness over all jobs.

(c) Minimize the maximum lateness.

(d) An arbitrary objective of this form is as follows: Let $J_1 \ldots J_n$ be jobs with deadlines $D_1 \ldots D_n$ and loss functions $f_1 \ldots f_n$ where $f_i(t)$ is the cost of being t units of time late in the completion of job J_i . The objective is to minimize $F(f_1(C_1-D_1),\ldots,f_n(C_n-D_n))$, where C_i is the completion time of job J_i in the schedule being evaluated and F is monotone non-decreasing for each of the arguments. For example, in (a) and (b), $F(f_1(C_1-D_1) \ldots f_n(C_n-D_n)) = \sum_{i=1}^{n} f_i(C_i-D_i)$ where f_i is a unit step function at $C_i = D_i$ for (a) and a unit slope ramp function commencing at $C_i = D_i$ for (b).

2. The workload at each service center should be reasonably level to avoid excessive overtime and idleness.

   This objective has been represented as:

   (a) Minimize the total idle time for all servers.

   (b) Minimize the total make-span (the time from the start of the first operation until the completion of the last operation).

3. In process inventory costs should be minimized.

   This objective can assume many forms depending on the "financial" structure of the organization.

   (a) Minimize the mean number of jobs in the shop.

   (b) Minimize the mean work completed or in progress on all jobs in the shop.

   (c) Minimize the mean total work on jobs in the shop.

   (d) Minimize the mean work remaining on jobs in the shop.

4. Minimize the shop congestion and flow times for jobs.

   Some common representations of this objective are:

   (a) Minimize mean flow time.

   (b) Minimize flow time variance.

   (c) Minimize the probability that the flow time for any job is greater than some given constant.

   (d) Minimize maximum flow time.

   (e) Minimize the mean number of jobs in the shop.

   (f) Minimize the variance of the number of jobs in the shop.

While this list of scheduling objectives is by no means exhaustive, most real-life objectives will fall into one of these classes.

The problem, then, is to devise some method for constructing Gantt Charts which optimizes the appropriate performance measure(s). The complexity of this problem depends on the following factors:

1. The number of facility types or service centers and whether this number is stochastic or deterministic.
2. The number and type of servers in each center and whether this number is stochastic or deterministic.
3. The type of possible routing patterns for jobs being processed by the shop.
4. The job arrival process.
5. The total number of jobs to be scheduled.
6. The number of operations to be scheduled and whether this number is stochastic or deterministic.
7. The performance measure to be optimized.
8. The presence or lack of "special shop properties" (for example, due-dates are proportional to processing times, etc.).

It is necessary to specify each of these factors before a reasonably well defined scheduling problem can emerge.

1.2 Job Shop Scheduling Models to be Considered

In order to obtain a reasonably workable and practical model, some simplifying assumptions are generally made concerning the factors listed above. The work in this dissertation falls into a class of job shop scheduling studies in which the following assumptions are made:

1. The number of facility types or service centers is a known constant.
2. The servers in each center are identical and their number is a known constant.
3. The job arrival process consists of one batch arrival of all jobs to be scheduled.
4. The operations for each job are fixed and known.

In particular a special subclass of these problems referred to as sequencing problems is considered where it is assumed that the shop has only one facility type or service center and that each job coming into the shop consists of one operation to be performed on that facility. In order to obtain analytic methods for optimizing the various performance measures in models of this type, the following assumptions are made:

1. The processing times (including set up and tear down) are known and assumed to be sequence independent.
2. If deadlines are associated with any jobs, these are known.
3. Once a job (operation) is started on a given server, it is processed to completion without interruption.
4. There are no machine breakdowns.
5. Only one operation can be performed by a server at one time.
6. A server is not permitted to be idle if jobs are queued for service.

This thesis deals primarily with single server systems although service centers with m (not necessarily identical) servers in parallel or 2 (non-identical) servers in series are also considered.

There are many known results for the single server case. Here the problem is to sequence a finite set of jobs through the server so as to optimize one of the previously mentioned objective functions. Most of these problems have been solved in the sense that for each job in the queue, an index may be calculated from its due-date and processing time. The performance measure is optimized by ordering the jobs according to these indices. A good summary of the state of the art in this area may be found in Conway, Maxwell, and Miller.(1) Little is known, however, about these problems if the objective function is due-date oriented (i.e. some realization of the first general objective previously specified). This work deals solely with these problems.

1.3 Objective Function Forms Considered

It was previously pointed out that a general objective for sequencing jobs subject to deadlines could be represented by a function of the following form:

$$\text{Minimize} \quad F(f_1(C_1-D_1), \dots , f_n(C_n-D_n))$$

The two general types of problems considered here are:

1. $F(f_1(C_1-D_1) \dots f_n(C_n-D_n)) = \sum_{i=1}^{n} f_i(C_i-D_i)$

2. $F(f_1(C_1-D_1) \dots f_n(C_n-D_n)) = \max (f_1(C_1-D_1) \dots f_n(C_n-D_n))$

An efficient solution method for the latter problem is presented in Chapter 5 when the f_i's are monotone non-decreasing functions. This result is an extension of a theorem obtained by Smith.(15)

The former problem is considered for two classes of penalty functions; one in which the f_i's are continuous, monotone non-decreasing and the other in which the f_i's are (single) step functions. These are discussed separately.

1.4 The f_i's are Continuous, Monotone Non-decreasing

This problem (and in some cases a more generalized form) has been considered for certain types of penalty functions by many authors (References 3,5,7,9,11,13,14 and 15). Held and Karp,(5) Lawler,(7) and Root(11) all present dynamic programming formulations of these problems for arbitrary f_i's . Unfortunately, the practical applicability of these methods is limited in that they are generally computationally infeasible for problems for twenty jobs or more. Mc Naughton(9) and Smith(15) present nice solutions for problems in which the penalty functions are linear and all due-dates are equal. However, attempts to extend these results to models in which the facility has m servers in parallel(11) or in which the due-dates are different(3,11,13) have met with little success. Problems in which the loss functions are quadratic have also been considered,(14) but not solved.

The primary objective in the first part of this work was to determine under what conditions a problem of this type can be easily solved and to show what the solution would be. The main result is the "reduction theorem" at the end of Chapter 2. This theorem states that for a certain class of these problems there exists an equivalence relation on the set of jobs to be scheduled. This relation induces a partition of the jobs into a set of equivalence classes, say $E_1 \ldots E_p$, on which a well ordering is defined so that there exists an optimal schedule (or sequence) for the jobs of the form $(E_{i_1}, \ldots, E_{i_p})$ where the pairs of classes $(E_{i_1}, E_{i_2}), \ldots, (E_{i_{p-1}}, E_{i_p})$ belong to the well ordering. The importance of this theorem is twofold. First, it follows that in these cases the only real problem involved in finding optimal schedules is that of scheduling jobs within these equivalence classes. It is further shown that the "nice" results in this area(9,11,15) arise when the latter problem is trivial. In this sense, they all follow directly from the reduction theorem.

Second, it broadens the practical applicability of exact techniques such as integer(8) or dynamic programming by possibly reducing the original problem to a series of proper subproblems.

On the basis of this result, Chapters 3 and 4 consider the characteristics of an optimal schedule for the jobs in such an equivalence class. In order to obtain meaningful results, only problems in which $f_1 = f_2 = \ldots = f_n = f$ are considered where f is a polynomial. It is shown that the reduction theorem can be applied to this class of problems but that little can be said concerning the characteristics of optimal schedules unless f is of the form $ax + bx^2$ $(a, b \geq 0)$. Further, it is shown that the most highly structured of these problems is that in which $f = ax$. Several characteristics that an optimal schedule for a problem of this type will have are determined. However, these are not sufficient to guarantee that any schedule having them will be optimal. This is shown by considering in detail problems involving just three jobs.

The investigation of the problems in this class ends on that rather negative note and it is this author's opinion that further analytic work in this area is likely to meet with only limited success.

The discussion of the three job problem, however, should provide a valuable source of counter-examples to proposed algorithms in this area.(3,13)

1.5 The f_i's are (single) step functions.

This problem has received little attention in the literature and the only known pertinent result is that obtained by Smith:(15) All jobs can be completed by their due-dates if and only if they are all on time in the schedule obtained by ordering the jobs in order of increasing due-dates.

Chapter 5 contains a simple algorithm for sequencing the jobs so as to minimize the number of late jobs (the f_i's are unit step functions). While the author was unable to find similar algorithms for minimizing the number of late jobs when the service center has m servers in parallel or for minimizing the weighted number of late jobs through a single server (f_i is a step function of height $r_i > 0$), efficient dynamic programming algorithms for these problems are presented in Chapter 6. Results not included here have also been obtained concerning the problem of sequencing jobs subject to some arbitrary partial ordering. A generalization of Smith's result was found, showing that there exists a well ordering of the jobs such that the jobs can all be completed on time consistent with the partial ordering if and only if they are all on time in the schedule defined by the well ordering. A rather simple algorithm was developed for determining this well ordering. In addition, a subclass of problems was specified for which an efficient dynamic programming algorithm can be used to find a schedule that minimizes the weighted number of late jobs subject to the condition that the jobs completed on time be consistent with the partial ordering. A final result defines one class of problems for which the dynamic programming results of the previous sections can be used to minimize the weighted number of late jobs subject to the condition that all jobs be scheduled in a manner consistent with the partial ordering.

CHAPTER 2

THE DEVELOPMENT OF THE REDUCTION THEOREM

2.1 Introduction

The problems considered here belong to a general class of problems in which a finite set of jobs must be sequenced through a single facility, minimizing some function of the lateness penalties incurred. Conway, Maxwell, and Miller(1) and Smith(15) contain good summaries of the known results in this area. Specifically, the special problem of minimizing the sum of the lateness penalties(3,5,7,9,11,13,14) is treated when the penalties are independent (i.e. the penalty incurred by the job does not depend on the penalties incurred by other jobs). The case of linear penalty functions,(3,9,11,13) quadratic penalty functions,(14) and general penalty functions(5,7) have been considered in the literature. Held and Karp(5) present a dynamic programming algorithm for solving these problems, which, unfortunately, is generally computationally infeasible for problems of 20 jobs or more.

This chapter and the two that follow concern conditions under which an optimal solution to such a problem can be obtained by solving a series of proper subproblems. The results, then, will broaden the practical applicability of exact algorithms, such as dynamic or integer programming.

The problem is to sequence n jobs, $J_1 \ldots J_n$, with known processing times, $t_1 \ldots t_n$, and due-dates, $D_1 \ldots D_n$, through a single production facility. For each job, J_i, there is a penalty or loss function, f_i, where $f_i(x)$ is the cost of being x units of time late in the completion of job J_i . The objective is to find a sequence for the set of jobs which minimizes the sum of the lateness penalty costs incurred over all jobs.

It is assumed that the processing times, which are defined to include the set-up and tear-down times, are independent of the sequence. In addition, the n jobs are to be available for production throughout the scheduling period. Once production begins, the facility operates continuously until all jobs are completed. Finally, no lot-splitting is allowed, so that production on a job continues from start to finish without interruption.

This chapter deals with the special class of these problems where the following assumptions are made concerning the penalty functions:

(1) $f_i(x) = 0 \ , \qquad x \leq 0$

$f_i(x) > 0 \ , \qquad x > 0$

(2) f_i is continuous*

(3) f_i is non-decreasing

The first and third assumptions imply that there is no penalty until a job is late, at which time a positive penalty is incurred which does not decrease with time. The continuity assumption rules out the possibility of jumps in the loss functions. In fact, as is illustrated in later chapters, the scheduling methods change considerably when this assumption is not satisfied.

*Mathematically, as will be shown in Appendix I, it is necessary that f_i be absolutely continuous. However, continuous functions which occur in real-life problems are normally absolutely continuous.

The theorems presented here concern the characterization of certain schedules which are optimal with respect to the specified performance criterion. The main result concerns the condition under which a special equivalence relation can be defined on the set of jobs, $J = \{J_i \ldots J_n\}$. This relation induces a partition of the jobs such that an optimal schedule for the complete set is obtained by finding an optimal schedule for the jobs in each equivalence class.

2.2 Two Job Sequencing Relations

Definition: Let $L(i,k,t,T)$ represent the loss associated with jobs J_i and J_k in a schedule where the processing of job J_i commences at time t and that of job J_k commences at time $t + t_i + T$.

Further, let $M_i = t_i + t - D_i$ and $N_k = t_k + t_i + t + T - D_k$ represent the lateness of jobs J_i and J_k respectively in a schedule of the form just defined.

Thus,

$$L(i,k,t,T) = f_i(M_i) + f_k(N_k)$$

and

$$L(k,i,t,T) = f_k(M_k) + f_i(N_i) \ .$$

The following set of results concerns the conditions under which a general inequality can be obtained between $L(i,k,t,T)$ and $L(k,i,t,T)$. Without loss of generality, assume that $t_i \leq t_k$ throughout this section.

<u>CASE I</u>: $D_i \leq D_k$.

LEMMA 2.1 Given jobs J_i and J_k with $D_i \leq D_k$.
If $f_i \succeq f_k$, f_i' is non-decreasing, and $f_i' \geq f_k'$,
then $L(i,k,t,T) \leq L(k,i,t,T)$ for all $T \geq 0$, $-\infty \leq t \leq +\infty$.

Proof: $L(i,k,t,T) = f_i(M_i) + f_k(N_k)$

$L(k,i,t,T) = f_k(M_k) + f_i(N_i)$

If $N_k \leq 0$, then $f_k(N_k) = f_k(M_k) = 0$

and $L(i,k,t,T) \leq L(k,i,t,T)$ since $f_i(M_i) \leq f_i(N_i)$.

Hence, assume $N_k > 0$.

Since $D_i \leq D_k$, $N_k = t_i + t_k + t + T - D_k \leq N_i = t_i + t_k + t + T - D_i$

and for $N_k > 0$ we have $N_i > 0$.

For $f_i \geq f_k$, $f_k(N_k) \leq f_i(N_i)$ and there are now three cases to consider.

(1) If $M_i \leq 0$, then $f_i(M_i) = 0$ and

$L(i,k,t,T) = f_k(N_k) \leq f_k(M_k) + f_i(N_i) = L(k,i,t,T)$.

(2) If $M_i > 0$ and $M_k \leq 0$, then $f_k(M_k) = 0$ and

$L(i,k,t,T) = f_i(M_i) + f_k(N_k)$

$L(k,i,t,T) = f_i(N_i) = f_i(N_k) + f_i(N_i) - f_i(N_k)$.

Since $f_i(N_k) \geq f_k(N_k)$, it is sufficient to show that

$[f_i(N_i) - f_i(N_k)] \geq f_i(M_i)$.

But, $M_k = t_k + t - D_k \leq 0$ or $t_k + t \leq D_k$.

Hence $M_i = t_i + t - D_i \leq t_k + t - D_i \leq D_k - D_i$.

Consequently, it is sufficient to show that

$[f_i(N_i) - f_i(N_k)] \geq f_i(D_k - D_i) \geq f_i(M_i)$.

This follows from the fact that $N_i - N_k = D_k - D_i$, f_i' is non-decreasing, and $N_k > 0$. A formal proof is obtained by applying the theorem presented in Appendix I.

(3) If $M_i > 0$ and $M_k > 0$, then $f_k(M_k) > 0$ and

$$L(i,k,t,T) = f_i(M_i) + f_k(N_k)$$

$$L(k,i,t,T) = f_k(M_k) + f_i(N_i)$$

$$= f_k(M_k) + f_i(M_i) + f_i(N_i) - f_i(M_i) .$$

In this case, it is sufficient to show that

$$f_k(M_k) + f_i(N_i) - f_i(M_i) \geq f_k(N_k)$$

or $$f_i(N_i) - f_i(M_i) \geq f_k(N_k) - f_k(M_k) .$$

This follows from the fact that $N_i \geq N_k$,

$$N_i - M_i = t_k + T \geq t_i + T = N_k - M_k ,$$

and $$f_i' \geq f_k' .$$

A formal proof results from applying the theorem presented in Appendix I.

Therefore, $L(i,k,t,T) \leq L(k,i,t,T)$ for all $T \geq 0$, t.

Reviewing the proof of this lemma, it is interesting to point out how each sufficient condition concerning the loss functions affects the result.

The three stated conditions were:

(1) $f_i \geq f_k$

(2) f_i' is non-decreasing

(3) $f_i' \geq f_k'$.

If the first condition is not satisfied,

$L(i,k,t,T) \leq L(k,i,t,T)$ if $N_k \leq 0$ (i.e. $T + t \leq D_k - t_i - t_k$) .

If $f_i \geq f_k$ but the second condition is not satisfied, then

$L(i,k,t,T) \leq L(k,i,t,T)$ for $t \leq D_i - t_i$.

If only the third condition is not satisfied, then

$L(i,k,t,T) \leq L(k,i,t,T)$ for $t \leq \max ((D_i - t_i), (D_k - t_k))$.

CASE II: $D_k < D_i$.

The following points in time are significant in the analysis to follow:

$$(D_k - t_i - t_k),\ (D_i - t_i - t_k),\ (D_k - t_k),\ (D_i - t_k)\ ,$$

and $(D_i - t_i)$.

Since $t_i \leq t_k$ and $D_k < D_i$,

$$(D_k - t_i - t_k) < (D_k - t_k),\ (D_i - t_i - t_k)$$

$$(D_i - t_k) > (D_k - t_k),\ (D_i - t_i - t_k)$$

and $$(D_i - t_i) \geq (D_i - t_k)\ .$$

There is no general relation, however, between $(D_k - t_k)$ and $(D_i - t_i - t_k)$. The two possible cases are shown below.

(1) $(D_k-t_i-t_k) < (D_k-t_k) \leq (D_i-t_i-t_k) < (D_i-t_k) \leq (D_i-t_i)$

(2) $(D_k-t_i-t_k) < (D_i-t_i-t_k) \leq (D_k-t_k) < (D_i-t_k) \leq (D_i-t_i)$

As before,

$$L(i,k,t,T) = f_i(M_i) + f_k(N_k)$$

$$L(k,i,t,T) = f_k(M_k) + f_i(N_i)$$

There are five general cases to consider.

(1) If $t \leq D_k - t_i - t_k$ with $T \geq 0$, then $t_i + t_k + t - D_k \leq 0$ and hence $f_i(M_i) = f_k(M_k) = 0$.

In this case,

$$L(i,k,t,T) = f_k(N_k)$$

$$L(k,i,t,T) = f_i(N_i)\ .$$

For Lemma 2.1, it was necessary to assume that $f_i \geq f_k$. In this case, however, no comparison can be made between $L(i,k,t,T)$ and $L(k,i,t,T)$ under this assumption in that $N_k > N_i$. Hence, it will be assumed throughout this discussion that $f_i \leq f_k$. Consequently, $L(k,i,t,T) \leq L(i,k,t,T)$ for the case being considered.

(2) $(D_k - t_i - t_k) \leq t \leq (D_i - t_i - t_k)$ with $T \geq 0$.

In this case if f_k' is non-decreasing, then $L(k,i,t,T) \leq L(i,k,t,T)$ for all $T \geq 0$.

$$L(i,k,t,T) = f_k(N_k) \quad \text{since } M_i < 0$$

$$L(k,i,t,T) = f_k(M_k) + f_i(N_i) .$$

Clearly $L(k,i,t,T) \leq L(i,k,t,T)$ when $t \leq D_k - t_k$ (i.e. $M_k \leq 0$). If $t > D_k - t_k$, then rewrite the expression for $L(i,k,t,T)$:

$$L(i,k,t,T) = f_k(N_i) + f_k(N_k) - f_k(N_i)$$

Since $f_k(N_i) \geq f_i(N_i)$, it is sufficient to show that

$$f_k(N_k) - f_k(N_i) \geq f_k(M_k) .$$

This follows from the fact that

$$f_k(N_k) - f_k(N_i) \geq f_k(N_k-T) - f_k(N_i-T) = f_k(N_k-T)$$

(since $N_i - T \leq 0$ for $t \leq D_i - t_i - t_k$), f_k' is non-decreasing, and $N_k - T > M_k$. A formal proof results by applying the theorem of Appendix I.

(3) $(D_i - t_i - t_k) \leq t \leq D_i - t_k$ with $T \geq 0$.

As in the previous case, $L(k,i,t,T) \leq L(i,k,t,T)$ for all $T \geq 0$ if f_k' is non-decreasing:

$$L(i,k,t,T) = f_k(N_k)$$

$$L(k,i,t,T) = f_k(M_k) + f_i(N_i) .$$

If $t \leq D_k - t_k$, then $L(k,i,t,T) \leq L(i,k,t,T)$.

If $t > D_k - t_k$, then write

$$L(i,k,t,T) = f_k(N_i) + f_k(N_k) - f_k(N_i)$$

and it is sufficient to show that

$$(f_k(N_k) - f_k(N_i)) \geq f_k(M_k) .$$

This follows directly from the theorem of Appendix I in that

$$f_k(N_k) - f_k(N_i) \geq f_k(N_k-N_i) = f_k(D_i - D_k) ,$$

f_k' is non-decreasing, and $M_k \leq D_i - D_k$ for $t \leq D_i - t_k$.

(4) $D_i - t_k \leq t \leq D_i - t_i$ with $T \geq 0$.

Here, the above development would hold except that

$$(D_i - D_k) \leq M_k \leq (D_i - D_k) + (t_k - t_i) .$$

Consequently, no general comparison can be made between $(f_k(N_k) - f_k(N_i))$ and $f_k(M_k)$. In this case, then, there is no general inequality between $L(k,i,t,T)$ and $L(i,k,t,T)$.

(5) $t \geq D_i - t_i$ with $T \geq 0$.

$$L(i,k,t,T) = f_i(M_i) + f_k(N_k)$$
$$L(k,i,t,T) = f_k(M_k) + f_i(N_i)$$

Since $M_i = t_i + t - D_i < t_k + t - D_k = M_k$

$N_i = t_i + t_k + T + t - D_i < t_i + t_k + T + t - D_k = N_k$

$f_i \leq f_k$

it follows that $f_i(N_i) < f_k(N_k)$

and $f_i(M_i) < f_k(M_k)$.

Therefore, it is necessary to compare

$$f_k(M_k) - f_i(M_i) \quad \text{and} \quad f_k(N_k) - f_i(N_i)$$

or $$f_i(N_i) - f_i(M_i) \quad \text{and} \quad f_k(N_k) - f_k(M_k) \text{ .}$$

But $$N_i - M_i = t_k + T \geq t_i + T = N_k - M_k$$

and $$f_k \geq f_i \text{ .}$$

Further, $M_k > M_i$.

Hence, no general inequality exists between

$$L(i,k,t,T) \quad \text{and} \quad L(k,i,t,T) \text{ .}$$

The results of this analysis are summarized below:

When $D_k < D_i$, no comparison can be made between $L(i,k,t,T)$ and $L(k,i,t,T)$ unless $f_k \geq f_i$.

(a) For $f_k \geq f_i$, $L(k,i,t,T) \leq L(i,k,t,T)$ when $t \leq D_k - t_k$ and $T \geq 0$.

(b) For $f_k \geq f_i$, f_k' non-decreasing, $L(k,i,t,T) \leq L(i,k,t,T)$ when $t \leq D_i - t_k$ and $T \geq 0$.

(c) No general inequality exists for $t > D_i - t_k$ and $T \geq 0$.

The statement that no general inequality exists between $L(i,k,t,T)$ and $L(k,i,t,T)$ for certain ranges of values of t , T should be clarified:

For given f_i , f_k , t , T , there exist values of t_i , t_k , D_i , D_k for which $L(i,k,t,T) < L(k,i,t,T)$ and others for which $L(k,i,t,T) < L(i,k,t,T)$.

So far the discussion has centered on sufficient conditions for $L(i,k,t,T) \leq L(k,i,t,T)$ or vice-versa for various values of t , T . One important necessary condition for such a relation to hold is given below.

LEMMA 2.2 Given J_i and J_k with arbitrary t_i, t_k, D_i, D_k, f_i, and f_k; $L(i,k,t,T) \leq L(k,i,t,T)$ for all $T \geq 0$ only if $D_i \leq D_k$.

Proof: Suppose $D_k < D_i$ and, without loss of generality, $T = 0$. Let t be such that $t_i + t_k + t - D_i = 0$, that is $t = D_i - t_i - t_k$.

Hence, $t_i + t - D_i = M_i = -t_k < 0$

$$t_i + t_k + t - D_k = N_k = D_i - D_k > 0$$

But $t_k + t - D_k = M_k = D_i - D_k - t_i$

If $M_k < 0$,

$$L(i,k,t,T) = f_k (D_i - D_k) > 0$$

$$L(k,i,t,T) = 0$$

and the proof is complete, otherwise $D_i - D_k - t_i > 0$

or $D_i = D_k - \delta = t_i$ where $\delta > 0$.

Then, let $t = D_i - t_i - t_k - \delta$ and

$$L(i,k,t,T) = f_k (D_i - D_k - \delta) > 0$$

$$L(k,i,t,T) = 0$$

and hence, there exists t such that

$$L(k,i,t,T) < L(i,k,t,T) \text{ for arbitrary } T \geq 0.$$

The results obtained so far are summarized in the diagram on the next page.

These results are not particularly encouraging and should indicate the potential complexity of these problems. However, the fact that there are conditions for which $L(i,k,t,T) \leq L(k,i,t,T)$ for all $T \geq 0$ can be used in problems with a special property to obtain the main result in this chapter, a reduction theorem for decomposing the original problem into a series of subproblems.

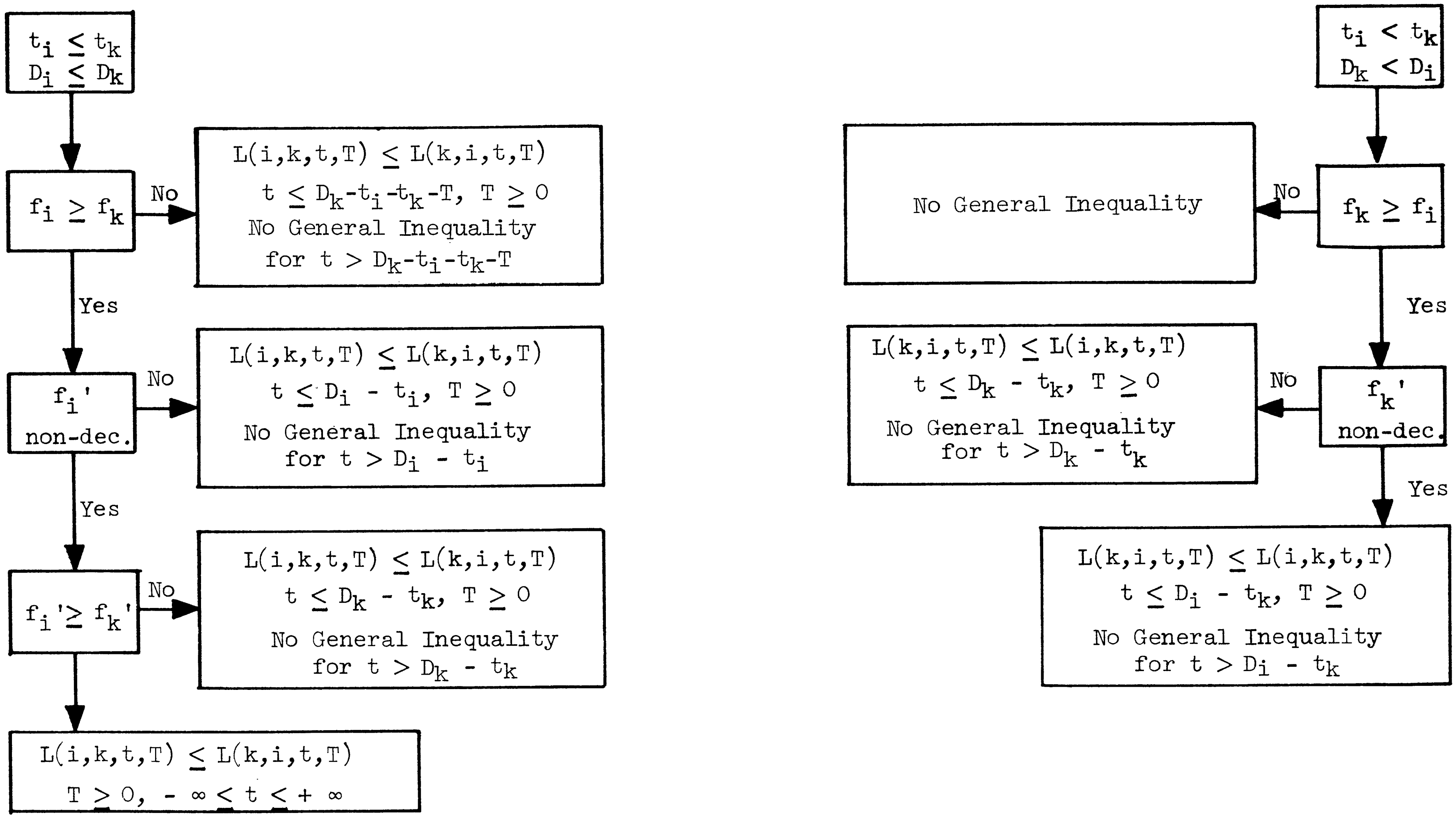

Diagram 1. Sequencing Rules for Two Jobs, J_i and J_k .

2.3 Reduction Theorem

Definition: Given a set of jobs, $J = \{J_1 \ldots J_n\}$ and values for $T \geq 0$ and t, a relation, $R(t,T)$, is defined on the set J as follows:

$$(J_i,J_k) \epsilon\ R(t,T) \text{ iff } L(i,k,t,T) \leq L(k,i,t,T)$$

Denote the fact that $(J_i,J_k) \epsilon\ R(t,T)$ by writing

$$J_i << (t,T)\ J_k \ .$$

Definition: Two jobs, J_i and J_k, are said to be time independent iff $J_i << (t,T)\ J_k$ for all $-\infty \leq t \leq +\infty$, $T \geq 0$ or $J_k << (t,T)\ J_i$ for all such t, T. Otherwise, the jobs are said to be time dependent.

Definition: Two jobs, J_i and J_k, are said to be adjacent time independent iff $J_i << (t,0) J_k$ or $J_k << (t,0) J_i$ for all t. Otherwise, the jobs are said to be adjacent time dependent.

It should be obvious that jobs which are time independent are adjacent time independent and those that are adjacent time dependent are time dependent.

When J_i and J_k are time independent with $J_i << (t,T) J_k$ for all $-\infty \leq t \leq +\infty$, $T \geq 0$, this is denoted by writing $J_i << J_k$. In like manner $J_i << (a)\ J_k$ denotes that J_i and J_k are adjacent time independent with $J_i << (t,0)\ J_k$. For a set of jobs, $J = \{J_1 \ldots Jn\}$, let R denote the relation defined by time independence and R' the relation defined by adjacent time independence.

One might expect that the flow diagram presented in the previous section would be simplified if it were assumed that $T = 0$ throughout the analysis. Unfortunately, no simplification arises if this assumption is made. However, the necessary conditions stated in Lemma 2.2 for time independence are also necessary for adjacent time independence. This can be seen by noting that the proof of the lemma does not depend on the value of T.

Definition: Given a schedule, $S = (J_{i_1} \ldots J_{i_n})$, for a set of jobs, $J = \{J_1 \ldots J_n\}$, with known processing times $t_1 \ldots t_n$, due-dates, $D_1 \ldots D_n$, and loss functions, $f_1 \ldots f_n$. Let $P(S,t)$ denote the total penalty incurred if the jobs are scheduled as in S with the first job commencing at time t where the due-dates are defined relative to $t = 0$.

$$P(S,t) = \sum_{r=1}^{n} f_{i_r}\left(t + \sum_{m=1}^{r} t_{i_m} - D_{i_r}\right)$$

LEMMA 2.3 Given a schedule, $S = (J_{i_1} \ldots J_{i_{k-1}}, J_{i_k}, J_{i_{k+1}} \ldots J_{i_n})$ and a schedule $S' = (J_{i_1} \ldots J_{i_{k-1}}, J_{i_{k+1}}, J_{i_k} \ldots J_{i_n})$, then

$P(S',t) \leq P(S,t)$ iff

$$L(i_{k+1}, i_k, t + \sum_{r=1}^{k-1} t_{i_r}, 0) \leq L(i_k, i_{k+1}, t + \sum_{r=1}^{k-1} t_{i_r}, 0)$$

Proof: $$P(S,t) = P((J_{i_1} \ldots J_{i_{k-1}}), t) + L(i_k, i_{k+1}, t + \sum_{r=1}^{k-1} t_{i_r}, 0)$$

$$+ P((J_{i_{k+2}} \ldots J_{i_n}), t + \sum_{r=1}^{k+1} t_{i_r})$$

and $$P(S',t) = P(S,t) + (L(i_{k+1}, i_k, t + \sum_{r=1}^{k-1} t_{i_r}, 0)$$

$$- L(i_k, i_{k+1}, t + \sum_{r=1}^{k-1} t_{i_r}, 0)).$$

Hence $P(S',t) \leq P(S,t)$ iff $L(i_{k+1}, i_k, t + \sum_{r=1}^{k-1} t_{i_r}, 0)$

$$\leq L(i_k, i_{k+1}, t + \sum_{r=1}^{k-1} t_{i_r}, 0)$$

Corollary: If $J_{i_{k+1}} \ll (a)\ J_{i_k}$, then $P(S',t) \leq P(S,t)$ for all t .

Proof: $L(i_{k+1}, i_k, t, 0) \leq L(i_k, i_{k+1}, t, 0)$ for all t .

LEMMA 2.4 Given a schedule, $S = (J_1 \ldots J_n)$ in which there are two jobs, J_i and J_k , and $i < k$ but $J_k \ll J_i$ with $t_k \leq t_i$. The schedule S' , obtained by interchanging J_i and J_k in the schedule S , is such that $P(S',t) \leq P(S,t)$.

Proof: $P(S,t) = f_1(t + t_1 - D_1) + f_2(t + t_1 + t_2 - D_2) +$

$$\ldots + f_i(t + \sum_{q=1}^{i} t_q - D_i) + \ldots + f_k(t + \sum_{q=1}^{k} t_q - D_k) +$$

$$\ldots + f_n(t + \sum_{q=1}^{n} t_q - D_n)$$

$$= L_1 + L_2 + \ldots + L_n$$

$$P(S',t) = L_1 + L_2 + \ldots + L_{i-1} + f_k\ (t + t_k + \sum_{q=1}^{i-1} t_q - D_k)$$

$$+ f_{i+1}\ (t + t_k + \sum_{q=1}^{i-1} t_q + t_{i+1} - D_{i+1}) +$$

$$\ldots + f_{k-1}\ (t + t_k + \sum_{q=1}^{i-1} t_q + \sum_{r=i+1}^{k-1} t_r - D_{k-1})$$

$$+ f_i \left(t + t_k + \sum_{q=1}^{i-1} t_q + \sum_{r=i+1}^{k-1} t_r + t_i - D_i\right)$$

$$+ L_{k+1} + \ldots + L_n$$

$$= L_1 + L_2 + \ldots + L_{i-1} + L'_k + L'_{i+1} + \ldots + L'_{k-1}$$

$$+ L'_i + L_{k+1} + \ldots + L_n$$

But $L'_k + L'_i \leq L_i + L_k$ since

$$L(k,i,t,T) \leq L(i,k,t,T) \quad \text{for all} \quad T \geq 0 \ .$$

And since $t_k \leq t_i$ and the f_i's are non-decreasing,

$$L'_{i+j} \leq L_{i+j} \quad \text{for} \quad j = 1 \ldots (k-1-i) \ .$$

Hence $P(S',t) \leq P(S,t)$.

At this point it is necessary to assume that the scheduling problems to be considered have the property that R' is a transitive relation. While this property is quite reasonable, it is not obvious that all scheduling problems in the general class previously defined have it. However, the author has not been able to find a numerical example where R' is not transitive.

LEMMA 2.5 Given a set of jobs $J = \{J_1 \ldots J_n\}$ where J_i and J_k are adjacent time dependent but are adjacent time independent with all other jobs in the set, $J' = J - \{J_i, J_k\}$, then J' can be partitioned into two sets, B and A , such that

$$J_q \in B \text{ iff } J_q \ll (a)\ J_i \quad \text{and} \quad J_q \ll (a)\ J_k$$
$$J_q \in A \text{ iff } J_i \ll (a)\ J_q \quad \text{and} \quad J_k \ll (a)\ J_q$$

Proof: Consider any job $J_q \in J'$. Since J_i and J_q are adjacent time independent, either $J_i \ll (a)\ J_q$ or $J_q \ll (a)\ J_i$.

(1) If $J_i \ll (a)\ J_q$, then consider J_k and J_q. As in the case of J_i, either $J_k \ll (a)\ J_q$ or $J_q \ll (a)\ J_k$. If $J_k \ll (a)\ J_q$, then $J_q \in B$. If $J_q \ll (a)\ J_k$, then $J_i \ll (a)\ J_q \ll (a)\ J_k$ and $J_i \ll (a)\ J_k$ since R' is transitive. But J_i and J_k are adjacent time dependent and we have a contradiction. Hence, $J_k \ll (a)\ J_q$.

(2) If $J_q \ll (a)\ J_i$, then the same type of argument as above can be used to show that $J_q \ll (a)\ J_k$ and hence $J_q \in A$.

Definition: For a set of jobs, $J = \{J_1 \ldots J_n\}$, a pair of jobs, J_i and J_k, are said to be <u>adjacent time dependently associated</u> if there exists a sequence of jobs, $(J_{i_1} \ldots J_{i_p})$, $J_{i_q} \in J$ for $q = i \ldots p$, such that (J_i, J_{i_1}), $(J_{i_1}, J_{i_2}) \ldots (J_{i_{p-1}}, J_{i_p})$, (J_{i_p}, J_k) are adjacent time dependent pairs. Such a sequence of jobs, $(J_i, J_{i_1}, \ldots, J_{i_p}, J_k)$, is called an <u>adjacent time dependent string</u>.

Define the relation, D, on the set J as follows:

(1) If $J_i \in J$ then $(J_i, J_i) \in D$.

(2) If $J_i, J_k \in J$ are adjacent time dependently associated, then (J_i, J_k) and $(J_k, J_i) \in D$.

LEMMA 2.6 D is an equivalence relation.

Proof: Since D is reflexive and symmetric by definition, it is only necessary to show that D is transitive. If (J_i, J_k) and $(J_k, J_q) \in D$, then there exist adjacent time dependent strings J_i, $J_{i_1}, \ldots, J_{i_p}$ J_k and J_k, J_{k_1}, $\ldots$, J_{k_s}, J_q.

Thus J_i , J_{i_1} , ... , J_{i_p} , J_k , J_{k_1} , ... , J_{k_s} , J_q is an adjacent time dependent string and $(J_i,J_q) \in D$.

Definition: Associated with the equivalence relation D is a partition of the set, J , into a set of equivalence classes. Such an equivalence class will be called an adjacent time dependent equivalence class.

LEMMA 2.7 Given a set of jobs, $J = \{J_1 \dots J_n\}$, and a proper subset, $J' = \{J_{p_1} \dots J_{p_m}\}$, which is an adjacent time dependent equivalence class. The remaining jobs in J can be partitioned into two sets, B and A , such that

$$J_i \in B \text{ iff } J_i \ll (a)\ J_{p_1} \dots J_i \ll (a)\ J_{p_m}$$

$$J_i \in A \text{ iff } J_{p_1} \ll (a)\ J_i \dots J_{p_m} \ll (a)\ J_i \ .$$

Proof: Consider any job, $J_i \in J - J'$.

Since J' is an adjacent time dependent equivalence class, J_i is adjacent time independent with each job in J' . In particular, either $J_i \ll (a)\ J_{p_1}$ or $J_{p_1} \ll (a)\ J_i$.

If $J_i \ll (a)\ J_{p_1}$, consider any other job, $J_{p_k} \in J'$. By definition of an adjacent time dependent equivalence class there exists an adjacent time dependent string of the form $(J_{p_1}, J_{q_1}, \dots J_{q_s}, J_{p_k})$ where J_{q_1} , ... , $J_{q_s} \in J'$.

But $(J_{p_1}, J_{q_1}) \dots (J_{q_s}, J_{p_k})$ are adjacent time dependent pairs.

Hence, apply Lemma 2.5 to each pair in the order given to obtain:

$$J_i \ll (a)\ J_{p_1}\ , J_i \ll (a)\ J_{q_1} \dots J_i \ll (a)\ J_{q_s}\ ,$$

$$J_i \ll (a)\ J_{p_k}\ .$$

Hence $J_i \ll (a)\ J_{p_k}$ for arbitrary $J_{p_k} \in J'$.

In like manner, if $J_{p_i} \ll (a)\ J_i$, the same argument can be used to show that $J_{p_k} \ll (a)\ J_i$ for arbitrary $J_{p_k} \in J'$.

Consequently each job, $J_i \in J - J'$, will be uniquely assigned to either set B or set A .

Remarks: Since it was assumed that the relation defined by adjacent time independence is transitive, it should be obvious that for any pair of jobs, $J_i \in B$ and $J_k \in A$, $J_i \ll (a)\ J_k$.

LEMMA 2.8 Given a set of jobs, $J = \{J_1 \ldots J_n\}$, and a proper subset, $J' = \{J_{p_1} \ldots J_{p_m}\}$, which is an adjacent time dependent equivalence class. Any schedule, S , in which $J_{p_1}, \ldots, J_{p_m}$ are not adjacent can be transformed into a schedule S' in which $J_{p_1}, \ldots, J_{p_m}$ are adjacent such that $P(S',t) \leq P(S,t)$.

Proof: Without loss of generality, assume S is of the form:

$$(J_1, \ldots, J_{p_1}, \ldots, J_{p_2}, \ldots, J_{p_m}, \ldots J_n)$$

where jobs $J_{i_1}, \ldots, J_{i_s}$ lie between jobs $J_{p_1}, \ldots, J_{p_m}$.
Using Lemma 2.7, each job in the set, $\{J_{i_1} \ldots J_{i_s}\}$ is a member of either the set B or the set A defined by the set of jobs J' . There are three general cases to consider.

(1) If $J_{i_1} \in B$, then $J_{i_1} \ll (a)\ J_{p_k}$ for $k = 1 \ldots m$.
In this case, define a new schedule, S_1 , of the form
$(J_1, \ldots, J_{i_1}, J_{p_1}, \ldots, J_{p_2}, \ldots, J_{p_m}, \ldots, J_n)$
and $P(S_1,t) \leq P(S,t)$ by repeated application of the corollary to Lemma 2.3. Now operate on S_1 .

(2) If (1) does not apply, then $J_{i_1} \in A$ and consider J_{i_s}. If $J_{i_s} \in A$, then $J_{p_k} \ll (a)\ J_{i_s}$ for $k = 1 \ldots m$ and a new schedule, S_1, can be defined of the form

$(J_1, \ldots, J_{p_1}, \ldots, J_{p_2}, \ldots, J_{p_m}, J_{i_s}, \ldots, J_n)$.

As in the previous case, $P(S_1,t) \leq P(S,t)$ by repeated applications of the corollary to Lemma 2.3. Now operate on S_1.

(3) If neither (1) or (2) apply, then $J_{i_1} \in A$ and $J_{i_s} \in B$. Hence, there must exist a pair of jobs, J_{i_b} and $J_{i_{b+1}}$ such that $J_{i_b} \in A$ and $J_{i_{b+1}} \in B$. But $J_{i_{b+1}} \ll (a)\ J_{p_k}$ for $k = 1 \ldots m$ and $J_{p_k} \ll (a)\ J_{i_b}$ for $k = 1 \ldots m$. Further, $J_{i_{b+1}} \ll (a)\ J_{i_b}$. Hence, J_{i_b} may be moved to the right by successive interchanges to obtain a schedule, S_1, of the form, $(J_1, \ldots, J_{p_1}, \ldots, J_{i_{b+1}}, J_{i_b}, \ldots, J_{p_m}, \ldots, J_n)$ and $P(S_1,t) \leq P(S,t)$ by repeated application of the corollary to Lemma 2.3. Now operate on S_1.

At each stage, either one of the jobs in the set, $\{J_{i_1} \ldots J_{i_s}\}$, is eliminated from the subsequence $(J_{p_1} \ldots J_{p_m})$, or a pair of jobs in the set, $\{J_{i_1} \ldots J_{i_s}\}$, is interchanged so that the set is one interchange closer to being ordered in such a way that the jobs, $J_{i_k} \in B$, precede the jobs, $J_{i_q} \in A$. Since there are a finite number of jobs, the process terminates with a schedule, S_r, of the form $(J_1, \ldots, J'_{i_1}, \ldots, J'_{i_q}, J_{p_1}, \ldots, J_{p_m}, J'_{i_{q+1}}, \ldots, J'_{i_s}, \ldots, J_n)$ in which $J_{p_1} \ldots J_{p_m}$ are adjacent and $J'_{i_k} \in B$ for $k = 1 \ldots q$ and $J'_{i_k} \in A$ for $k = (q+1) \ldots s$.

Now let $S' = S_r$ and $P(S',t) = P(S_r,t) \leq \ldots \leq P(S,t)$.

LEMMA 2.9 Given a set of jobs, $J = \{J_1 \ldots J_n\}$, with a proper subset, $J' = \{J_{p_1} \ldots J_{p_m}\}$, which is an adjacent time dependent equivalence class. Any schedule, S, can be transformed into a schedule of the form $S' = (B, J', A)$ where B and A are defined for the equivalence class, J', where $P(S',t) \leq P(S,t)$.

Proof: Consider a schedule, S, and assume that S is not of the form (B, J', A). If the jobs in the set, J', are not adjacent in the schedule, S, apply Lemma 2.8 to obtain a new schedule, S_1, in which they are adjacent, and S_1 will be of the form (B_1, J', A_1) with $P(S_1,t) \leq P(S,t)$. There are now two steps that must be performed to obtain the schedule of the desired form.

1. If $B_1 \subset B$, go to step 2. Otherwise, consider the set of jobs $B_1 \cap A$. Pick the last such job, J_q, in the current schedule and clearly $J_i \ll (a)\ J_q$ for all $J_i \in B$ and $J_{p_k} \ll (a)\ J_q$ for all $J_{p_k} \in J'$. Now move the job, J_q, to the right by successive interchanges until a job, J_r, is encountered where $J_r \in A$.

Since J_q was the last job in the current schedule which was a member of $B_1 \cap A$, the new schedule, S_2, so defined most be of the form, (B_2, J', A_2) where

$$B_2 = B_1 - \{J_q\}$$

$$A_2 = A_1 \cup \{J_q\}$$

and $P(S_2,t) \leq P(S_1,t)$ by repeated applications of the corollary to Lemma 2.3. Now use S_2 as the current schedule and repeat this step. This process terminates in a finite number of steps with a schedule, S_r, where S_r is of the form (B_r, J', A_r) and $B_r \subset B$ with $P(S_r,t) \leq \ldots \leq P(S_1,t) \leq P(S,t)$.

2. Consider the set A_r and if $A_r \cap B = \emptyset$, the schedule S_r is of the form (B, J', A) and the proof is complete. If $A_r \cap B \neq \emptyset$ pick the first job, J_q, in the current schedule where $J_q \in A_r \cap B$. Clearly $J_q \ll (a)\ J_i$ for all $J_i \in A$ and $J_q \ll (a)\ J_{p_k}$ for all $J_{p_k} \in J'$. Now move the job J_q to the left by successive interchanges until a job, J_r, is encountered where $J_r \in B$. Since J_q was the first job in the current schedule which was a member of $A_r \cap B$, the new schedule, S_{r+1}, so defined must be of the form (B_{r+1}, J', A_{r+1}) where

$$A_{r+1} = A_r - \{J_q\}$$

$$B_{r+1} = B_r \cup \{J_q\}$$

and $P(S_{r+1},t) \leq P(S_r,t)$ by repeated applications of the corollary to Lemma 2.3. Now use S_{r+1} as the current schedule and repeat this step. This process terminates in a finite number of steps with a schedule, S_{r+s}, where S_{r+s} is of the form (B_{r+s}, J', A_{r+s}) and $A_{r+s} \cap B = \emptyset$. But $B_{r+s} \cap A = \emptyset$ and hence S_{r+s} is of the form (B, J', A) with $P(S_{r+s},t) \leq \ldots \leq P(S_{r+1},t) \leq \ldots \leq P(S_1,t) \leq P(S,t)$.

Definition: Given a set of jobs, $J = \{J_1 \ldots J_n\}$, for which there are at least two adjacent time dependent equivalence classes. For two such classes, E_i and E_k, we say that E_i precedes E_k, denoted $E_i \ll E_k$, if $J_q \in E_i \ll (a)\ J_r \in E_k$.

Remark: The relation "precedes" as defined above is obviously transitive and anitsymmetric. In fact, for a set of m equivalence classes for a set of jobs, J, this relation constitutes a well ordering of the equivalence classes.

LEMMA 2.10 Given a set of jobs, $J = \{J_1 \dots J_n\}$ for which there are at least two adjacent time dependent equivalence classes, E_i and E_k, where $E_i \ll E_k$. Then any schedule, S, can be transformed into a schedule of the form

$$S' = (B_i, E_i, (A_i \cap B_k), E_k, A_k)$$

where B_i and A_i correspond to E_i and B_k and A_k correspond to E_k with $P(S',t) \leq P(S,t)$.

Proof: Since $E_i \ll E_k$, $B_i \subset B_k$ and $A_k \subset A_i$.
By Lemma 2.9 the schedule, S, can be transformed into a schedule of the form, $S_1 = (B_i, E_i, A_i)$ where $E_k \subset A_i$ since $E_i \ll E_k$ and $P(S_1,t) \leq P(S,t)$.

Now consider the set A_i. Using Lemma 2.9 and the principle of optimality, the schedule S_1, can be transformed into a schedule of the form, $S_2 = (B_i, E_i, B_k', E_k, A_k')$, where B_k' and A_k' are defined for E_k relative to the set of jobs A_i. But, $B_k' = B_k \cap A_i$ and $A_k' = A_k \cap A_i$. Hence the schedule, S_2, can be transformed into a schedule of the form,

$$S' = (B_i, E_i, (A_i \cap B_k), E_k, (A_i \cap A_k))$$

or $$S' = (B_i, E_i, (A_i \cap B_k), E_k, A_k)$$

and $$P(S',t) \leq P(S_2,t) \leq P(S_1,t) \leq P(S,t).$$

THEOREM 2.1 (Reduction Theorem): Given a set of jobs, $J = \{J_1 \dots J_n\}$, for which there are m adjacent time dependent equivalence classes, $E_1 \dots E_m$. Any schedule, S, can be transformed into a schedule of the form, $S' = (E_{i_1}, E_{i_2}, \dots E_{i_s})$ where

$$E_{i_1} \ll E_{i_2} \ll .. \ll E_{i_s} \text{ and } P(S',t) \leq P(S,t) .$$

Proof: Using Lemma 2.10, the schedule, S , can be transformed into a schedule of the form,

$$S_1 = (B_{i_1}, E_{i_1} , (A_{i_1} \cap B_{i_2}), E_{i_2}, A_{i_2})$$

where $P(S_1,t) \leq P(S,t)$.

However, since $E_{i_1} \ll E_{i_2} \ll \ldots \ll E_{i_s}$, it follows that there does not exist a job, $J_q \in J$, such that for some job, $J_r \in E_{i_1}$, $J_q \ll (a) J_r$ and $J_q \notin E_{i_1}$. Hence, $B_{i_1} = \emptyset$. Further, since the set of E_{i_k}'s constitutes a partition of the set J , there cannot exist a job, $J_q \in J$, where for jobs $J_r \in E_{i_1}$ and $J_s \in E_{i_2}$, $J_r \ll (a) J_q \ll (a) J_s$ and $J_q \notin E_{i_1}$ or E_{i_2} . Consequently, S_1 is of the form $(E_{i_1}, E_{i_2}, A_{i_2})$. Now operate on A_{i_2} to obtain a schedule of the form,

$$S_2 = (E_{i_1} , E_{i_2} , E_{i_3} , E_{i_4} , A_{i_4}) \text{ and}$$

$$P(S_2,t) \leq P(S_1,t) \leq P(S,t), \text{ etc.}$$

This process terminates in a finite number of steps with the schedule $S_q = S'$ and

$$P(S',t) = P(S_q,t) \leq \ldots \leq P(S_1,t) \leq P(S,t) .$$

Corollary: If R' is a well ordering of the jobs, then the schedule defined by R' is optimal.

Proof: If R' is a well ordering, there exists a sequence of jobs, $J_{i_1} \ll (a) J_{i_2} \ll (a) \ldots \ll (a) J_{i_n}$, and the schedule $(J_{i_1} \ldots J_{i_n})$ is clearly optimal.

The results obtained by McNaughton(9) and Smith(15) on scheduling jobs with linear deferral costs follow directly from the corollary using the following lemma.

LEMMA 2.11 If $L(i,k,t,0) = a_i(t_i + t) + a_k(t_i + t_k + t)$ for all J_i, J_k, $t \geq 0, = 0$, $t < 0$ then $J_i << (a) J_k$ iff $\frac{t_i}{a_i} \leq \frac{t_k}{a_k}$.

Proof:

$$L(i,k,t,0) = a_i\ (t_i + t) + a_k(t_i + t_k + t)$$
$$L(k,i,t,0) = a_k\ (t_k + t) + a_i(t_i + t_k + t)$$
$$L(k,i,t,0) - L(i,k,t,0) = a_i t_k - a_k t_i\ .$$

Hence, R' is a well ordering of the jobs and the result follows from the corollary to the reduction theorem.

Incidentally, this type of problem provides an excellent example of where two jobs that are adjacent time independent are not time independent. Consider J_i and J_k where $t_i/a_i \leq t_k/a_k$ but $a_k > a_i$. Assuming that t is sufficiently large,

$$L(k,i,t,T) - L(i,k,t,T) = a_i t_k - a_k t_i + (a_i - a_k)\ T\ ,$$

which is negative for sufficiently large T.

This theorem, then, demonstrates that there will be at least one optimal schedule of the form specified above. This reduces the problem to one of determining the optimal sequence within each adjacent time dependent equivalence class. One property that such an optimal sequence will have was given by Lemma 2.4. There are other properties, however, which at least one optimal schedule will possess if additional assumptions are made concerning the f_i's. These are the subject of the next chapter.

CHAPTER 3

SEQUENCING PROBLEMS WITH IDENTICAL LOSS FUNCTIONS

3.1 Introduction

The problem of sequencing two jobs, J_i and J_k, with known processing times, t_i and t_k, due-dates, D_i and D_k, and loss functions, f_i and f_k, was considered in the previous chapter. For $t_i \leq t_k$ and $D_i \leq D_k$, it was shown that $f_i \geq f_k$, $f_i' \geq f_k'$ and f_i' non-decreasing were sufficient conditions for $J_i \ll J_k$. When $t_i < t_k$ and $D_k < D_i$, however, $f_k \geq f_i$ was a sufficient condition for $J_k \ll (t,T)J_i$ for limited values of t. Hence, it would appear that a more structured problem might emerge if the assumption were made that the loss functions for the set of jobs to be sequenced are the same.

This chapter, then, considers the general problem posed in the previous chapter in the special case where $f_1 = \ldots = f_n = f$. The assumptions of that chapter concerning f will be retained and the additional assumption made that f' is non-decreasing. The latter means that at least as much penalty is incurred during the n-th unit of lateness for a job as was incurred during the $(n-1)^{st}$ unit of lateness.

3.2 Two Job Sequencing Rules

Under the assumption that the loss functions are identical and that f' is non-decreasing, the results of the previous chapter concerning the relation between $L(i,k,t,T)$ and $L(k,i,t,T)$ can be refined. There are two general cases to consider.

(1) If $t_i \leq t_k$ and $D_i \leq D_k$, then

$L(i,k,t,T) \leq L(k,i,t,T)$ for all $T \geq 0$.

Proof: For $f_i = f_k = f$, it follows that $f_i \geq f_k$ and $f_i' \geq f_k'$. Hence, the sufficient conditions stated in the previous chapter for $J_i \ll J_k$ are satisfied.

Consequently, since $t_i \leq t_k$ and $D_i \leq D_k$ are sufficient conditions for time independence, $t_i < t_k$ and $D_k < D_i$ or vice-versa are necessary conditions for time dependence.

(2) If $t_i < t_k$ and $D_k < D_i$, the results of the previous chapter obtained when $f_k \geq f_i$ hold since $f_k = f_i = f$. Consequently,

$$L(k,i,t,T) \leq L(i,k,t,T) \quad \text{for} \quad t \leq D_i - t_k\ ,\ T \geq 0\ .$$

Unfortunately, there is still no general relation that can be derived for $t > D_i - t_k$, $T \geq 0$.

Further, the situation does not change if one considers the special case where $T = 0$. The question is, then, if $t_i < t_k$ and $D_k < D_i$, what are sufficient conditions for the time dependence or adjacent time dependence of J_i and J_k. To determine such conditions, recall

$$L(i,k,t,T) = f(M_i) + f(N_k)$$

$$L(k,i,t,T) = f(M_k) + f(N_i)$$

and $L(k,i,t,T) \leq L(i,k,t,T)$ when $t \leq D_i - t_k$. For J_i and J_k to be time dependent, it is necessary to find t and $T \geq 0$ such that $L(i,k,t,T) < L(k,i,t,T)$. Sufficient conditions for the existence of such t, T are expressed in the lemma below.

LEMMA 3.1 Given two jobs, J_i and J_k, where $t_i < t_k$, $D_k < D_i$, and $f_i = f_k = f$. If $\lim \dfrac{f'(y)}{f'(x+y)} = 1$ as $y \to \infty$ for arbitrary $0 < x < \infty$, then J_i and J_k are time dependent.

Proof: Assume that $L(k,i,t,T) \leq L(i,k,t,T)$ for $t = D_i - t_i$

and $$L(i,k,(D_i - t_i),T) = f(t_k + T + (D_i - D_k))$$

$$L(k,i,(D_i - t_i),T) = f(t_k - t_i + D_i - D_k) + f(t_k + T)$$

Now let $$x_1 = \min((t_k - t_i + D_i - D_k), (t_k + T))$$

$$x_2 = \max((t_k - t_i + D_i - D_k), (t_k + T))$$

$$x_3 = t_k + T + (D_i - D_k)$$

and $$x_1 \leq x_2 < x_3 , \ x_1 + x_2 > x_3 .$$

Hence, $$L(i,k, (D_i - t_i), T) = f(x_3)$$

$$L(k,i, (D_i - t_i), T) = f(x_1) + f(x_2)$$

Consider $t > D_i - t_i$ and let $y = t - (D_i - t_i)$

Hence, $$L(i,k,t,T) = f(x_3 + y) + f(y)$$

and $$L(k,i,t,T) = f(x_1 + y) + f(x_2 + y)$$

The question now is under what conditions does there exist $y > 0$ such that

$$f(x_1 + y) + f(x_2 + y) > f(x_3 + y) + f(y) .$$

It is shown in Appendix II that a sufficient condition for this to hold is that

$$\lim_{y \to \infty} \frac{f'(y)}{f'(x+y)} = 1 \quad \text{for} \quad 0 < x < \infty .$$

It should be pointed out that the above proof does not depend on the value of T. Hence, the sufficient conditions stated for time dependence are also sufficient for adjacent time dependence.

Hence, for loss functions having the property that

$$\lim_{y \to \infty} \frac{f'(y)}{f'(x+y)} = 1 \quad \text{for } 0 < x < \infty ,$$

a necessary and sufficient condition for the time dependence (and adjacent time dependence) of two jobs is $t_i < t_k$ and $D_k < D_i$ or vice-versa.

The class of functions having the property is quite broad. For example, all polynomials with a finite number of terms belong to this class. Functions such as an exponential function, however, do not.

LEMMA 3.2 The reduction theorem can be applied to sequencing problems in which all the loss functions are the same function, f , and f' is non-decreasing with

$$\lim_{y \to \infty} \frac{f'(y)}{f'(x+y)} = 1 \quad \text{for} \quad 0 < x < \infty$$

Proof: It is sufficient to show that the relation defined by adjacent time independence is transitive. Consider $J_i \ll (a)\, J_k$ and $J_k \ll (a)\, J_p$. Since $t_i \leq t_k$, $D_i \leq D_k$ and $t_k \leq t_p$, $D_k \leq D_p$ are necessary and sufficient conditions for these adjacent time independence relations to hold, it follows that $t_i \leq t_p$, $D_i \leq D_p$ and thus $J_i \ll (a)\, J_p$.

This, then, defines one rather broad class of sequencing problems for which the reduction theorem holds.

3.3 Single-Time Dependent Jobs

Definition: Two jobs, J_i and J_k , are said to be single-time dependent if there exists t^* such that for arbitrary $T \geq 0$, $L(i,k,t,T) \leq L(k,i,t,T)$ for $t \leq t^*$ and $L(k,i,t,T) < L(i,k,t,T)$ for $t > t^*$ or vice-versa.

In order to investigate the question of sufficient conditions for two jobs to be single-time dependent, consider J_i and J_k where $t_i < t_k$ and $D_k < D_i$. If there exists t' such that $L(i,k,t',T) < L(k,i,t',T)$ then $t' > D_i - t_k$ since $L(k,i,t,T) \leq L(i,k,t,T)$

for $t \leq D_i - t_k$. The question, then, is what are sufficient conditions for $L(i,k,t,T) \leq L(k,i,t,T)$ for all $t \geq t'$. There are two general cases to consider.

(1) Consider $D_i - t_k < t' < D_i - t_i$.

a. For $t' \leq t \leq D_i - t_i$,

$$L(i,k,t,T) = f(t_i + t_k + t + T - D_k)$$

$$L(k,i,t,T) = f(t_k + t - D_k) + f(t_i + t_k + t + T - D_i)$$

Let $x_1 = \min((t_i + t_k + t' + T - D_i),(t_k + t' - D_k))$

$x_2 = \max((t_i + t_k + t' + T - D_i),(t_k + t' - D_k))$

$x_3 = t_i + t_k + t' + T - D_k$

$y = t - t'$

and $0 < x_1 \leq x_2 < x_3$.

Now $L(i,k,t,T) = f(x_3 + y)$

$L(k,i,t,T) = f(x_1 + y) + f(x_2 + y)$

The question is what are sufficient conditions for $f(x_1 + y) + f(x_2 + y) > f(x_3 + y)$ for $0 < y < \infty$. Appendix III shows that f'/f non-increasing is such a condition.

b. Now consider $t > D_i - t_i$. For f'/f non-increasing, $L(k,i,t,T) \geq L(i,k,t,T)$ for $t' \leq t \leq D_i - t_i$.

In particular, $L(k,i,D_i-t_i,T) \geq L(i,k,D_i-t_i,T)$

or $L(k,i,D_i-t_i,T) = L(i,k,D_i-t_i,T) + \lambda$

where $\lambda \geq 0$. By assuming that $\lambda = 0$,

(i.e. $t' = D_i - t_i$) this becomes a special case of of the second general case and is treated below.

(2) Now consider $t' \geq D_i - t_i$. For $t \geq t'$

$$L(i,k,t,T) = f(t_i + t - D_i) + f(t_i + t_k + t + T - D_k)$$

$$L(k,i,t,T) = f(t_k + t - D_k) + f(t_i + t_k + t + T - D_i)$$

Let

$$x_1 = \min((t_i + t_k + t' + T - D_i), (t_k + t' - D_k))$$

$$x_2 = \max((t_i + t_k + t' + T - D_i), (t_k + t' - D_k))$$

$$x_3 = t_i + t_k + t' + T - D_k$$

$$x_4 = t_i + t' - D_i$$

$$y = t - t'$$

Hence $0 \leq x_4 < x_1 \leq x_2 < x_3$ and $x_1 + x_2 > x_3 + x_4$

$$L(i,k,t,T) = f(x_4 + y) + f(x_3 + y)$$

$$L(k,i,t,T) = f(x_1 + y) + f(x_2 + y)$$

and the question is under what conditions is

$$f(x_1 + y) + f(x_2 + y) > f(x_4 + y) + f(x_3 + y)$$

for all $y \geq 0$.

It should be obvious now that (1) b. is a special case of this situation by simply considering $t' = D_i - t_i$ (i.e. $x_4 = 0$) .

The author has been unable to determine a "nice" set of sufficient conditions for which

$$f(x_1 + y) + f(x_2 + y) > f(x_4 + y) + f(x_3 + y)$$

for all $y \geq 0$. The class of functions for which this does hold is likely to be rather limited in that even most polynomials do not appear to belong to this class. Appendix IV contains a proof that general quadratic functions with positive coefficients have this property, but the following counter example illustrates that general cubic equations do not.

It should be noted that since the results obtained above do not depend on the actual value that T assumes, no simplification arises if $T = 0$.

Counter example for cubic equations

Let $f(x) = 10x + x^3$ and assume $x_1 = x_2 = 1$ and $x_4 = 0$. (The example will illustrate that x_1 and x_2 need not be equal and that x_4 need not be zero. These values were selected to simplify the computation.)

$$f(x_1) + f(x_2) = 22$$

and $$f(x_3) + f(x_4) = 10x_3 + x_3^3$$

Solving the equation $10x_3 + x_3^3 = 22$ by radicals,

$$x_3 = \sqrt[3]{\frac{-22}{2} + \frac{1}{2}\sqrt{(22)^2 + 4\left(\frac{10}{3}\right)^3}} - \sqrt[3]{\frac{-22}{2} - \frac{1}{2}\sqrt{(22)^2 + 4\left(\frac{10}{3}\right)^3}}$$

and $x_3 \cong 1.703$.

Hence, let $x_3 = 1.7$ and clearly

$$0 \leq x_4 < x_1 \leq x_2 < x_3 \quad \text{where} \quad x_1 + x_2 > x_3 + x_4$$

with $$f(x_1) + f(x_2) > f(x_3) + f(x_4)$$

Now consider $y = 1$ and

$$f(x_1 + y) + f(x_2 + y) = 56 < f(x_3 + y) + f(x_4 + y) = 57.68$$

The following lemma is based on these results.

LEMMA 3.3 If two jobs, J_i and J_k, have processing times, $t_i < t_k$, due-dates, $D_k < D_i$ and loss functions $f_i = f_k = f$ of the form $ax + bx^2$ $(a,b \geq 0)$, then J_i and J_k are single-time dependent.

Proof: Consider $f' = a + 2bx$ and for $0 \leq x' < \infty$

$$\lim_{y \to \infty} \frac{f'(x'+y)}{f'(y)} = \lim_{y \to \infty} \frac{a + 2b(x'+y)}{a + 2by}$$

$$= \lim_{y \to \infty} \frac{\frac{a}{y} + \frac{2bx'}{y} + 2b}{\frac{a}{y} + 2b} = \frac{0 + 0 + 2b}{0 + 2b} = 1$$

Hence, J_i and J_k are time dependent by Lemma 2.1.

$$\frac{f'}{f} = \frac{a+2bx}{ax+bx^2} = \frac{\frac{a}{x}+2b}{a+bx}$$

is clearly non-increasing and f is of the form, $ax + bx^2$. Therefore, the conditions of both case 1 and 2 are satisfied and J_i and J_k are single-time dependent.

3.4 Further Characterization Results

Definition: Let J_i and J_k be single-time dependent jobs where $t_i < t_k$ and $D_k < D_i$. For fixed $T \geq 0$, let $\mathcal{T}_{k,i}(T)$ denote the value of t for which

$$L(k,i,t,T) \leq L(i,k,t,T) \quad \text{for} \quad t \leq \mathcal{T}_{k,i}(T)$$

and $$L(i,k,t,T) < L(k,i,t,T) \quad \text{for} \quad t > \mathcal{T}_{k,i}(T)$$

LEMMA 3.4 $\mathcal{T}_{k,i}(T)$ is a non-decreasing function of T.

Proof: Consider $T_1 < T_2$ and let $y = T_2 - T_1$. From the definition of $\mathcal{T}_{k,i}(T)$, it is sufficient to show that if

$$L(k,i,t,T_1) \leq L(i,k,t,T_1), \text{ then}$$

$$L(k,i,t,T_2) \leq L(i,k,t,T_2)$$

and hence $\mathcal{T}_{k,i}(T_1) \leq \mathcal{T}_{k,i}(T_2)$

$$\begin{aligned} L(k,i,t,T_2) &= f(t_k + t - D_k) + f(t_i + t_k + t + T_2 - D_i) \\ &= L(k,i,t,T_1) + f(t_i + t_k + t + T_2 - D_i) \\ &\quad - f(t_i + t_k + t + T_1 - D_i) \end{aligned}$$

and

$$\begin{aligned} L(i,k,t,T_2) &= L(i,k,t,T_1) + f(t_i + t_k + t + T_2 - D_k) \\ &\quad - f(t_i + t_k + t + T_1 - D_k) \end{aligned}$$

Let
$$\begin{aligned} x_i &= t_i + t_k + t + T_1 - D_i \\ x_k &= t_i + t_k + t + T_1 - D_k \\ y &= T_2 - T_1 \end{aligned}$$

and
$$\begin{aligned} L(k,i,t,T_2) &= L(k,i,t,T_1) + \int_{x_i}^{x_i+y} f'(x)dx \\ L(i,k,t,T_2) &= L(i,k,t,T_1) + \int_{x_k}^{x_k+y} f'(x)dx \end{aligned}$$

But $x_i < x_k$ since $D_k < D_i$ and f' is non-decreasing,

hence $\int_{x_i}^{x_i+y} f'(x)dx \leq \int_{x_k}^{x_k+y} f'(x)dx$

Consequently, for $L(k,i,t,T_1) \leq L(i,k,t,T_1)$, $L(k,i,t,T_2) \leq L(i,k,t,T_2)$.

Corollary: If $f' = c$ (a constant), then $\mathcal{T}_{k,i}(T) = \mathcal{T}_{k,i}$ (a constant) for all $T \geq 0$.

Proof: Let $\mathcal{T}_{k,i} = \mathcal{T}_{k,i,}(0)$ and consider the proof to the above lemma. If $f' = c$, then

$$\int_{x_i}^{x_i+y} f' = \int_{x_k}^{x_k+y} f' .$$

But $\quad L(k,i, \mathcal{T}_{k,i},0) = L(i,k,\mathcal{T}_{k,i},0)$

Hence $\quad L(k,i, \mathcal{T}_{k,i},T) = L(i,k, \mathcal{T}_{k,i},T)$

for all $T \geq 0$ by the proof to the above lemma.

Since f is continuous,

$$L(i,k, \mathcal{T}_{k,i} + \epsilon,0) < L(k,i,\mathcal{T}_{k,i} + \epsilon,0)$$

for all $\epsilon > 0$ by definition of $\mathcal{T}_{k,i}$.

Consequently, $L(i,k, \mathcal{T}_{k,i} + \epsilon,T) < L(k,i, \mathcal{T}_{k,i} + \epsilon,T)$ for all $T \geq 0$ by the proof to Lemma 3.4.

Hence, by definition, $\mathcal{T}_{k,i}(T) = \mathcal{T}_{k,i}$ for all $T \geq 0$.

The results obtained in this chapter can now be used to determine an additional characteristic that some optimal schedules will have.

LEMMA 3.5 Given a schedule, $S = (J_1 \ldots J_n)$, where J_1 is scheduled at time t_o , having jobs J_i and J_k , $i < k$, which are single-time dependent with $\mathcal{T}_{i,k}(T)$.

If $\quad \mathcal{T}_{i,k}\left(\sum_{q=i+1}^{k-1} t_q\right) \leq t_o + \sum_{j=1}^{i-1} t_j$,

then the schedule, S' , defined by interchanging J_i and J_k has $P(S',t_o) \leq P(S,t_o)$.

Proof: For $\mathcal{T}_{i,k}(T)$ to exist, it is necessary that $t_k < t_i$ and $D_i < D_k$.

Let $\quad x_1 = \sum_{q=1}^{i-1} t_q$, $x_2 = \sum_{q=i+1}^{k-1} t_q$

$$P(S,t_o) = P((J_1 \ldots J_{i-1}), t_o) + P((J_{i+1} \ldots J_{k-1}), t_o + x_1 + t_i) + P((J_{k+1} \ldots J_n), x_1 + t_i + x_2 + t_k)$$

$$\begin{aligned} & + L(i,k,t_o + x_1, x_2) \\ & = P_1 + P_2 + P_3 + L_{i,k} \\ P(S',t_o) & = P_1 + P((J_{i+1} \ldots J_{k-1}), t_o + x_1 + t_k) + P_3 \\ & \quad + L(k,i,t_o + x_1, x_2) \\ & = P_1 + P_2' + P_3 + L_{k,i} \end{aligned}$$

But $P_2' \leq P_2$ since $t_k < t_i$

And $L_{k,i} < L_{i,k}$ since $\mathcal{T}_{i,k}(x_2) \leq t_o + x_1$

Hence, $P(S',t_o) \leq P(S,t_o)$

Corollary: Given a set of jobs, $J = \{J_1 \ldots J_n\}$ with adjacent time dependent equivalence classes $E_1 \ll E_2 \ll \ldots \ll E_p$. If J_i, $J_k \in E_m$ are single-time dependent with $\mathcal{T}_{i,k}(T)$ and

$$\mathcal{T}_{i,k} \Big(\sum_{\substack{q \ni J_q \in E_m \\ q \neq i,k}} t_q \Big) \leq \sum_{\substack{p \ni J_p \in E_i \\ i = 1 \ldots m-1}} t_p$$

then there is an optimal schedule in which J_k precedes J_i.

Proof: Consider any schedule S in which J_i precedes J_k. Since $\mathcal{T}_{i,k}(T)$ is non-decreasing the above lemma can be applied to obtain S' and $P(S',0) \leq P(S,0)$.

It is also reasonable to consider the consequences of

$$\mathcal{T}_{i,k}(0) > \sum_{\substack{q \ni J_q \in E_m \\ q \neq i,k}} t_q + \sum_{\substack{p \ni J_p \in E_i \\ i = 1 \ldots m-1}} t_p$$

in the scheduling problem defined in the above corollary. Intuitively, one would expect a "dual" to the above corollary to hold, namely for J_i to precede J_k in some optimal schedule. However, there is no such general result. While counter examples are difficult to construct, one is given below for quadratic loss functions.

Counter example for Quadratic Loss Functions

Let $f(x) = x^2$ and consider the problem of sequencing three jobs, J_i , J_{i+1}, J_k starting at $t = 0$.

For $t_{i+1} = 1/2 \quad t_k = 2 \quad t_i = 3$

and $D_i = 1\ 1/2 \quad D_k = 2 \quad D_{i+1} = 2.2$

$\mathcal{T}_{i,k}(0)$ can be found as follows:

$$L(i,k,t,0) = (t + 3/2)^2 + (t + 3)^2$$

$$L(k,i,t,0) = t^2 + (t + 7/2)^2$$

Equating and solving for t , $\mathcal{T}_{i,k}(0) = 1/2$

Hence, $\mathcal{T}_{i,k}(0) \geq t_{i+1}$. (It will be seen that the strict inequality can hold).

$$L(i,k,0,0) = 9/4 + 9 = 45/4 < L(k,i,0,0) = 49/4$$

Hence, $P((J_i,\ J_k,\ J_{i+1})\ 0) < P((J_k,\ J_i,\ J_{i+1}),0)$

In like manner,

$L(k,i+1,0,0) = 0 + .09 < L(i+1,k,0,0) = 0 + .25$

$$P((J_k,\ J_{i+1}\ ,\ J_i),0) < P((J_{i+1}\ ,\ J_k\ ,\ J_i),0)$$

$L(i+1,i,2,0) = .09 + 16 < L(i,i+1,2,0) = 49/4 + (3.3)^2$

$$P((J_k,\ J_{i+1},\ J_i),0) < P((J_k,\ J_i,\ J_{i+1}),0)$$

$L(i,i+1,0,0) = 9/4 + 1.69 = 3.94 < L(i+1,i,0,0) = 4$

$$P((J_i,\ J_{i+1},\ J_k),0) < P((J_{i+1},\ J_i,\ J_k),0)$$

Hence there are only three possible optimal schedules:

$$P((J_k,\ J_{i+1},\ J_i),0) = 0 + .09 + 16 = 16.09$$

$$P((J_i,\ J_{i+1},\ J_k),0) = 9/4 + 1.69 + 49/4 = 3.94 + 12.25$$

$$= 16.19$$

$$P((J_i,\ J_k,\ J_{i+1}),0) = 9/4 + 9 + (3.3)^2 = 22.14$$

and the schedule $(J_k\ ,\ J_{i+1}\ ,\ J_i)$ is optimal.

The general result does hold, however, for linear loss functions as will be demonstrated in the next chapter.

CHAPTER 4

SEQUENCING PROBLEMS WITH LINEAR LOSS FUNCTIONS

4.1 Introduction

The previous chapter dealt with problems having identical loss functions. For these problems an additional characteristic of an optimal schedule was obtained by deriving a restriction on the ordering of single-time dependent jobs. For problems where the loss function is of the form, $f(x) = ax + bx^2$, $x \geq 0$, all time dependent jobs are single-time dependent. It was shown, however, that the "dual" of the derived restriction did not hold for all problems in this class. In particular, a counter example was given for $f(x) = x^2$. This chapter considers problems having loss functions of the form, $f(x) = ax$, $x \geq 0$, and shows that the "dual" restriction holds for problems of this type. While these problems are the most structured of those considered so far, it will be shown that, even in this case, no easy solution technique is likely to arise.

4.2 Two Job Sequencing Rules

Without loss of generality, it will be assumed that $f(x) = x$, $x \geq 0$, (i.e. $a = 1$) . Based on the results of the previous chapter, the following observations can be made.

(1) For two jobs, J_i and J_k , $t_i \leq t_k$ and $D_i \leq D_k$, are necessary and sufficient conditions for $J_i << J_k$ or $J_i << (a)\ J_k$. This follows from the fact that $f'(x) = 1$ and consequently $f'(x)/f'(x+y) = 1$.

(2) As a consequence of the above, the reduction theorem holds for problems of this type.

(3) All time dependent jobs are single-time dependent since f is of the form $ax + bx^2$, $x \geq 0$.

(4) For any pair of time dependent jobs, $\mathcal{T}_{i,k}(T) = \mathcal{T}_{i,k}(0)$ for all $T \geq 0$ since $f'(x) = 1$.

Remark: Since the necessary and sufficient conditions for time independence and adjacent time independence are the same and since $\mathcal{T}_{i,k}(T) = \mathcal{T}_{i,k}(0)$ for all $T \geq 0$, it follows that these concepts are equivalent. The terms time independence or dependence actually used in the chapter are completely interchangeable with those of adjacent time independence or dependence.

In addition to (4) an even stronger statement can be made concerning $\mathcal{T}_{i,k}(0)$.

Notation: Let $\mathcal{T}_{i,k}$ denote $\mathcal{T}_{i,k}(0)$.

LEMMA 4.1 Consider two time dependent jobs, J_i and J_k , where $\mathcal{T}_{k,i}$ exists. Then $\mathcal{T}_{k,i} = D_i - t_k$ for D_i defined relative to $t = 0$.

Proof: Since $\mathcal{T}_{k,i}$ exists, it is necessary that $t_i < t_k$ and $D_k < D_i$. Now consider $t = D_i - t_k$, $T = 0$ and

$$\begin{aligned} L(k,i,D_i-t_k,0) &= f(t_k+t-D_k) + f(t_i+t_k+t-D_i) \\ &= D_i - D_k + t_i \end{aligned}$$

$$\begin{aligned} L(i,k,D_i-t_k,0) &= f(t_i+t-D_i) + f(t_i+t_k+t-D_k) \\ &= f(t_i-t_k) + D_i - D_k + t_i \\ &= 0 + D_i - D_k + t_i \end{aligned}$$

Hence, $L(k,i,D_i-t_k,0) = L(i,k,D_i-t_k,0)$

Consider $\epsilon > 0$ such that $D_i - t_k + \epsilon < D_i - t_i$

$$\begin{aligned} L(k,i,D_i-t_k+\epsilon,0) &= D_i - D_k + \epsilon + t_i + \epsilon \\ &= L(k,i,D_i-t_k,0) + 2\epsilon \\ L(i,k,D_i-t_k+\epsilon,0) &= L(i,k,D_i-t_k,0) + \epsilon \end{aligned}$$

Hence, $L(i,k,D_i-t_k+\epsilon,0) < L(k,i,D_i-t_k+\epsilon,0)$ for every such ϵ and consequently, $\mathcal{T}_{k,i} = D_i - t_k$ by definition.

Remark: Since $\mathcal{T}_{k,i}(T)$ is a constant, it follows that the relation $R(t,T)$, is independent of the value of T (i.e. $R(t,0) = R(t,T)$ for all $T \geq 0$) . For the remainder of this chapter, let $R(t)$ denote $R(t,0)$ and $J_i \ll (t)\ J_k$ denote $J_i \ll (t,0)\ J_k$.

LEMMA 4.2 The relation $R(t)$ is transitive.

Proof: It is necessary to show that if $J_i \ll (t)\ J_k$ and $J_k \ll (t)\ J_q$, then $J_i \ll (t)\ J_q$. The proof of this involves considering all possible combinations of reasons for $J_i \ll (t)\ J_k$ and $J_k \ll (t)\ J_q$. It is presented in detail in Appendix V.

Having these results, a final characteristic of an optimal schedule can be obtained.

4.3 Further Characterization Results

LEMMA 4.3 Given a schedule, $S = (J_1 \ldots J_n)$, where J_1 is scheduled at time t_o , having the following properties:

$$(1)\quad J_1 \ll (t_o)\ J_2 \ll (t_o+t_1)\ J_3 \ldots J_{n-1} \ll (t_o+\sum_{i=1}^{n-2} t_i)\ J_n$$

(2) J_1 and J_n are time dependent where

$$\mathcal{T}_{n,1} > t_2 + t_3 + \ldots + t_{n-1} + t_o \text{ .}$$

A new schedule, S' , can then be defined in which J_n precedes J_1 and $P(S',t_o) \leq P(S,t_o)$.

Proof: By induction on the number of jobs separating J_1 and J_n .

Initial: Consider the schedule $S = (J_1,J_2,J_3)$

where $J_1 \ll (t_o)\ J_2 \ll (t_o+t_1)\ J_3$ and $\mathcal{T}_{3,1} > t_2 + t_o$.

Since $\mathcal{T}_{3,1} > t_2 + t_o$, it follows that:

(1) $J_3 \ll (t_o)\ J_1$

(2) $t_1 < t_3$, $D_3 < D_1$, and $\mathcal{T}_{3,1} = D_1 - t_3 > t_2 + t_o$

or $D_1 > t_2 + t_3 + t_o$.

For $J_3 \ll (t_o)\ J_1$ and $J_1 \ll (t_o)\ J_2$, it follows that $J_3 \ll (t_o)\ J_2$ by the transitivity of $R(t)$.

But $J_2 \ll (t_o+t_1)\ J_3$ and hence J_2 and J_3 are time dependent with $t_o \leq \mathcal{T}_{3,2} < t_o + t_1$.

Hence

(3) $t_2 < t_3$, $D_3 < D_2$, and $t_o \leq \mathcal{T}_{3,2} < t_o + t_1$

or $t_o + t_3 \leq D_2 < t_o + t_1 + t_3$.

In this case, it will be shown that

$$P(\ (J_3,J_2,J_1),t_o) \leq P(\ (J_1,J_2,J_3),t_o)$$

or $S' = (J_3,J_2,J_1)$.

$$P(\ (J_1,J_2,J_3),t_o) = f(t_1+t_o-D_1) + f(t_1+t_2+t_o-D_2)$$
$$+ f(t_1+t_2+t_3+t_o-D_3)$$

$$P(\,(J_3,J_2,J_1),t_o) = f(t_3+t_o-D_3) + f(t_3+t_2+t_o-D_2)$$
$$+ f(t_1+t_2+t_3+t_o-D_1)$$

Since $D_1 > t_2 + t_3 + t_o > t_2 + t_1 + t_o$.

$P(\,(J_1,J_2,J_3),t_o) = f(t_1+t_2+t_o-D_2) + f(t_1+t_2+t_3+t_o-D_3)$.

Since $D_2 < t_o + t_1 + t_3$ and $D_1 \geq t_o + t_2 + t_3$, it follows that if $t_1 \leq t_2$ then

$$D_2 < t_o + t_1 + t_3 \leq t_o + t_2 + t_3 \leq D_1$$

or $D_2 < D_1$.

Further, if $t_2 < t_1$, then $D_1 < D_2$ since $J_1 \ll (t_o)\, J_2$ (i.e. $t_2 < t_1$, and $D_2 \leq D_1$ implies $J_2 \ll J_1$) . These two cases are considered separately.

(a) For $t_2 < t_1$ and $D_1 < D_2$,

$$t_o + t_1 + t_3 > D_2 > D_1 > t_2 + t_3 + t_o$$

and since $t_1 < t_3$

$$f(t_2+t_1+t_o-D_2) = f(t_2+t_3+t_o-D_2) = 0 \ .$$

Hence, $P(\,(J_1,J_2,J_3),\ t_o) = f(t_1+t_2+t_3+t_o-D_3)$

$$= t_1 + t_2 + t_3 + t_o - D_3$$

since $D_3 < D_2 < t_o + t_1 + t_3$

and $P(\,(J_3,J_2,J_1),t_o) = f(t_3+t_o-D_3) + f(t_1+t_2+t_3+t_o-D_1)$

$$= f(t_3+t_o-D_3) + t_1 + t_2 + t_3 + t_o - D_1$$

Clearly $P(\,(J_3,J_2,J_1),t_o) < P(\,(J_1,J_2,J_3),t_o)$

if $f(t_3 + t_o-D_3) = 0$ since $D_3 < D_1$.

Therefore, assume $f(t_3 + t_o -D_3) = t_3 + t_o - D_3 > 0$.

$P(\,(J_3,J_2,J_1),t_o) = t_3+t_o-D_3+t_1+t_2+t_3+t_o-D_1$

or $= t_3+t_o-D_1+ P(\,(J_1,J_2,J_3),t_o)$

But $D_1 > t_3 + t_o + t_2$

and hence

$$P((J_3,J_2,J_1),t_o) < P((J_1,J_2,J_3),t_o)$$

(b) For $t_1 \leq t_2$ and $D_2 < D_1$,

$$D_3 < D_2 \leq t_o+t_1+t_3 \leq t_o+t_2+t_3 \leq D_1$$

$$P((J_1,J_2,J_3),t_o) = f(t_1+t_2+t_o-D_2) + t_1+t_2+t_3+t_o-D_3$$

$$P((J_3,J_2,J_1),t_o) = f(t_3+t_o-D_3) + t_o+t_2+t_3-D_2 + f(t_1+t_2+t_3+t_o-D_1)$$

$$P((J_1,J_2,J_3),t_o) - P((J_3,J_2,J_1),t_o)$$
$$= [f(t_1+t_2+t_o-D_2) + t_1 - D_3] - [f(t_3+t_o-D_3) - D_2 + f(t_1+t_2+t_3+t_o-D_1)]$$

or $$= t_1 + (D_2-D_3) + f(t_1+t_2+t_o-D_2) - f(t_3+t_o-D_3) - f(t_1+t_2+t_3+t_o-D_1)$$

In the worst case, assume $f(t_1+t_2+t_o-D\ ) = 0$ and that the terms preceded by negative signs are positive.

Then

$$P((J_1,J_2,J_3),t_o) - P((J_3,J_2,J_1),t_o)$$
$$\geq t_1 + (D_2-D_3) + D_3-t_3-t_o+D_1-t_1-t_2-t_3-t_o$$
$$= (D_1-t_2-t_3-t_o) + (D_2-t_3-t_o) > 0$$

since $D_1 > t_o+t_2+t_3$ and $D_2 > t_o+t_3$.

Hence, $P((J_3,J_2,J_1),t_o) \leq P((J_1,J_2,J_3),t_o)$.

Induction: Consider the schedule, $S = (J_1 \ldots J_n)$ where

$$J_1 \ll (t_o)\ J_2 \ll (t_o+t_1)\ J_3 \ldots J_{n-1} \ll (t_o + \sum_{i=1}^{n-2} t_i)\ J_n$$

and $\tau_{n,1} > t_o + t_2 + \ldots + t_{n-1}$.

Following the reasoning used in the proof for the initial case,

$$t_1 < t_n,\ D_n < D_1, \text{ and } D_1 - t_n > t_o + t_2 + \ldots + t_{n-1}$$

or $D_1 > t_o + t_2 + \ldots + t_{n-1} + t_n$

and hence, $D_1 > t_o + t_1 + \ldots + t_{n-1}$.

It will now be shown that for any job $J_i \in \{J_2 \ldots J_{n-1}\}$ either J_1 or J_n may be interchanged with J_i resulting in a schedule S_1 where $P(S_1,t_o) \leq P(S,t_o)$ or $D_i \geq D_1$. In the former case, J_1 and J_n can be interchanged to obtain the schedule S' where $P(S',t_o) \leq P(S_1,t_o)$ by the induction hypothesis and hence $P(S',t_o) \leq P(S,t_o)$. If the latter case holds for all $J_i \in \{J_2 \ldots J_{n-1}\}$, then each of these jobs is early in the schedules, $S = (J_1 \ldots J_n)$ and $S' = (J_n J_2 \ldots J_{n-1} J_1)$ and

$$P(\ (J_1 \ldots J_n), t_o) = L(1,n,t_o, \sum_{i=2}^{n-1} t_i)$$

$$P(\ (J_n J_2 \ldots J_{n-1} J_1), t_o) = L(n,1,t_o, \sum_{i=2}^{n-1} t_i)$$

and $P(S',t_o) \leq P(S,t_o)$ since $\tau_{n,1} > t_o$.

Now for $J_i \in \{J_2 \ldots J_{n-1}\}$, consider the following possibilities:

(1) If $J_i \ll J_1$, then J_1 and J_i can be interchanged to form S_1 and the proof is complete.

(2) If $\tau_{1,i}$ exists and $\tau_{1,i} \leq t_o$, then J_1 and J_i can be interchanged to form S_1 .

(3) If $\tau_{i,1}$ exists, then $\tau_{i,1} = D_1 - t_i$.

$$\text{But } D_1 - t_i \geq t_o + \sum_{\substack{k=2 \\ k\neq i}}^{n} t_k \geq t_o + \sum_{k=2}^{i-1} t_k$$

and the induction hypothesis can be employed to interchange J_1 and J_i resulting in the schedule S_1.

The only remaining possibilities are:

(4) If $J_1 \ll J_i$, then $D_1 \leq D_i$

(5) If $\mathcal{T}_{1,i}$ exists, then $D_1 \leq D_i$ and $t_i < t_1$.

Hence, either J_1 and J_i are interchanged, or $D_i \geq D_1$.

Corollary: Given a set of jobs, $J = \{J_1 \ldots J_n\}$ with time dependent equivalence classes $E_1 \ll E_2 \ll \ldots \ll E_p$. If J_i, $J_k \in E_m$ are time dependent with $\mathcal{T}_{k,i}$ and

$$\mathcal{T}_{k,i} > \sum_{\substack{p \ni J_p \in E_i \\ i=1\ldots m-1}} t_p + \sum_{\substack{q \ni J_q \in E_m \\ q\neq i,k}} t_q$$

then there is an optimal schedule in which J_k precedes J_i.

Proof: Consider any schedule, S, in which J_i precedes J_k, and by the above lemma a schedule, S', can be defined in which J_k precedes J_i and $P(S') \leq P(S)$.

In summary, the following statements can now be made concerning an optimal schedule for a set of jobs, $J = \{J_1 \ldots J_n\}$ where $f_i = f_k = f$ is of the form $f(x) = ax$, $x \geq 0$:

(1) If J is decomposed into time dependent equivalence classes, $E_1 \ldots E_m$, where $E_1 \ll E_2 \ll \ldots \ll E_m$, there is an optimal schedule of the form $(E_1 \ldots E_m)$.

(2) Consider an arbitrary time dependent equivalence class E_i . If, in a schedule of the form given above, the jobs in the set E_i are to be scheduled in the time interval.

$$(t_o \ , \ t_o + \sum_{q \ni J_q \in E_i} t_q \ )$$

then there is an optimal schedule, $(J_{i_1} \ldots J_{i_p})$, for these jobs having the following properties.

(a) $J_{i_k} \ll (t_o + \sum_{r=1}^{k-1} t_{i_r}) \, J_{i_{k+1}}$

(b) If $J_{i_k} \ll J_{i_q}$, then J_{i_k} precedes J_{i_q} .

(c) If J_{i_k} and J_{i_q} are time dependent with $\mathcal{T}_{i_q, i_k} \leq t_o$, then J_{i_k} precedes J_{i_q} .

Further, J_{i_k} precedes J_{i_q} if

$$\mathcal{T}_{i_q, i_k} \leq t_o + \sum_{r=1}^{\min(q,k)-1} t_{i_r}$$

(d) If J_{i_k} and J_{i_q} are time dependent with

$$\mathcal{T}_{i_q, i_k} \geq t_o + \sum_{\substack{r=1 \\ r \neq k, q}}^{p} t_{i_q} \ ,$$

then J_{i_q} precedes J_{i_k} . Further, J_{i_q} precedes J_{i_k}

if $\mathcal{T}_{i_q, i_k} \geq t_o + \sum_{\substack{r=1 \\ r \neq k, q}}^{\max(k,q)} t_{i_r}$.

Unfortunately, as is shown in the next section, the conditions specified above do not always determine an optimal schedule. An optimal schedule can always be found by brute force (i.e. generate and evaluate all schedules satisfying the conditions specified), but this approach is not recommended. The best techniques should arise from the use of these restrictions in conjunction with dynamic programming methods such as those given by Held and Karp.(5)

4.4 Schedules for Three Jobs

In order to demonstrate that there are non-optimal schedules satisfying the conditions of the previous section, consider the problem of sequencing three jobs, J_1, J_2, and J_3, starting at $t = 0$. Without loss of generality, assume that $t_1 \leq t_2 \leq t_3$, and there are six general cases to consider which depend on the relative values of the due-dates.

(1) $D_1 \leq D_2 \leq D_3$

The schedule, (J_1, J_2, J_3), which satisfies the conditions of the previous section, is always optimal. All other schedules violate these conditions.

(2) $D_1 \leq D_3 < D_2$

$J_1 \ll J_2$ and $J_1 \ll J_3$, but J_2 and J_3 are time dependent. Hence, either (J_1, J_2, J_3) or (J_1, J_3, J_2) is optimal depending on the relative values of $L(2,3,t_1,0)$ and $L(3,2,t_1,0)$. The optimal schedule so selected satisfies the conditions specified while all other schedules will violate at least one of them.

(3) $D_2 < D_1 \leq D_3$

$J_2 \ll J_3$ and $J_1 \ll J_3$, but J_2 and J_1 are time dependent. This case is similar to (2) and a unique optimal schedule, consistent with the stated conditions, is determined by the ordering of J_1 and J_2 at $t = 0$.

(4) $D_2 < D_3 < D_1$

$J_2 \ll J_3$, but J_1, J_2 and J_1, J_3 are time dependent pairs with $\tau_{2,1} = D_1 - t_2 \geq D_1 - t_3 = \tau_{3,1}$. There are three schedules consistent with $J_2 \ll J_3$:

$S_1 = (J_1, J_2, J_3)$, $S_2 = (J_2, J_3, J_1)$, and $S_3 = (J_2, J_1, J_3)$

(a) If $\tau_{3,1} \leq 0$, then J_1 precedes J_3 and either S_1 or S_3 is optimal. This is determined by the ordering of J_1 and J_2 at $t = 0$ which produces the only schedule consistent with the conditions of the previous section.

(b) If $\tau_{3,1} > 0$, then $\tau_{2,1} > 0$ and $J_2 \ll (0)\ J_1$. This is similar to the above case except that S_2 or S_3 is selected as the optimal schedule on the basis of the ordering of J_1 and J_3 at $t = t_2$.

(5) $D_3 < D_1 \leq D_2$

$J_1 \ll J_2$, but J_1, J_3 and J_3, J_2 are time dependent pairs with $\tau_{3,1} = D_1 - t_3 \leq \tau_{3,2} = D_2 - t_3$. There are three schedules consistent with $J_1 \ll J_2$:

$S_1 = (J_1, J_2, J_3)$, $S_2 = (J_1, J_3, J_2)$, and $S_3 = (J_3, J_1, J_2)$

(a) If $\mathcal{T}_{3,1} \leq 0$, then J_1 precedes J_3 and either S_1 or S_2 is optimal depending on the ordering of J_2 and J_3 at $t = t_1$.

(b) If $\mathcal{T}_{3,2} \geq t_1$, then J_3 precedes J_2 and either S_2 or S_3 is optimal depending on the ordering of J_1 and J_3 at $t = 0$.

(c) If $\mathcal{T}_{3,1} > 0$ and $\mathcal{T}_{3,2} < t_1$, then $J_3 \ll (0)J_1$ and $J_2 \ll (t_1)J_3$ which means that only S_1 and S_3 need be considered. Further, S_1 and S_3 both satisfy all the conditions specified in the previous section.

$$\begin{aligned} P(S_1,0) &= f(t_1-D_1) + f(t_1+t_2-D_2) + f(t_1+t_2+t_3-D_3) \\ &= 0 + f(t_1+t_2-D_2) \ +t_1+t_2+t_3-D_3 \end{aligned}$$

and
$$\begin{aligned} P(S_3,0) &= f(t_3-D_3) + f(t_1+t_3-D_1) + f(t_1+t_2+t_3-D_2) \\ &= f(t_3-D_3) + t_1+t_3-D_1 + t_1+t_2+t_3-D_2 \end{aligned}$$

since $0 < D_1 - t_3 \leq D_2 - t_3 \leq t_1$

or $t_1 \leq t_2 \leq t_3 < D_1 \leq D_2 \leq t_1 + t_3 \leq t_2 + t_3$

and $D_3 < D_1$.

$$\begin{aligned} P(S_1,0) - P(S_3,0) &= f(t_1+t_2-D_2) - f(t_3-D_3) \\ &\quad + D_1 - D_3 + D_2 - t_1 - t_3 \end{aligned}$$

and there are four possible sets of values for this difference:

(1) $(D_2-t_1-t_3) < 0 + (D_1-D_3) > 0$

if $f(t_1+t_2-D_2) = f(t_3-D_3) = 0$

(2) $(D_2-t_1-t_3) < 0 + (D_1-t_3) > 0$

if $f(t_1+t_2-D_2) = 0, \ f(t_3-D_3) \neq 0$

(3) $(D_1-D_3) > 0 + (t_2-t_3) \leq 0$
if $f(t_1+t_2-D_2) \neq 0, \quad f(t_3-D_3) = 0$

(4) $(D_1-t_3) > 0 + (t_2-t_3) \leq 0$
if $f(t_1+t_2-D_2) \neq 0 \; , \; f(t_3-D_3) \neq 0$

In each case, the value of the difference can be positive or negative depending on the relative values of the variables involved. Computationally, then, there is no simple rule for selecting S_1 or S_3 as the optimal schedule. A second example of this type of situation is shown in the last case.

(6) $D_3 < D_2 < D_1$

All pairs of jobs are time dependent pairs with

$\tau_{3,2} = D_2 - t_3 < \tau_{3,1} = D_1 - t_3 < \tau_{2,1} = D_1 - t_2$.

Consider the schedule, (J_1,J_3,J_2) . If $J_1 \ll (0)J_3$, then $\tau_{3,2} < \tau_{3,1} \leq 0$ and $J_2 \ll (t_1)J_3$. Hence, (J_1,J_3,J_2) cannot be optimal. This leaves five possible optimal schedules.

$S_1 = (J_1,J_2,J_3)$; $S_2 = (J_2,J_1,J_3)$; S_3 (J_2,J_3,J_1)
$S_4 = (J_3,J_1,J_2)$ and $S_5 = (J_3,J_2,J_1)$.

(a) If $\tau_{3,1} < 0$, then $\tau_{3,2} < 0$ and J_1 and J_2 precede J_3 . Hence, either S_1 or S_2 is optimal depending on the ordering of J_1 and J_2 at $t = 0$

(b) If $\tau_{3,1} = D_1 - t_3 > t_2$, then $\tau_{2,1} = D_1 - t_2 > t_3$ and J_2 and J_3 precede J_1. Hence, either S_3 or S_5 is optimal depending on the ordering of J_2 and J_3 at $t = 0$.

(c) Thus, assume $0 < \tau_{3,1} = D_1-t_3 \leq D_1-t_2 = \tau_{2,1}$ and $\tau_{3,2} = D_2-t_3 < \tau_{3,1} = D_1-t_3 < t_2$.

1. If $\mathcal{T}_{3,2} < 0$, then J_2 precedes J_3 and it is only necessary to consider S_1 , S_2 , and S_3 . But for $\mathcal{T}_{2,1} > 0$, $J_2 \ll (0)J_1$ and for $\mathcal{T}_{3,1} < t_2$, $J_1 \ll (t_2)J_3$. Hence, S_2 is optimal.

2. If $\mathcal{T}_{3,2} > t_1$, then J_3 precedes J_2 and it is only necessary to consider S_4 and S_5 . But for $\mathcal{T}_{2,1} < t_3$, $J_1 \ll (t_3)J_2$ and S_4 is optimal.

3. Hence, assume $\mathcal{T}_{3,2} = D_2 - t_3 > 0$ and $\mathcal{T}_{3,2} < t_1$. Since $J_3 \ll (0)J_2$, $J_2 \ll (0)J_1$, and $J_1 \ll (t_3)J_2$, the only schedules that need be considered are

$$S_2 = (J_2,J_1,J_3) \quad \text{and} \quad S_4 = (J_3,J_1,J_2)$$

both of which satisfy the conditions of the previous section.

$$\begin{aligned} P(S_2,0) &= f(t_2-D_2) + f(t_2+t_1-D_1) + f(t_1+t_2+t_3-D_3) \\ &= 0 + f(t_2+t_1-D_1) + t_1+t_2+t_3-D_3 \end{aligned}$$

$$\begin{aligned} P(S_4,0) &= f(t_3-D_3) + f(t_3+t_1-D_1) + f(t_1+t_2+t_3-D_2) \\ &= f(t_3-D_3) + f(t_3+t_1-D_1) + t_1+t_2+t_3-D_2 \end{aligned}$$

$$\begin{aligned} P(S_4,0) - P(S_2,0) = f(t_3-D_3) &+ f(t_3+t_1-D_1) \\ &- f(t_2+t_1-D_1) + (D_3-D_2) \end{aligned}$$

and there are six possible sets of values for the difference:

(1) $(D_3-D_2) < 0$ if $f(t_3-D_3) = f(t_3+t_1-D_1) = f(t_2+t_1-D_1) = 0$
and $P(S_4,0)$ is optimal.

(2) $(t_3-D_3) + (D_3-D_2) = t_3-D_2 > 0$ if $f(t_3-D_3) \neq 0$
$f(t_3+t_1-D_1) = f(t_2+t_1-D_1) = 0$ and $P(S_2,0)$ is optimal.

(3) $(t_1+t_2-D_1) > 0 + (D_3-D_2) < 0$ if $f(t_2+t_1-D_1) \neq 0$
$f(t_3-D_3) = f(t_3+t_1-D_1) = 0$.

(4) $(t_1+t_2-D_1) > 0 + (t_3-D_2) < 0$ if $f(t_3+t_1-D_1) = 0$
$f(t_3-D_3)$, $f(t_2+t_1-D_1) \neq 0$

(5) $(t_3-t_2) \geq 0 + (D_3-D_2) < 0$ if $f(t_3-D_3) = 0$
$f(t_3+t_1-D_1)$, $f(t_2+t_1-D_1) \neq 0$

(6) $(t_3-t_2) \geq 0 + (t_3-D_2) < 0$ if $f(t_3-D_3)$,
$f(t_3+t_1-D_1)$, $f(t_2+t_1-D_1) \neq 0$

In the last four cases, the value of the difference can be either positive or negative depending on the relative values of the variables involved.

The problem of sequencing just three jobs, then, provides several examples where a non-optimal schedule can satisfy the conditions of the previous section. Further, there appears to be no straight-forward way of selecting the optimal schedule other than by evaluating each schedule satisfying those conditions.

CHAPTER 5

AN n JOB, ONE MACHINE SEQUENCING ALGORITHM FOR MINIMIZING THE NUMBER OF LATE JOBS

5.1 Introduction

The problems considered in this paper belong to a general class of problems in which a finite set of jobs must be sequenced through a single facility, minimizing some function of the lateness penalties incurred. Reference 5 contains one algorithm for solving these problems, a dynamic programming technique formulated by Held and Karp. While this method is applicable to the problem defined here, it is generally computationally infeasible for problems of 20 jobs or more. The algorithm developed here, however, consists of only two sorting operations performed on the total set of jobs, and a maximum of $n(n+1)/2$ additions and comparisons. Consequently, this method will be computationally feasible for very large problems and can be performed manually on many smaller problems.

The problem is to sequence n jobs, $J_1 \dots J_n$, with known processing times, $t_1 \dots t_n$, and due dates $D_1 \dots D_n$, through a single production facility in such a way as to minimize the number of late jobs. The processing times, which are defined to include set-up and tear-down times, are independent of the sequence. It is assumed that the n jobs are available for production throughout the scheduling period and that the production facility operates continuously until all jobs are completed. No lot-splitting is allowed, so that production on a job continues from start to finish without interruption.

It is assumed that each job can be completed by its due-date defined relative to $t = 0$, if the processing of that job were to begin immediately (i.e., $\forall i$, $t_i \leq D_i$). If this were not the case, the problem size would be reduced by eliminating those jobs that cannot be completed on time regardless of the sequence.

5.2 The Algorithm

The algorithm consists of a decision rule for dividing the set of jobs into two subsets. This division produces an optimal schedule if the jobs in the first set are sequenced according to their due-dates, and are followed by the jobs in the second set in any order.

Step 1: Order the set of jobs according to the shortest processing time rule and call the resulting ordering $((J_{i_1} \ldots J_{i_n})$ where $t_{i_1} \leq \ldots \leq t_{i_n})$ the current sequence.

Step 2: Using the current sequence, find the first late job, J_{i_q}, and go to Step 3. If no such job is found, the algorithm terminates with an optimal schedule obtained by ordering the jobs in the current sequence according to their due-dates followed by the set of jobs that have been rejected in any order.

Step 3: Reorder the jobs, $J_{i_1} \ldots J_{i_q}$, according to the due-date rule producing a sequence of the form:

$$J_{\ell_1} \ldots J_{\ell_q}, J_{i_{q+1}} \ldots J_{i_n} \quad \text{where} \quad D_{\ell_1} \leq \ldots \leq D_{\ell_q} .$$

There are two cases to consider:

(1) If all the jobs in the subsequence $J_{\ell_1} \dots J_{\ell_q}$ are early, define the total sequence as the current sequence and go to Step 2.

(2) Otherwise reject the job J_q and remove it from the sequence $J_{\ell_1} \dots J_{\ell_q} J_{i_{q+1}} \dots J_{i_n}$. Now go to Step 2 with the resulting sequence as the current sequence.

Example

The process is illustrated in the following numerical example:

Job	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
Processing Time	10	6	3	1	4	8	7	6
Due-Date	35	20	11	8	6	25	28	9

Step 1: Ordering the jobs according to the shortest processing time rule results in the following current sequence:

Job	J_4	J_3	J_5	J_2	J_8	J_7	J_6	J_1
Processing Time	1	3	4	6	6	7	8	10
Due-Date	8	11	6	20	9	28	25	35
Completion Time	1	4	8	--	--	--	--	--

Step 2: The result of computing the first 3 completion times indicates that J_5 is the first late job. Therefore, Step 3 is performed.

Step 3: The result of re-ordering the first three jobs according to the due-date rule is shown below.

Job	J_5	J_4	J_3	J_2	J_8	J_7	J_6	J_1
Processing Time	4	1	3	6	6	7	8	10
Due-Date	6	8	11	20	9	28	25	35
Completion Time	4	5	8	14	20	--	--	--

As indicated above, the first three jobs are now early. Hence, Step 2 is re-performed using the above sequence

Step 2: The first late job is now J_8 as in shown in the table above. Consequently, Step 3 is performed.

Step 3: Ordering the first 5 jobs according to the due-date rule gives the sequence below:

Job	J_5	J_4	J_8	J_3	J_2	J_7	J_6	J_1
Processing Time	4	1	6	3	6	7	8	10
Due-Date	6	8	9	11	20	28	25	35
Completion Time	4	5	11	14	20	--	--	--

Since J_8 and J_3 are both late, J_8 is rejected and the new current sequence is shown below:

	Current Sequence							Rejected Jobs
Job	J_5	J_4	J_3	J_2	J_7	J_6	J_1	J_8
Processing Time	4	1	3	6	7	8	10	6
Due-Date	6	8	11	20	28	25	35	9
Completion Time	4	5	8	14	21	29	--	--

Step 2: The first late job is now J_6 .

Step 3: Re-ordering the first six jobs according to due-dates gives the following sequence:

	Current Sequence							Rejected Jobs
Jobs	J_5	J_4	J_3	J_2	J_6	J_7	J_1	J_8
Processing Time	4	1	3	6	8	7	10	6
Due-Date	6	8	11	20	25	28	35	9
Completion Time	4	5	8	14	22	29	--	--

Since J_7 is late, J_6 is rejected, resulting in the following new current sequence.

	Current Sequence							Rejected Jobs
Jobs	J_5	J_4	J_3	J_2	J_7	J_1	J_8	J_6
Processing Time	4	1	3	6	7	10	6	8
Due-Date	6	8	11	20	28	35	9	25
Completion Time	4	5	8	14	21	31	--	--

Step 2: All jobs in the current sequence are now early and the algorithm terminates with the above sequence $(J_5,J_4,J_3,J_2,J_7,J_1,J_8,J_6)$ as an optimal schedule.

On the final step of the algorithm it is not necessary to re-order the current sequence according to the due-date rule. Doing so, however, will still produce an optimal schedule. The proof for the optimality of this algorithm follows.

5.3 Theoretical Development

Definition: A schedule, S , is defined as a specific ordering, $(J_{i_1} \ldots J_{i_n})$, of the set of jobs to be processed.

For each schedule, S , two sets, E and L , are defined as follows:

$$\begin{array}{lll} Ji \in E & \text{iff} & C_i \leq D_i \\ Ji \in L & \text{iff} & C_i > D_i \end{array}$$

where C_i is the completion time of job J_i in the schedule S .

Definition: For a given schedule, S , let A denote the ordered set defined by ordering the elements of the set E according to their order in the schedule S .

Similarly, let R denote the ordered set defined by ordering the elements of the set L according to their order in the schedule S .

Finally, let π denote an arbitrary permutation of the set R and P denote the resulting ordered set.

LEMMA 5.1 Given an optimal schedule, S , for a set of jobs, $J = \{J_1 \ldots J_n\}$, with known processing times, $t_1 \ldots t_n$, and due-dates, $D_1 \ldots D_n$, the schedule,

$$S^* = (A,R), \text{ is optimal as is any schedule of the form,}$$
$$S^{**} = (A,P).$$

Proof: Suppose $S = (J_{i_1} \ldots J_{i_n})$ is not of the form (A,R) . Then there must exist a pair of adjacent jobs, J_{i_k} and $J_{i_{k+1}}$, in S such that

$$C_{i_{k+1}} \leq D_{i_{k+1}}$$

and

$$C_{i_k} > D_{i_k}$$

Hence, define a new optimal schedule, S_1, by interchanging J_{i_k} and $J_{i_{k+1}}$ and S_1 will have $A_1 = A$ and $R_1 = R$. If S_1 is of the form (A,R), we are done. Otherwise, operate on S_1 to obtain S_2, etc. This process terminates in a finite number of steps with an optimal schedule S_q where $A_q = A$, $R_q = R$, and $S_q = (A,R) = S^*$.

LEMMA 5.2 (Smith) Given an optimal schedule, S, of the form, (A,R), a new optimal schedule, $S_D = (A_D,R)$, may be defined by re-ordering the set A according to the due-date rule.

Proof: Suppose $A = (J_{i_1} \dots J_{i_p})$ is not ordered according to the due-date rule. Then there must exist a pair of adjacent jobs, J_{i_k} and $J_{i_{k+1}}$, in A such that

$$D_{i_{k+1}} < D_{i_k}$$

But

$$\sum_{j=1}^{k+1} t_{i_j} \leq D_{i_{k+1}} < D_{i_k}$$

Consequently, the jobs J_{i_k} and $J_{i_{k+1}}$ may be interchanged to obtain a new optimal schedule $S_1 = (A_1,R)$. If $A_1 = A_D$, we are done. Otherwise, operate on S_1 to obtain S_2, etc. This process terminates in a finite number of steps wihh a schedule S_q where $A_q = A_D$ and hence $S_q = S_D$.

Corollary (Smith): Given a set of jobs, $J = \{J_1 \ldots J_n\}$, with known processing times, $t_1 \ldots t_n$, and due-dates, $D_1 \ldots D_n$, there exists a schedule, S , having no late jobs if and only if there are no late jobs in the schedule, S_d , defined by ordering the set of jobs according to their due-dates.

Proof:

"If" Obvious

"Only If" If there exists a schedule, S , having no late jobs, then $L = \emptyset$ (the empty set) and consequently $S = A$. Now apply lemma 5.2 to obtain $S_D = A_D = S_d$ which is still optimal.

These lemmas show that there is associated with any optimal schedule, S , an optimal schedule $S_D = (A_D,P)$. Consequently, the original problem can be restated as one of producing a schedule of this form where the cardinality of the associated set, E_D , is maximal.

LEMMA 5.3 Suppose there exists an optimal schedule, S , for a set of jobs, J , where L is not empty. Let $J*$ denote any subset of L . Then for any optimal schedule, $S' = (A',R')$, for the set of jobs $J' = J-J*$, an optimal schedule, $S'' = (A'',P'')$, for the complete set of jobs, J , can be defined by letting $A'' = A'$ and $L'' = J*uL'$.

Proof: Without loss of generality, assume that S is of the form (A,R) and consider any optimal schedule, $S' = (A',R')$, for the set of jobs J' .

If the cardinality of the set E' , denoted $/E'/$, equals the cardinality of the set E , denoted $/E/$, we are done. Therefore assume $/E'/ \neq /E/$.

(a) If $/E'/ < /E/$, we can define a new optimal schedule, $S_1 = (A_1,P_1)$, for the set of jobs, J' , by letting $A_1 = A$ and $L_1 = L - J*$. But $/E_1/ = /E/ > /E'/$. Hence, S' is not an optimal schedule for the set of jobs J' --a contradiction. Therefore, $/E'/ \geq /E/$.

(b) If $/E'/ > /E/$, we can define a new optimal schedule, $S_2 = (A_2,P_2)$, for the set of jobs, J , by letting $A_2 = A'$ and $L_2 = L' \cup J^*$. But $/E_2/ = /E'/ > /E/$ and hence S is not an optimal schedule for the set of jobs, J --a contradiction. Consequently, $/E'/ = /E/$.

The problem is now one of developing an algorithm which will find a job, $J_i \in J$, such that $J_i \in L$ for some optimal schedule when not all of the jobs in the original set can be completed on time (i.e., the schedule S_d has at least one late job). An optimal schedule for the original set of jobs can then be obtained by repeatedly applying the algorithm and eliminating the jobs found. The process will terminate at some point where the set of jobs, $J* = (J_{i_1} \ldots J_{i_q})$ has been elimanated, but the algorithm fails to find a job in the set $J - J*$. An optimal schedule for the original set of jobs, J , is now defined by letting:

$$E = J - J*$$

$$L = J*$$

and hence $$S = (A_D, P)$$

where A_D is the ordered set obtained by ordering the set E according to the due-date rule. One can show that this schedule is optimal by repeatedly applying lemma 5.3. (i.e., Clearly, $S_q = A_D$ is an optimal schedule for the set of jobs $J - J^*$. Then, using lemma 5.3, $S_{q-1} = (A_D, J_{i_q})$ is an optimal schedule for the set of jobs $\{J-J^*\} \cup \{J_{i_q}\}$, etc).

Selection Algorithm:

Given a set of jobs, $J = \{J_1 \dots J_n\}$, with known processing times, $t_1 \dots t_n$, and due-dates, $D_1 \dots D_n$, the following algorithm will find a job $J_i \in J$ and $J_i \in L$ for some optimal schedule if such a job exists. Define the initial current sequence as $(J_1 \dots J_n)$ assuming that $t_1 \leq t_2 \leq \dots \leq t_n$.

Start: Find the first late job, J_q, in the current sequence where $q > 1$. (If no such job exists, the current sequence has no late jobs and the algorithm terminates.). Re-order the previous q-1 jobs according to the due-date rule to obtain a sequence of the form:

$$(J_{i_1} \dots J_{i_{q-1}}, J_q \dots J_n)$$

where

$$t_q \leq \dots \leq t_n$$

$$D_{i_1} \leq \dots \leq D_{i_{q-1}}$$

and $$t_q \geq t_{i_j} \quad \text{for} \quad j = 1 \dots q-1$$

Insert J_q into the sequence $(J_{i_1} \dots J_{i_{q-1}})$ according to the due-date rule, obtaining the sequence:

$$(J_{i_1} \dots J_{i_k}, J_q, J_{i_{k+1}} \dots J_{i_{q-1}}, J_{q+1} \dots J_n)$$

where

$$D_{i_1} \leq \dots \leq D_{i_k} \leq D_q \leq D_{i_{k+1}} \leq \dots \leq D_{i_{q-1}}$$

There are three cases to consider:

(1) If all jobs in the sub-sequence $(J_{i_1} \ldots J_{i_k} , J_q , J_{i_{k+1}} \ldots J_{i_{q-1}})$ are early, resume the analysis at start using the sequence just defined as the new current sequence.

(2) If J_q is late in the above sequence, then $J_q \in L$ for some optimal schedule for the jobs in the current sequence.

Proof: Consider an optimal schedule of the form, $S_D = (A_D,P)$ for the set of jobs in the current sequence. If $J_q \in P$, we are done. Hence, assume $J_q \in A_D$ and that S_D is of the form:

$$S_D = (J_{\ell_1} \ldots J_{\ell_r} \ldots J_{\ell_t} , P)$$

and

$$D_{\ell_1} \leq \cdots \leq D_{\ell_r} < D_{\ell_{r+1}} \leq \cdots \leq D_{\ell_t}$$

where

$$J_{\ell_r} = J_q$$

In the current sequence $t_q \leq \ldots \leq t_n$ and $t_{i_j} \leq t_q$ for $j = 1 \ldots q-1$. Thus, the summation of the processing times for the jobs in the set $\{J_{i_1} \ldots J_{i_k}\}$ is minimal over the class of all subsets of k jobs from the current sequence having due-dates less than or equal to D_q. Hence, since J_q is late in the sequence $(J_{i_1} \ldots J_{i_k} , J_q)$, it will be late in all due-date ordered sequences in which it is the $k + 1$st job. But $J_{\ell_r} = J_q$ is early in the schedule S_D and therefore r must be strictly less than $k + 1$. Further

$$\{J_{\ell_{r+1}} \ldots J_{\ell_t}\} \cap \{J_{i_1} \ldots J_{i_k}\} = \emptyset$$

Since $D_{i_j} \leq D_q$ for $j = 1 \ldots k$ and $D_q < D_{\ell_m}$ for $m = r + 1 \ldots t$. Thus, there are $(k - (r-1)) \geq 1$ jobs in the set of jobs $\{J_{i_1} \ldots J_{i_k}\}$ that are members of L for the schedule S_D. A new optimal schedule, S_D^*, can now be defined. Order the set of jobs, $\{J_{i_1} \ldots J_{i_k}\}$, according to the shortest processing time rule and select the first r jobs from this set. Replace the initial sequence, $(J_{\ell_1} \ldots J_{\ell_r})$, in the schedule S_D by this set of jobs ordered according to the due-date rule and call this schedule S_D^*. The job $J_{\ell_r} = J_q$ is now a member of L^* for the optimal schedule S_D^*.

(3) If J_q is not late in the above sequence but at least one of the jobs in the sequence $(J_{i_{k+1}} \ldots J_{i_{q-1}})$ is late, then $J_q \in L$ for some optimal schedule for the set of jobs in the current sequence.

Proof: As in the proof for 2), assume we have an optimal schedule, S_D, of the form:

$$S_D = (J_{\ell_1} \ldots J_{\ell_r} \ldots J_{\ell_t}, P)$$

and

$$D_{\ell_1} \leq \ldots \leq D_{\ell_r} < D_{\ell r+1} \leq \ldots \leq D_{\ell_t}$$

where

$$J_{\ell_r} = J_q$$

(a) If $r < k + 1$, the same optimal schedule, S_D^*, can be defined as in (2) where $J_q \in L^*$.

(b) If $r \geq k + 1$, the argument developed in the proof for (2) can be used to show that the completion time of $J_{\ell_r} = J_q$ in the schedule S_D is greater than or equal to its completion time in the sequence $(J_{i_1} \ldots J_{i_k}, J_q)$. Therefore, since at least one of the jobs in the set $\{J_{i_{k+1}} \ldots J_{i_{q-1}}\}$ was late in the original sequence, at least one of these jobs, say J_{i_m}, must belong to the set L for the schedule S_D.

But

$$t_{i_m} \leq t_q$$

and

$$D_{i_m} \geq D_q$$

Hence, J_q can be removed from A_D and replaced by J_{i_m} to form a new optimal schedule S_D^* where $J_q \in L^*$.

This process continues and each succeeding job is either accepted into the initial due-date ordered sub-sequence or is rejected. The algorithm terminates with a due-date ordered sequence of early jobs, $(J_{a_1} \cdots J_{a_t})$, and a set of jobs, $J_{a_{t+1}} \cdots J_{a_n}$ that have been rejected. An optimal schedule to the original problem will be

$$S_D = (J_{a_1} \cdots J_{a_t} , P)$$

where P is defined for the set of jobs, $\{J_{a_{t+1}} \cdots J_{a_n}\}$.

5.4 Minimizing the Maximum Deferral Cost*

Retaining the assumptions made in the introduction, a new problem can be formulated. Associated with each job is a continuous, bounded, monotonically non-decreasing function, P_i , called a deferral cost where $P_i(t)$ represents the cost of completing the job at time t. Sequencing problems with these cost structures have been considered by McNaughton(9) and Lawler.(7) Their results, however, concern the problem of minimizing the sum of the deferral costs. The objective here is to find a schedule for which the maximum deferral cost incurred is minimal.

*This extension was suggested by Professor E. L. Lawler, Department of Electrical Engineering, The University of Michigan.

Definition: For each P_i, define a function P_i^* as follows:

(1) $P_i^* (y) = P_i^{-1}(y)$ if $P_i^{-1}(y)$ (the inverse) exists

(2) Otherwise

(a) $P_i^*(y) = \max \{t | P_i(t) = y\}$ if $\exists t \ni P_i(t) = y$

(b) $P_i^*(y) = 0$ if $\forall t, P_i(t) > y$

(c) $P_i^*(y) = + \infty$ if $\forall t, P_i(t) < y$

The corollary to lemma 5.2 can now be used to find an optimal schedule. For arbitrary $y > 0$, let $D_i = P_i^*(y)$ and let $S_D(y)$ be the schedule obtained by ordering the jobs according to the "due-dates" so defined. In that the P_i's are bounded, there always exists $y > 0$ such that $S_D(y)$ has no "late" jobs.* Since the P_i's are monotonically non-decreasing, the P_i^*'s are monotonically non-decreasing and hence there exists $y^* > 0$ such that:

(1) $S_D(y^*)$ has no "late" jobs.

(2) For all $y < y^*$, $S_D(y)$ has at least one late job.

This value can be found to whatever accuracy is desired by a binary search technique. The schedule $S_D(y^*)$ has minimal maximum deferral cost.

*Actually the bounded condition is stronger than necessary. It is sufficient if there exists $y > 0$ such that $S_D(y)$ has no "late" jobs.

CHAPTER 6

DYNAMIC PROGRAMMING ALGORITHMS FOR THE SEQUENCING OF JOBS SUBJECT TO DEADLINES*

6.1 Introduction

The previous chapter considered the problem of finding a sequence for a finite set of jobs that minimized the number of late jobs. It was shown that there is an optimal schedule in which the early jobs precede the late jobs and the former are ordered according to their due-dates. For this problem and others similar to it, a certain "consistency principle" holds. This principle can be stated as follows:

> Given a set of jobs which are to be processed subject to deadlines, there exists a well ordering of the complete set of jobs, such that for any subset of these jobs, an optimal sequence exists in which the ordering of the jobs completed on time is consistent with the well ordering.

Problems of this type can always be solved using the Held and Karp(5) algorithm, but this chapter demonstrates how this consistency principle can be exploited to obtain more efficient solution methods. Three types of problems considered are (1) one machine and jobs with individual deadlines and priorities (previous chapter); (2) two machines in series and jobs with priorities and a common deadline (Johnson(6)); and (3) a single machine and jobs with priorities,

*The dynamic programming formulations of the problems in this chapter were suggested by Professor E. L. Lawler, Department of Electrical Engineering, University of Michigan.

a common deadline, and individual linear deferral costs (Mc Naughton(9) and Smith(15)). These problems are dealt with in sections 2, 3, and 4 respectively. A final section discusses multi-machine generalizations.

6.2 One Machine, Multiple Deadlines and Priorities

Consider a set of jobs, $J = \{J_1 \ldots J_n\}$, where t_i and D_i represent the processing time and due-date for job i. In addition, let $r_i > 0$ denote a reward that is earned if job J_i is completed on time. The problem is to find a sequence for these jobs which maximizes the sum of the earned rewards.

The previous chapter dealt with the special case of this problem where $r_1 = r_2 = \ldots = r_n$ and the first two lemmas developed there also apply here. Hence, there is an optimal schedule in which the early jobs precede the late jobs and the former are ordered according to their due-dates. The problem, then, is that of selecting the jobs which are to be completed on time. A dynamic programming formulation of this problem is given below where it is assumed, without loss of generality, that $D_1 \leq D_2 \leq \ldots \leq D_n$.

Definition: Let $f_i(t)$ = the maximum attainable total reward for a selection of jobs from the set of jobs $\{J_1 \ldots J_i\}$, subject to the constraint that no job is completed later than time t .

A recursion relation for $f_i(t)$ can be formulated as follows:

(1) Consider $t \leq D_i$. If there exists an optimal schedule in which J_i is completed on time, then J_i can be the last job and $f_i(t) = f_{i-1}(t-t_i) + r_i$. If no such schedule exists, then $f_i(t) = f_{i-1}(t)$.

(2) For $t \geq D_i \geq \ldots \geq D_1$, $f_i(t) = f_i(D_i)$ and can be computed in the manner indicated above.

Hence, for $i = 1, \ldots, n$:

$$f_i(t) = f_i(D_i) \text{ for } t > D_i$$

$$f_i(t) = \max \{f_{i-1}(t),\ r_i + f_{i-1}(t-t_i)\} \text{ for } 0 < t \leq D_i$$

subject to the boundary conditions

$$f_i(0) = 0$$

$$f_0(t) = 0$$

$$f_i(t) = -\infty \quad \text{for } t < 0$$

The maximum attainable total reward for the complete set of jobs is given by $f_n(D_n)$.

Assuming that all of the processing times and deadlines are integers, the length of the computation grows no more rapidly than $n(D_n)$, i.e., proportional to the product of the number of jobs and the longest deadline.

6.3 Two Machines in Series, Common Deadline

The problem just discussed can be viewed as that of a single machine, individual starting times, and a common deadline. We now deal with the problem of two machines in series, a common starting time, and a common deadline where each job must be processed by both machines, in sequence.

Consider a set of jobs, $J = \{J_1 \ldots J_n\}$ where for each job J_i , a_i denotes its processing time on the first of two machines, and b_i its processing time on the second. Let r_i denote a reward which is earned if job J_i is completed on the second machine by a deadline T (common to all jobs). The problem is to find a sequence for the set of jobs which maximizes the sum of the earned rewards.

It follows from the results of the previous chapter and those of Johnson[(6)] that there exists such a maximal sequence in which (1) the jobs completed on time precede the tardy jobs; (2) the jobs are processed in the same order by both machines; and (3) the on-time jobs are ordered according to the following relation:

Job J_p precedes J_q only if $\min\{a_p, b_q\} \leq \min\{a_q, b_p\}$

Once again, the problem consists of making a selection of the jobs which are to be completed on time. Given such a selection, the ordering of these jobs is determined by Johnson's relation;[(6)] the ordering of the remaining jobs which follow is arbitrary.

Without loss of generality, assume that

$$\min\{a_j, b_{j+1}\} \leq \min\{a_{j+1}, b_j\}\ , \text{ for } j = 1 \ldots n-1\ .$$

Let $f_i(t_1, t_2)$ = the maximum attainable reward for a selection of jobs from among $J_1 \ldots J_i$, subject to the constraint that the completion time of no job is later than time t_1 on the first machine or t_2 on the second.

Following the type of argument used in the previous section, a recursion relation for $f_i(t_1, t_2)$ where t_1 , $t_2 \leq$ T can be formulated as follows.

For $i = 0, 1, 2, \ldots, n$:

$$f_i(t_1, t_2) = \max\{f_{i-1}(t_1, t_2),\ r_i + f_{i-1}(\min\{t_1 - a_i,\ t_2 - a_i - b_i\}\ ,\ t_2 - b_i\})\}$$

Subject to the boundary conditions:

$$f_0(t_1,t_2) = 0$$
$$f_i(0,0) = 0$$
$$f_i(t_1,t_2) = -\infty \qquad (t_1 < 0 \text{ or } t_2 < 0)$$

The maximum attainable total reward for the complete set of jobs is, of course, given by $f_n(T,T)$. Assuming all processing times are integers, the length of the computation grows no more rapidly than $n(T^2)$.

6.4 Single Machine, Common Deadline, Linear Deferral Costs

As a final example of the principle of consistency, consider the problem of a single machine, a common deadline, and linear deferral costs.

Consider a set of jobs $J = \{J_1 \ldots J_n\}$, where for each job J_i , a_i denotes its processing time and r_i a reward which is earned if the job is completed by a deadline T (common to all jobs). In addition, let p_i denote a linear deferral cost coefficient. Thus, if the job J_i is completed at a time $t \leq T$, the net profit earned for that job (reward minus deferral cost) is $r_i - p_i\, t$.

As a possible example of such a problem, consider the position of a contractor who is free to accept or reject jobs. For each job J_i which is accepted and completed prior to the deadline, a reward r_i is earned. However, the deferral of the job causes a cost to be incurred which is determined by the coefficient p_i . What selection of jobs maximizes profit?

Given any selection of jobs, such that the sum of their processing times is no greater than T , the jobs should be ordered according to the ratios p_i/a_i, the job with the largest ratio being processed first. This result has been found by McNaughton[(9)] and Smith.[(15)]

Without loss of generality, assume $(p_i/a_i \leq p_{i+1}/a_{i+1})$. Let $f_i(t_1,t_2)$ = the maximum attainable net profit for a selection of jobs from among $J_1, J_2 \ldots J_i$, subject to the constraint that the starting time of the first job is t_1 and the last job is to be completed before t_2.

An appropriate set of recursion equations is as follows for $t_1, t_2 \geq 0$ and $t_2 \leq T$.

$$f_i(t_1,t_2) = \max \{f_{i-1}(t_1,t_2),\ r_i - p_i(t_1+a_i) + f_{i-1}(t_1+a_i,\ t_2)\}$$

subject to the intial conditions

$$f_i(0,0) = 0$$

$$f_0(t_1,t_2) = 0$$

$$f_i(t_1,t_2) = -\infty \qquad \text{for } t_1 > t_2 .$$

The maximum attainable total profit for the complete set of jobs is given by $f_n(0,T)$. Assuming all processing times are integers, the length of the computation grows as $n(T)$.

There is an interesting variation of this problem in which the deadline is not controlling. E.g., T is as large as the sum of all the processing times. In this case, the selection of jobs is controlled solely by the question of whether or not the jobs can be completed before their deferral costs exceed their rewards.

6.5 Machines in Parallel

Each of the problem formulations and solution methods given above can be extended in a very natural way to the situation in which there are many machines, or sets of machines, in parallel, and any given job can be processed by any given machine. In each such extension, the jobs that are assigned to any given machine are processed in an order which is consistent with the ordering obtained by solving the associated single machine problem.

Consider the extension of the problem of multiple deadlines (section 6.2 above). Let there be given a set of jobs, $J = \{J_1 \ldots J_n\}$. For each job J_i , let $a_{i,k}$ denote its processing time on the kth of m machines and $r_{i,k}$ a reward which is earned if processing of the job is performed on machine k and completed prior to the deadline for the job, D_i .

As before, we assume, without loss of generality, that $D_i \leq D_{i+1}$, for $i = 1, 2, \ldots n$. (Note that it is feasible to have different deadlines on different machines. However, it must be the case that $D_{i,k} \leq D_{i+1,k}$ for all k .)

Let $f_i(t_1 \ldots t_m)$ = the maximum attainable reward for a selection of jobs from among $J_1, J_2 \ldots J_i$, subject to the constraint that the completion time of no job is later than t_k for machine k .

Now we have, for $i = 0, 1, \ldots \;, n$:

$$f_i(t_1 \ldots t_m) = f_i(t_1, t_2 \ldots t_{k-1}, D_i, t_{k+1} \ldots t_m), \text{ if } t_k > D_i$$

$$= \max \{f_{i-1}(t_1, t_2, \ldots, t_m), \max_k \{r_{i,k} + f_{i-1}(t_1, t_2, \ldots, t_k - a_{i,k}, \ldots, t_m)\}\} .$$

otherwise

subject to the boundary conditions

$$f_o(t_1,t_2 \ldots t_m) = 0$$

$$f_i(0,0,\ldots,0) = 0$$

$$f_i(t_1,t_2 \ldots t_m) = -\infty \qquad \text{if any } t_k < 0$$

The length of the computation implied by these recursion equations grows as $m \times n \times T^m$. Some saving, of course, can be realized through exploitation of symmetry in the case in which all machines are identical, i.e. $a_{j,p} = a_{j,q}$ and $r_{j,p} = r_{j,q}$ for all j, p, q.

The formulation of recursion equations for the extensions of the problem described in sections 6.3 and 6.4 is quite similar, and results in computations which grow as $m \times n \times T^{2m}$ and $m \times n \times T^m$, respectively. One should compare the extension of the deferral cost case with the solution method given by RothKopf(12) for the multi-machine deferral cost problem without deadlines. In that case, a computation growth of $m \times n \times T^{m-1}$ is possible.

We note that in the generalizations of the problems of sections 6.3 and 6.4, it is possible to have some variation in the characteristics of the individual machines, provided the existence of a single linear ordering is not interfered with. In the case of sets of two machines in series, it must be possible to find a linear ordering such that, for all k,

$$\min(a_{j,k}, b_{j+1,k}) \leq \min(a_{j+1,k}, b_{j,k})$$

and in the case of linear deferral costs,

$$\frac{p_{j,k}}{a_{j,k}} \leq \frac{p_{j+1,k}}{a_{j+1,k}}$$

The rewards, $r_{j,k}$, need be related in no particular way.

APPENDICES

APPENDIX I

BASIC MATHEMATICAL THEORY FOR CHAPTER 2

In the first chapter it was assumed that the penalty functions were continuous and non-decreasing. In order to obtain some of the results presented, however, it is necessary to assume that the functions are absolutely continuous. The assumption of absolute continuity rules out certain unusual functions for which the fundamental theorem of calculus does not hold. This is shown by the following theorem.

THEOREM: A necessary and sufficient condition that a continuous non-decreasing function, f , be absolutely continuous is that

$$\int_a^b f'(x)\,dx = f(b) - f(a)$$

An excellent discussion of absolute continuity can be found in Natanson[(10)] who also provides an example of a function that is continuous, non-decreasing but not absolutely continuous.

THEOREM: Given two absolutely continuous, non-decreasing functions, f and g , where $f \geq g$, $f' \geq g'$, and f' is non-decreasing. For arbitrary $0 < x_1 < x_2$, $0 < x_3 < x_4$ where $x_4 \geq x_2$ and $(x_4 - x_3) \geq (x_2 - x_1)$, $f(x_4) - f(x_3) \geq g(x_2) - g(x_1)$

Proof: Using the fundamental theorem of calculus,

$$\int_{x_3}^{x_4} f'(x)dx = f(x_4) - f(x_3)$$

$$\int_{x_1}^{x_2} g'(x)dx = g(x_2) - g(x_1)$$

Let $x_i = \max(x_2,x_3)$ and $x_k = \min(x_2,x_3)$,

then $f(x_4) - f(x_3) - [g(x_2) - g(x_1)] \geqq$

$$\int_{x_i}^{x_4} f'(x)dx - \int_{x_1}^{x_k} g'(x)dx$$

Now use the following theorem from real analysis:

THEOREM: If f is an integrable function, α and β are real numbers, and E is a measurable set such that, for x in E , $\alpha \leq f(x) \leq \beta$, then $\alpha \cdot \mu(E) \leq \int_E f d\mu \leq \beta\mu(E)$*.

Hence $\int_{x_i}^{x_4} f'(x)dx \geq f'(x_i)\ (x_4 - x_i)$ since f' is non-decreasing.

Further,

$$\int_{x_1}^{x_k} g'(x)\ dx \leq \int_{x_1}^{x_k} f'(x)\ dx \leq f'(x_k)\ (x_k - x_1)$$

since $f' \geq g'$ and f' is non-decreasing.

But $f'(x_i) \geq f'(x_k)$ since $x_i \geq x_k$ and $(x_4 - x_i) \geq (x_k - x_1)$ since $(x_4 - x_3) \geq (x_2 - x_1)$.

Hence, $\int_{x_i}^{x_4} f'(x)\ dx - \int_{x_1}^{x_k} g'(x)\ dx \geq$

$$f'(x_i)\ (x_4 - x_i) - f'(x_k)\ (x_k - x_1) \geq 0$$

or $$f(x_4) - f(x_3) \geq g(x_2) - g(x_1) .$$

*$\mu(E)$ denotes, of course, the measure of the set E in the underlying measure space.

APPENDIX II

BASIC MATHEMATICAL THEORY FOR CHAPTER 3

THEOREM: Consider an absolutely continuous, non-decreasing function, f , and $0 < x < \infty$. If

$$\lim \frac{f'(x+y)}{f'(y)} = 1$$

as $y \to \infty$, then for all $0 < x_1 \leq x_2 < x_3$ where $x_1 + x_2 > x_3$ and $f(x_1) + f(x_2) \leq f(x_3)$ there exists $y > 0$ such that:

$$f(x_1 + y) + f(x_2 + y) > f(x_3 + y) + f(y)$$

Proof: Consider $y > x_3$ and using the fundamental theorem of calculus

$$f(x_1+y) + f(x_2+y) = \int_o^{x_1+y} f'(x)dx + \int_o^{x_2+y} f'(x)dx$$

$$= \int_o^{x_1} f'(x)dx + \int_{x_1}^{y} f'(x)dx + \int_{y}^{x_1+y} f'(x)dx$$

$$+ \int_o^{x_3} f'(x)dx + \int_{x_3}^{x_2+y} f'(x)dx$$

$$f(x_3+y) + f(y) = \int_o^{x_3+y} f'(x)dx + \int_o^{y} f'(x)dx$$

$$= \int_o^{x_3} f'(x)dx + \int_{x_3}^{x_2+y} f'(x)dx + \int_{x_2+y}^{x_3+y} f'(x)dx$$

$$+ \int_o^{x_1} f'(x)dx + \int_{x_1}^{y} f'(x)dx$$

Hence if there exists y such that

$$\int_y^{x_1+y} f'(x)dx > \int_{x_2+y}^{x_3+y} f'(x)dx$$

the theorem is proved.

For f' non-decreasing,

$$\int_{y}^{x_1+y} f'(x)dx \geq x_1 f'(y)$$

$$\int_{x_2+y}^{x_3+y} f'(x)dx \leq (x_3 - x_2)\ f'(x_3 + y)$$

But

$$\lim \left(\frac{f'(x_3+y)}{f'(y)}\right) \to 1 \quad \text{as} \quad y \to \infty$$

Hence, for every $\epsilon > 0$, there exists $M > 0$ such that

$$\left| \frac{f'(y+x_3)}{f'(y)} - 1 \right| < \epsilon \quad \text{for} \quad y > M$$

Now let $\epsilon = \left(\frac{x_1}{x_3-x_2} - 1\right) > 0$ since $x_1 + x_2 > x_3$

or $x_1 > x_3 - x_2$

and select y so that

$$\left| \frac{f'(y+x_3)}{f'(y)} - 1 \right| < \frac{x_1}{x_3-x_2} - 1$$

or

$$\frac{f'(y+x_3)}{f'(y)} < \frac{x_1}{x_3-x_2}$$

Hence,

$$\int_{x_2+y}^{x_3+y} f'(x)dx \leq f'(y+x_3)\ (x_3-x_2) < f'(y)x_1 \leq \int_{y}^{x_1+y} f'(x)dx$$

and $\qquad f(x_1+y) + f(x_2+y) > f(x_3+y) + f(y)$

APPENDIX III

ADDITIONAL MATHEMATICAL THEORY FOR CHAPTER 3

THEOREM: For an absolutely continuous, non-decreasing function f, $f(x) > 0$ for $x > 0$, if f'/f is non-increasing then for every set of points $0 < x_1 < x_2 < x_3$ where $f(x_1) + f(x_2) \geq f(x_3)$, $f(x_1+y) + f(x_2+y) \geq f(x_3+y)$ for all $y \geq 0$.

Proof: Since f'/f is non-increasing,

$$\int_{x_1}^{x_1+y} \frac{f'(x)}{f(x)}\,dx \geq \int_{x_2}^{x_2+y} \frac{f'(x)}{f(x)}\,dx \geq \int_{x_3}^{x_3+y} \frac{f'(x)}{f(x)}\,dx$$

or $$\log f(x) \Big|_{x_1}^{x_1+y} \geq \log f(x) \Big|_{x_2}^{x_2+y} \geq \log f(x) \Big|_{x_3}^{x_3+y}$$

or $$\log f(x_1+y) - \log f(x_1) \geq \log f(x_2+y) - \log f(x_2) \geq \log f(x_3+y) - \log f(x_3)$$

or $$\log \frac{f(x_1+y)}{f(x_1)} \geq \log \frac{f(x_2+y)}{f(x_2)} \geq \log \frac{f(x_3+y)}{f(x_3)}$$

or $$\frac{f(x_1+y)}{f(x_1)} \geq \frac{f(x_2+y)}{f(x_2)} \geq \frac{f(x_3+y)}{f(x_3)}$$

Hence, $$f(x_2+y) \geq \frac{f(x_2)\,f(x_3+y)}{f(x_3)} \geq \frac{f(x_2)\,f(x_3+y)}{f(x_1)+f(x_2)}$$

or $$f(x_2+y)\,[f(x_1) + f(x_2)] \geq f(x_2)\,f(x_3+y)$$

or $$f(x_2+y)\ \frac{f(x_1)}{f(x_2)} + f(x_2+y) \geq f(x_3+y)$$

But $$f(x_1+y) \geq \frac{f(x_1)}{f(x_2)}\ f(x_2+y)$$

Therefore $$f(x_1+y) + f(x_2+y) \geq f(x_3+y)$$

APPENDIX IV

PROOF OF SINGLE TIME DEPENDENCE FOR IDENTICAL QUADRATIC LOSS FUNCTIONS

THEOREM: Given a function, f, of the form $ax + bx^2 (a,b \geq 0)$ and four points $0 \leq x_4 < x_1 < x_2 < x_3$ such that $x_1 + x_2 \geq x_3 + x_4$ and $f(x_1) + f(x_2) > f(x_3) + f(x_4)$, then for all $y \geq 0$, $f(x_1+y) + f(x_2+y) > f(x_3+y) + f(x_4+y)$.

Proof:

$$\begin{aligned} f(x_1+y) + f(x_2+y) &= a(x_1+y) + b(x_1{}^2 + 2x_1 y + y^2) \\ &\quad + a(x_2+y) + b(x_2{}^2 + 2x_2 y + y^2) \\ &= f(x_1) + f(x_2) + 2ay + 2by(x_1+x_2) + 2by^2 \end{aligned}$$

and in like manner,

$$f(x_3+y) + f(x_4+y) = f(x_3) + f(x_4) + 2ay + 2by(x_3+x_4) + 2by^2$$

But $\quad f(x_1) + f(x_2) > f(x_3) + f(x_4)$

and $\quad 2by(x_1 + x_2) \geq 2by(x_3 + x_4)$

Hence $\quad f(x_1+y) + f(x_2+y) > f(x_3+y) + f(x_4+y)$.

APPENDIX V

PROOF THAT R(t) IS TRANSITIVE

Consider jobs J_r and J_s . There are three possible reasons for $J_r \ll (t)\ J_s$.

A. If $t_r \leq t_s$ and $D_r \leq D_s$, $J_r \ll J_s$.

B. If $t_s < t_r$ and $D_r < D_s$ and $t \leq \mathcal{T}_{r,s} = D_s - t_r$, then $J_r \ll (t)\ J_s$.

C. If $t_r < t_s$ and $D_s < D_r$ and $t > \mathcal{T}_{s,r} = D_r - t_s$, then $J_r \ll (t)\ J_s$.

Now consider $J_i \ll (t)\ J_k$ and $J_k \ll (t)\ J_q$. There are three general cases to consider, each having three subcases.

A. Consider $J_i \ll (t)\ J_k$ for reason A above, and hence, $t_i \leq t_k$, $D_i \leq D_k$.

1. If $J_k \ll (t)\ J_q$ by A , then

$$t_i \leq t_k \leq t_q$$

$$D_i \leq D_k \leq D_q$$

and $J_i \ll (t)\ J_q$ for reason A .

2. If $J_k \ll (t)\ J_q$ by B , then

$$t_q < t_k \text{ , } D_k < D_q \text{ , and}$$

$$t < D_q - t_k \text{ .}$$

Hence $D_i \leq D_k < D_q$.

If $t_i \leq t_q$, then $J_i \ll (t)\ J_q$ for reason A .

Therefore, assume $t_q < t_i$.

But $t < D_q - t_k \leq D_q - t_i$ and consequently $J_i \ll (t) J_k$ for reason B .

3. If $J_k \ll (t)\ J_q$ by C , $t_k < t_q$, $D_q < D_k$ and $t > D_k - t_q$.

Hence $t_i \leq t_k < t_q$.

If $D_i \leq D_q$, then $J_i \ll (t)\ J_q$ for reason A .

Therefore, assume $D_q < D_i \leq D_k$.

But $t > D_k - t_q \geq D_i - t_q$ and

hence $J_i \ll (t) J_q$ for reason C .

B. Let $J_i \ll (t) J_k$ for reason B and hence

$$t_k < t_i \text{ , } D_i < D_k \text{ , } \quad \text{and}$$

$$t < D_k - t_i \text{ .}$$

1. If $J_k \ll (t)\ J_q$ by A , then

$$D_i < D_k \leq D_q \quad \text{and}$$

$$t_k \leq t_q$$

If $t_i \leq t_q$, then $J_i \ll (t)\ J_q$ for reason A .

Hence, assume $t_q < t_i$.

But $t < D_k - t_i \leq D_q - t_i$ and $J_i \ll (t)\ J_q$ for reason B .

2. If $J_k \ll (t)\ J_q$ by B , then

$$t_q < t_k, D_k < D_q \text{ , } \quad \text{and}$$

$$t < D_q - t_k \text{ .}$$

Hence $t_q < t_k < t_i$ and $D_i < D_k < D_q$.

But $t < D_k - t_i < D_q - t_i$ and $J_i \ll (t)\ J_q$ for reason B .

3. If $J_k \ll (t)\ J_q$ by C , then

$t_k < t_q$, $D_q < D_k$, and

$t > D_k - t_q$.

Hence, $D_k - t_q < t < D_k - t_i$ and $t_i < t_q$

If $D_i \leq D_q$, then $J_i \ll (t)\ J_q$ by reason A .

Hence, assume $D_q < D_i$.

But $t > D_k - t_q > D_i - t_q$ and $J_i \ll (t)\ J_q$ for reason C .

C. Let $J_i \ll (t)\ J_k$ for reason C and hence,

$t_i < t_k$,

$D_k < D_i$, and

$t > D_i - t_k$.

1. If $J_k \ll (t)\ J_q$ by A , then

$t_i < t_k \leq t_q$ and

$D_k \leq D_q$.

If $D_i \leq D_q$, then $J_i \ll (t)\ J_q$ for reason A .

Hence, assume $D_q < D_i$.

But $t > D_i - t_k \geq D_i - t_q$ and $J_i \ll (t) J_q$ for reason C .

2. If $J_k \ll (t)\ J_q$ by B , then

$t_q < t_k$,

$D_k < D_q$, and

$t \leq D_q - t_k$.

Hence, $D_i - t_k < t \leq D_q - t_k$ and $D_i < D_q$.

If $t_i \leq t_q$, then $J_i \ll (t)\ J_q$ for reason A .

Therefore, assume $t_q < t_i$.

But $t \leq D_q - t_k < D_q - t_i$ and $J_i \ll (t)\ J_q$ for reason B .

3. If $J_k \ll (t)\ J_q$ by C , then

$t_k < t_q$,

$D_q < D_k$, and

$t > D_k - t_q$.

Hence $t_i < t_k < t_q$ and $D_q < D_k < D_i$.

But $t > D_i - t_k > D_i - t_q$ and $J_i \ll (t)\ J_q$ for reason C .

BIBLIOGRAPHY

1. Conway, R. W., Maxwell, W. L., and Miller, L., *Theory of Scheduling*, Addison-Wesley (In Press).

2. Dantzig, George B., *Linear Programming and Extensions*, Princeton University Press, Princeton, New Jersey, 1963.

3. Eastman, Willard L., "Comments on a Paper by Schild and Fredman," *Management Science*, Vol. 11, No. 5 (March, 1965), pp. 754-755.

4. Halmos, Paul R., *Measure Theory*, D. Van Nostrand Company, Inc., Princeton, New Jersey, 1950.

5. Held, Michael, and Karp, Richard M., "A Dynamic Programming Approach to Sequencing Problems," *Journal of the Society for Industrial and Applied Mathematics*, Vol. 10, No. 1 (March 1962), pp. 196-210.

6. Johnson, S. M., "Optimal Two- and Three- Stage Production Schedules with Set-Up Times Included," *Naval Research Logistics Quarterly*, No. 1 (1954), pp. 61-68.

7. Lawler, Eugene L., "On Scheduling Problems with Deferral Costs.," *Management Science*, Vol. 11, No. 2 (November 1964), pp. 280-288.

8. Manne, A. S., "On the Job-Shop Scheduling Problem," *Operations Research*, No. 8, Vol. 2 (March-April 1960), pp. 219-223.

9. McNaughton, Robert, "Scheduling with Deadlines and Loss Functions," *Management Science*, Vol. 6, No. 1 (October 1959), pp. 1-12.

10. Natanson, I. P., *Theory of Functions of a Real Variable*, Frederick Ungar Publishing Co., New York, 1961.

11. Root, James Gordon, "Scheduling with Deadlines and Loss Functions and the Optimal Selection of Resources to Perform Tasks," Ph.D. Thesis, Ohio State University, 1963.

12. Rothkopf, Michael H., "Scheduling Independent Tasks on Parallel Processors," *Management Science*, Vol. 12, No. 6 (January, 1966), pp. 437-447.

13. Schild, Albert, and Fredman, Irwin J., "On Scheduling Tasks with Associated Linear Loss Functions," *Management Science*, Vol. 7, No. 3 (April, 1961), pp. 280-285.

14. ______________ and ______________, "Scheduling Tasks with Deadlines and Non-Linear Loss Functions," *Management Science*, Vol. 9, No. 1 (October, 1962), pp. 73-81.

15. Smith, Wayne E., "Various Optimizers for Single-Stage Productions," *Naval Research Logistics Quarterly*, Vol. 3, Nos. 1-2, March-June, 1956, pp. 59-66.

16. Weisner, Louis, *Introduction to the Theory of Equations*, The MacMillan Company, New York, 1956.