1965

Near optimal sequencing :N jobs and M machines; all jobs to be processed through the same order of machines

Larry Glen Clark

Department:

NEAR OPTIMAL SEQUENCING
N Jobs and M Machines; All Jobs to be Processed Through the Same Order of Machines

BY

LARRY GLEN CLARK

1939

33P

A

THESIS

submitted to the faculty of the

UNIVERSITY OF MISSOURI AT ROLLA

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

Rolla, Missouri

1965

Approved by

Billy E. Gillett (advisor) Earl Richards

Charles E. Antle Hughes M. Zerr

ABSTRACT

There is a need, in industry, for an efficient method of determining an optimal sequence for processing a number of jobs through two or more machines. A method which requires a minimum amount of time would be most beneficial.

The purpose of this study has been to define such a method. Tests on the method described indicate that although an optimal solution is not always obtained, the solutions obtained are quite good and due to the time factor the method would be beneficial to industry.

ACKNOWLEDGEMENT

The author wishes to express his sincere appreciation to Dr. B. E. Gillett for his help in selecting this subject and also for his assistance and guidance throughout the preparation of this thesis.

The author would also like to extend a special thanks to Carolyn Kowalskey for an excellent job of typing this thesis.

TABLE OF CONTENTS

LIST OF TABLES

I. INTRODUCTION

The solution of sequencing problems is an area which has received considerable attention in recent years. Due to the increasing demand on our industrial society the solution of sequencing problems is becoming more important every day.

Consider a company which produces a number of items. Each item must be processed on one, or more, of several machines. Certain machine processes may require that an item be processed on one machine prior to another, therefore a fixed sequence of machines may be necessary, i.e. an item must be cleaned before it can be painted. For other processes, however, it may be possible to deviate from a fixed sequence of machines. An item that requires both the boring of holes and sanding may be sent to the sanders first when the drill presses are overloaded and returned to the drill presses at a later time.

By efficient utilization of the machines' time, such a company would increase their output and expected profits. The availability of a solution for sequencing problems would enable such a company to determine one or more sequences which would utilize their machine time efficiently.

From the previous description, it can be seen that sequencing is concerned with determining an optimal order for a number of jobs to be performed on a number of machines

with regard to some measure of effectiveness(1)*. Various criterion may be chosen for a measure of effectiveness. The most common chosen is the total elapsed time for processing all jobs through all machines, i.e. we wish to determine one or more sequences that will allow all jobs to be processed through all machines in a minimum total elapsed time. Other criteria which may be chosen are the total man hours involved or the expected profits(2). Regardless of the measure of effectiveness chosen for a problem the objective is to determine one or more sequences that will yield an optimal solution with respect to that measure of effectiveness.

Sequencing problems can be classified in two categories. For the first type there are n jobs to be performed, each of which requires processing on one or more of several machines. The objective is to choose, from the $(n!)^m$ theoretically possible sequences, one or more sequences which give an optimal solution with regard to the measure of effectiveness chosen. In problems of this type all jobs must be known before any assignments are made and once processing is started no deviation from the chosen sequence is allowed.

In problems of the second type, again there are m machines and a number of jobs to be performed. However,

---

*All numbers (a) refer to the bibliography while the numbers (a.b) refer to equations.

the number of jobs is dependent on time, that is, new jobs that arrive are allowed to be considered for assignment as well as any jobs which have not been assigned up to that time. The objective now is to decide on the next job to be started each time a machine completes the task on which it is engaged such that an optimal solution with regard to the measure of effectiveness is achieved.

Although both types of problems have proven difficult, solutions for some special cases of the first type have been developed. At the present time there appears to be no mathematical approach to problems of the second type. However, in this paper we will be concerned only with problems of the first type.

The purpose of the present study is to further develop criterion for determining the solution to sequencing problems involving n machines and m jobs, where all jobs are to be processed in a prescribed order on the machines. The investigation will be based on a study of the idle time of the last machine when the order of processing for two jobs is interchanged. This is appropriate since the total elapsed time can be interpreted as the sum of the processing time for all jobs on the last machine and the idle time for that machine. Thus a criterion which will minimize the idle time on the last machine will minimize the total elapsed time for processing all jobs on all the machines.

II. REVIEW OF LITERATURE

In the general sequencing problem of the first type as discussed in Chapter I, there are n jobs (1, 2, $\cdots$, n), each of which must be processed on m machines (A, B, $\cdots$). The problem is to find a sequence (i_1, i_2, $\cdots$, i_n), where (i_1, i_2, $\cdots$, i_n) is a permutation of the integers (1, 2, $\cdots$, n), for each machine such that the total elapsed time is a minimum.

Basic assumptions made for most sequencing models are described by Hardgrave and Nemhauser(3), as the following:

1. The time to process each job on each machine is known.
2. The sequence of machines on which each job is to be processed is known.
3. A job may not be processed by more than one machine at a time.
4. A machine may not process more than one job at a time.
5. Once a machine has begun to process a job, it must complete the job before starting on another.

As stated in Chapter I, solutions exist only for some special cases of sequencing problems. According to Sasieni, Yaspan and Friedman(1), satisfactory solutions are available for only the three following special cases.

1. n jobs and two machines A and B; all jobs processed

in the order AB.

2. n jobs and three machines A, B, and C; all jobs processed in the order ABC.

However, to obtain a solution to the second case, one or both of the following conditions must hold.

Condition 1: The smallest processing time for machine A is at least as great as the largest processing time for machine B.

Condition 2: The smallest processing time for machine C is at least as great as the largest processing time for machine B.

3. two jobs and m machines; each job to be processed through the machines in a prescribed order which is not necessarily the same for both jobs.

As a result of an extension of the procedure developed by Sasieni, Yaspan, and Friedman(1) for the third case, a solution to the more general problem, n jobs and m machines, each job to be processed through the machines in a prescribed order which is not necessarily the same for all jobs, has been obtained by Hardgrave and Nemhauser(3). However for large n, it was found that the procedure required a large amount of time, therefore making it inefficient.

Certainly one method for finding the optimal solution to an n-job m-machine sequencing problem is by calculating

the total elapsed times for all possible sequences. Obviously the number of possibilities becomes quite large for large values of n and m. Therefore the time involved in enumerating all possible sequences makes it infeasible to solve problems by this method. By solving the sequencing problem, it is meant to find a method such that only a minimum number of sequences have to be enumerated.

Basically three methods have been applied to sequencing problems. The first was a non-numerical approach developed by Akers and Friedman(2) for solving problems involving two jobs and m machines; each job to be processed through the machines in a prescribed order which is not necessarily the same for both jobs. The procedure to obtain a solution involves an examination of all possible sequences and by the use of specified rules, all sequences which are not feasible are eliminated. By a feasible sequence, it is meant to be any sequence which can actually be completed. The rules for eliminating the non-feasible sequences were developed by purely logical considerations without regard to any specific numerical data.

Further elimination of possible sequences was obtained by defining rules for determining a set of optimal sequences. The set of optimal sequences have the following two properties:

1. for any assignment of time intervals the optimal sequence will be in the set.

2. every sequence in the set is optimal for some assignment of time intervals.

Although this method eliminates a large number of possible sequences, for large m the number of sequences that must be enumerated is still quite large.

A second method which has been used is a graphical approach, first introduced by Sasieni, Yaspan, and Friedman(1) as a solution to the two-job m-machine problem; each job to be processed through the machines in a prescribed order. Later, Hardgrave and Nemhauser(3) extended the work and developed a geometric algorithm for a solution of the n-job m-machine problem; all jobs to be processed through the machines in the same prescribed order. The procedure is based primarily on the fact that all feasible sequences can be represented geometrically within an n-dimensional closed rectangle. However, Hardgrave and Nemhauser(3) state that for large n this method may require as many as n! trials. Thus the time required makes the use of the method infeasible.

One of the latest attempts at solving sequencing problems has employed dynamic programming concepts. Bellman and Dreyfus(4) have developed a procedure for determining the solution of an n-job two-machine problem where all jobs are to be processed through the machines in the same order. The procedure involves only scanning the processing times of both machines for the minimum time and scheduling

that job either first or last, depending on which machine the minimum time appears under. Bellman and Dreyfus(4) stated that no corresponding solution seemed to exist for the more general problem, n jobs and m machines, all jobs to be processed through the machines in the same order.

Dudek and Ottis(5) announced an algorithm for the solution of the n job m machine problem, all jobs to be processed through the machines in the same order. In the procedure m - 1 conditions have to be satisfied, for an m-machine problem, to determine which of two jobs should come first. Recently, however, Karush(6) has shown an example for which the algorithm does not give the correct result. Karush(6) suggested that the inability of the algorithm to solve all problems may be due to an assumption made in the derivation of the conditions to be satisfied. The assumption questioned was that the order of processing for all jobs, other than the two under consideration, would have no effect on the decision to interchange two jobs.

III. THEORETICAL DEVELOPMENT

The mathematical formulation of the m machine problem deals with the minimization of the idle time on the last machine. This analysis follows Johnson's(6) approach to the two-and three-machine sequencing problem.

Assumptions which are necessary for this development are as follows:

1. All jobs are known and completely organized before any processing is started.
2. Jobs are processed by the machines as soon as possible.
3. No job may be processed on more than one machine at a time.
4. No machine may process more than one job at a time.
5. A job, once started, must be processed to completion.

The following notation will be used throughout this development. Let:

A_{ij} = time required by job i on machine j.

X_{ij} = idle time on machine j from the end of job i - 1 to the start of job i.

T = total elapsed time to process all jobs through all machines.

The problem is to find one or more permutations of the integers 1 through n such that T will be a minimum. For the n job case

$$T = \sum_{i=1}^{n} A_{im} + \sum_{i=1}^{n} X_{im} . \qquad (3.1)$$

Since $\sum_{i=1}^{n} A_{im}$ is fixed once all jobs are known the problem becomes one of minimizing $\sum_{i=1}^{n} X_{im}$.

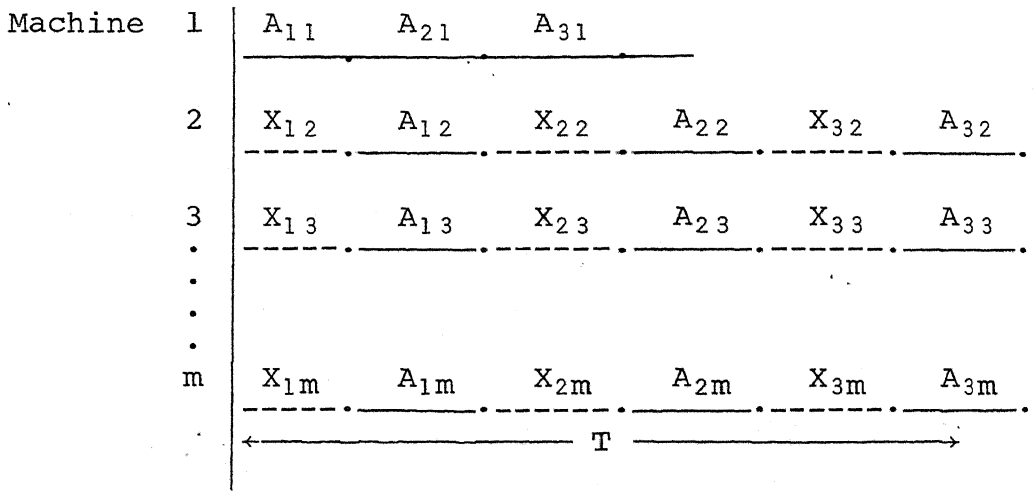

Figure 1. Gantt Chart (m-machines)

From Figure 1, the idle times for job 1 are

$$X_{12} = A_{11}$$

$$X_{13} = X_{12} + A_{12}$$

and generalizing

$$X_{1m} = X_{1m-1} + A_{1m-1} \qquad (3.2)$$

Continuing for job 2,

$$X_{22} = \max[A_{11} + A_{21} - A_{12} - X_{12}; 0]$$

$$X_{23} = \max[X_{12} + A_{12} + X_{22} + A_{22} - A_{13} - X_{13}; 0]$$

and again generalizing

$$X_{2m} = \max[X_{1m-1} + A_{1m-1} + X_{2m-1} + A_{2m-1} - A_{1m} - X_{1m};0]. \quad (3.3)$$

Combining the idle times for jobs 1 and 2 we have

$$X_{12} + X_{22} = \max[A_{11} + A_{21} - A_{12};A_{11}]$$

$$X_{13} + X_{23} = \max[X_{12} + A_{12} + X_{22} + A_{22} - A_{13};X_{12} + A_{12}]$$

$$\vdots$$

$$X_{1m} + X_{2m} = \max[X_{1m-1} + A_{1m-1} + X_{2m-1} + A_{2m-1} - A_{1m};X_{1m-1} + A_{1m-1}]. \quad (3.4)$$

The idle times for job 3 are given by:

$$X_{32} = \max[\sum_{i=1}^{3} A_{i1} - \sum_{i=1}^{2} A_{i2} - \sum_{i=1}^{2} X_{i3};0]$$

$$X_{33} = \max[\sum_{i=1}^{3} X_{i2} + \sum_{i=1}^{3} A_{i2} - \sum_{i=1}^{2} A_{i3} - \sum_{i=1}^{2} X_{i3};0]$$

$$\vdots$$

$$X_{3m} = \max[\sum_{i=1}^{3} X_{im-1} + \sum_{i=1}^{3} A_{im-1} - \sum_{i=1}^{2} A_{im} - \sum_{i=1}^{2} X_{im};0]. \quad (3.5)$$

The total idle time for the first three jobs then becomes

$$\sum_{i=1}^{3} X_{i2} = \max[\sum_{i=1}^{3} A_{i1} - \sum_{i=1}^{2} A_{i2};A_{11} + A_{21} - A_{12};A_{11}]$$

$$\sum_{i=1}^{3} X_{i3} = \max[\sum_{i=1}^{3} X_{i2} + \sum_{i=1}^{3} A_{i2} - \sum_{i=1}^{2} A_{i3};\sum_{i=1}^{2} X_{i2} + \sum_{i=1}^{2} A_{i2} - A_{13};X_{12} + A_{12}]$$

$$\vdots$$

$$\sum_{i=1}^{3} X_{im} = \max[\sum_{i=1}^{3} X_{im-1} + \sum_{i=1}^{3} A_{im-1} - \sum_{i=1}^{2} A_{im};\sum_{i=1}^{2} X_{im-1} + \sum_{i=1}^{2} A_{im-1} - A_{1m};X_{1m-1} + A_{1m-1}]. \quad (3.6)$$

Inductively, the idle times for n jobs are given by the following:

$$\sum_{i=1}^{n} X_{i2} = \max_{1 \le u \le n} K_{u2}$$

$$\text{where } K_{u2} = \sum_{i=1}^{u} A_{i1} - \sum_{i=1}^{u-1} A_{i2}$$

$$\sum_{i=1}^{n} X_{i3} = \max_{1 \le u \le n} K_{u3}$$

$$\text{where } K_{u3} = \sum_{i=1}^{u} X_{i2} + \sum_{i=1}^{u} A_{i2} - \sum_{i=1}^{u-1} A_{i3}$$

$$\vdots$$

$$\sum_{i=1}^{n} X_{im} = \max_{1 \le u \le n} K_{um} \tag{3.7}$$

$$\text{where } K_{um} = \sum_{i=1}^{u} X_{im-1} + \sum_{i=1}^{u} A_{im-1} - \sum_{i=1}^{u-1} A_{im} .$$

The problem becomes one of finding a permutation of the integers 1, 2, $\cdots$, n, such that the expression in equation (3.7) is a minimum. Rewriting the expression for K_{um} using a recurrence relation we have

$$K_{um} = \sum_{i=1}^{u} A_{im-1} - \sum_{i=1}^{u-1} A_{im} + \max_{1 \le v \le u} K_{vm-1} . \tag{3.8}$$

and equation (3.7) can be written as

$$\sum_{i=1}^{n} X_{im} = \max_{1 \le u \le m} \left[\sum_{i=1}^{u} A_{im-1} - \sum_{i=1}^{u-1} A_{im} + \max_{1 \le v \le u} K_{vm-1} \right] . \tag{3.9}$$

To determine if job k should precede job k + 1, define two sequences

$S_1 = 1, 2, 3, \cdots, k-1, k, k+1, k+2, \cdots, n,$

and $S_2 = 1, 2, 3, \cdots, k-1, k+1, k, k+2, \cdots, n.$

Let

$K_{uj}^{(1)}$ represent the value of K_{uj} for sequence S_1,

and $K_{uj}^{(2)}$ represent the value of K_{uj} for sequence S_2.

Also let

$$I^{(1)} = \sum_{i=1}^{n} X_{im} \text{ for sequence } S_1 \text{ and,}$$

$$I^{(2)} = \sum_{i=1}^{n} X_{im} \text{ for sequence } S_2.$$

Now for $u = 1, 2, \cdots, k-1$, $K_{uj}^{(1)} = K_{uj}^{(2)}$; but for $u = k, k+1, k+2, \cdots, n$, $K_{uj}^{(1)}$ does not necessarily equal $K_{uj}^{(2)}$. This might make the idle time, $I^{(1)}$, for sequence S_1 different than the idle time, $I^{(2)}$, for sequence S_2, thereby making sequence S_2 preferable to sequence S_1 if

$$I^{(2)} < I^{(1)}.$$

The following statement can now be made: Job $k+1$ precedes job k if:

$$\max_{1 \le u \le n} [K_{um}^{(2)}] < \max_{1 \le u \le m} [K_{um}^{(1)}]. \qquad (3.10)$$

IV. COMPUTATIONAL TECHNIQUE

The computation involved to determine whether job $k + 1$ should precede job k involves only sums and differences of the processing times. However by an examination of equation (3.8) a relationship between the values of $K_{uj}^{(1)}$ and $K_{uj}^{(2)}$ can be determined.

Let

$$Q_{uj} = \sum_{i=1}^{u} A_{ij-1} - \sum_{i=1}^{u-1} A_{ij} \quad ; \; j = 2, 3, \cdots, m. \tag{4.1}$$

Also let $Q_{uj}^{(1)}$ represent the value of Q_{uj} for sequence S_1 and $Q_{uj}^{(2)}$ represent the value of Q_{uj} for sequence S_2.

For sequence S_1:

$$Q_{uj}^{(1)} = \sum_{i=1}^{u} A_{ij-1} - \sum_{i=1}^{u-1} A_{ij} \qquad u = 1, 2, \cdots, k - 1$$

$$Q_{kj}^{(1)} = \sum_{i=1}^{k} A_{ij-1} - \sum_{i=1}^{k-1} A_{ij}$$

$$Q_{k+1j}^{(1)} = \sum_{i=1}^{k+1} A_{ij-1} - \sum_{i=1}^{k} A_{ij} \tag{4.2}$$

$$Q_{uj}^{(1)} = \sum_{i=1}^{u} A_{ij-1} - \sum_{i=1}^{u-1} A_{ij} \qquad u = k + 2, k + 3, \cdots, n.$$

For sequence S_2:

$$Q_{uj}^{(2)} = \sum_{i=1}^{u} A_{ij-1} - \sum_{i=1}^{u-1} A_{ij} \qquad j = 2, 3, \cdots, m$$

$$Q_{kj}^{(2)} = \sum_{i=1}^{k+1} A_{ij-1} + A_{k+1j-1} - \sum_{i=1}^{k-1} A_{ij}$$

$$Q^{(2)}_{k+1j} = \sum_{i=1}^{k+1} A_{ij-1} - \sum_{i=1}^{k-1} A_{ij} - A_{k+1j} \tag{4.3}$$

$$Q^{(2)}_{uj} = \sum_{i=1}^{u} A_{ij-1} - \sum_{i=1}^{u-1} A_{ij} \qquad u = k+2, k+3, \cdots, n.$$

Comparing equations (4.2) and (4.3) it can be observed that the following relations hold.

$$\begin{aligned} Q^{(2)}_{uj} &= Q^{(1)}_{uj} \qquad u = 1, 2, \cdots, k-1 \\ Q^{(2)}_{kj} &= Q^{(1)}_{kj} + A_{k+1j-1} - A_{kj-1} \\ Q^{(2)}_{k+1j} &= Q^{(1)}_{k+1j} + A_{kj} - A_{k+1j} \\ Q^{(2)}_{uj} &= Q^{(1)}_{uj} \qquad u = k+2, k+3, \cdots, n. \end{aligned} \tag{4.4}$$

Rewriting equation (3.8) in expanded form

$$K_{um} = \sum_{i=1}^{u} A_{im-1} - \sum_{i=1}^{u-1} A_{im} + \max_{1\le v\le u}\left[\sum_{i=1}^{v} A_{im-2} - \sum_{i=1}^{v-1} A_{im-1} + \max_{1\le w\le v}\left[\sum_{i=1}^{w} A_{im-3} - \sum_{i=1}^{w-1} A_{im-2} + \max_{1\le x\le w}\left[\cdots + \max_{1\le b\le a}\left[\sum_{i=1}^{b} A_{i2} - \sum_{i=1}^{b-1} A_{i3} + \max_{1\le c\le b}\left[\sum_{i=1}^{c} A_{i1} - \sum_{i=1}^{c-1} A_{i2}\right]\right]\right]\right]\right]$$

and making the substitution

$$Q_{uj} = \sum_{i=1}^{u} A_{ij-1} - \sum_{i=1}^{u-1} A_{ij}$$

equation (3.8) takes the form

$$K_{um} = Q_{um} + \max_{1\le v\le u}\left[Q_{vm-1} + \max_{1\le w\le v}\left[Q_{wm-2} + \max_{1\le x\le w}\left[\cdots + \max_{1\le b\le a}\left[Q_{b3} + \max_{1\le c\le b} Q_{c2}\right]\right]\right]\right]. \qquad (4.5)$$

Note that, $Q_{c2} = K_{c2} = \sum_{i=1}^{c} A_{i1} - \sum_{i=1}^{c} A_{i2}$.

A simple computation scheme for both values of $K_{um}^{(1)}$ and $K_{um}^{(2)}$ can now be noted. Starting the computation by considering only machines one and two and making use of the relation given by equations (4.4) a minimum number of calculations is required.

To keep the computing time to a minimum for large values of n and m, a method was needed to obtain an "initial" sequence such that the number of interchanges of jobs could be kept as small as possible. After examining the solutions to a number of problems it was observed that the solution obtained by the following rules required very few interchanges to obtain an optimal sequence.

1. For machines one through m - 1, calculate the total processing time for each job.
2. For machines two through m, calculate the total processing time for each job.
3. Scan the processing times for all jobs calculated in steps 1 and 2 for the minimum time.
4. If the minimum time appears in the times calculated in step 1, assign the corresponding job first.

If the minimum time appears in the times calculated in step 2, assign the corresponding job last.

5. Delete the times corresponding to the job assigned and repeat steps 1 through 5 on the remaining times until all jobs have been assigned.

A near optimal sequence can now be obtained by use of the following steps:

Step 1: Compute $M_i' = \sum_{k=1}^{m-1} A_{ik}$; $i = 1, 2, \cdots, n$

and $M_j'' = \sum_{k=2}^{m} A_{ik}$; $j = 1, 2, \cdots, n$.

Step 2: Determine the minimum of M_i' and M_j''.

a. If the minimum is a M_i', assign job i first.

b. If the minimum is a M_j'', assign job j last.

c. If a tie exists, assign job i first.

Step 3: Eliminate all jobs assigned by the procedure of (2) and repeat procedure (2) for all remaining jobs until all assignments have been made.

Step 4: Arrange the processing time matrix to correspond to the sequence determined by steps 1 through 3.

Step 5: For jobs k and k + 1 calculate the values of $K_{um}^{(1)}$ and $K_{um}^{(2)}$ by use of equations (4.2), (4.4) and (4.5).

Step 6: Do one of the following:

a. If $K_{um}^{(2)} < K_{um}^{(1)}$ interchange jobs k and k + 1. Proceed to Step 7.

b. If $K_{um}^{(1)} = K_{um}^{(2)}$ store an optional sequence with jobs k and k + 1 interchanged. Proceed to Step 7.

c. If $K_{um}^{(1)} < K_{um}^{(2)}$ continue with job k preceding job k + 1.

Step 7: Do one of the following:

a. If k + 1 < n, increase k by one and proceed to Step 5.

b. If k + 1 = n, the last job to be processed has been determined. Decrease n by one.

Step 8: Do one of the following:

a. If n > 1, set k equal to one and proceed to Step 5.

b. If n = 1, a near optimal sequence has been determined.

Step 9: Enumerate all sequences found to be near optimal.

While the procedure is primarily designed for machine computation, hand computation can be made without difficulty when $m \leq 4$ and $n \leq 4$. To illustrate the method of solving small problems using this technique consider the following problem taken from Karush(6), page 325:

Problem

Consider three jobs 1, 2, 3 with operation times given by the following table:

		Machine		
		A	B	C
	1	3	22	2
Job	2	22	20	20
	3	20	14	18

Applying the steps described

$M_1' = 25$ $\quad$ $M_1'' = 24$

$M_2' = 42$ $\quad$ $M_2'' = 40$

$M_3' = 34$ $\quad$ $M_3'' = 32$

Thus the initial sequence is

2, 3, 1.

Place job 2 in sequence position 1, job 3 in position 2 and job 1 in position 3. The processing time matrix becomes:

		Machine			
		A	B	C	
	2	22	20	20	1
Job	3	20	14	18	2
	1	3	22	2	3

Sequence Position

Compare jobs 3 and 2

$Q_{12}^{(1)} = 22$ $\quad$ $Q_{12}^{(2)} = 22+20-22 = 20$

$Q_{22}^{(1)} = 22+20-20 = 22$ $\quad$ $Q_{22}^{(2)} = 22+20-14 = 28$

$Q_{32}^{(1)} = 22+3-14 = 11$ $\quad$ $Q_{32}^{(2)} = 11$

$Q_{13}^{(1)} = 20$ $\quad$ $Q_{13}^{(2)} = 20+14-20 = 14$

$Q_{23}^{(1)} = 20+14-20 = 14$ $\quad$ $Q_{23}^{(2)} = 14+20-18 = 16$

$Q_{33}^{(1)} = 14+22-18 = 18 \qquad Q_{33}^{(2)} = 18$

$$\max_{1\le u\le 3}\left[K_{u3}^{(1)}\right] = \max[22+20;22+14;22+18] = 42$$

$$\max_{1\le u\le 3}\left[K_{u3}^{(2)}\right] = \max[20+14;28+16;28+18] = 46$$

Now $\max_{1\le u\le 3}\left[K_{u3}^{(1)}\right] < \max_{1\le u\le 3}\left[K_{u3}^{(3)}\right]$; therefore job 2 preceeds job 3.

Compare jobs 3 and 1

$$\max_{1\le u\le 3}\left[K_{u3}^{(1)}\right] = \max[22+20;22+14;22+18] = 42$$

$$\max_{1\le u\le 3}\left[K_{u3}^{(2)}\right] = \max[22+20;22+22;22+34] = 56$$

Again $\max_{1\le u\le 3}\left[K_{u3}^{(1)}\right] < \max_{1\le u\le 3}\left[K_{u3}^{(2)}\right]$; hence job 3 preceeds job 1.

Since no jobs have been interchanged, the sequence 2, 3, 1 must be optimal. By direct enumeration the following elapsed times are computed to be:

T(123) = 83

T(132) = 85

T(312) = 96

T(321) = 86

T(231) = 82

T(213) = 96

Hence the sequence 2, 3, 1 is indeed the optimal sequence.

V. RESULTS AND CONCLUSIONS

Throughout this chapter, Method A will denote the method developed in this paper for solving sequencing problems. Method B will denote the method enumerating all possible sequences to determine one or more optimal solutions to a sequencing problem.

Evaluation of Method A was accomplished by comparing the sequence, or sequences, generated by the use of Method A, with the optimal sequence, or sequences found by Method B for many problems. An IBM 1620 digital computer was programmed to generate all of the sequences obtainable using both methods. In all problems tested the elements of the processing time matrix consisted of uniformly distributed random numbers between 0 and 1000, thus subjecting Method A to its most severe test. One hundred fifty-six 3-machine, twenty-one 4-machine, twenty-three 5-machine, and nineteen 6-machine problems with n ranging from three to seven were tested for a total of two hundred twenty problems.

Although Method A did not give an optimal sequence for every problem, it was found that the sequences generated were always better than 95% of all possible sequences. In more than one-half of the problems in which an optimal sequence was not obtained, the sequence generated was the second best sequence.

The relative efficiency of Method A is shown by the

data presented in Tables I through III. An examination of these data will indicate that: (a) on the average the percentage of correct results decreases as n increases; (b) the percentage of correct results decreases as m increases; (c) in general the number of optimal sequences increases as n increases; (d) the number of optimal sequences generated by Method A increases as n increases; and (e) the ratio of time required by Method A to determine an optimal sequence to the time for Method B to generate all possible sequences decreases as the number of jobs increase. However for $n \leq 5$ and $m \leq 4$ the data indicates that it would be best to enumerate all possible sequences or to use Method A with hand computation.

As stated previously Method A does not guarantee an optimal sequence. However, for problems with $n \geq 6$, finding a sequence better than 95% of all possible sequences in a relatively small amount of time should make the use of Method A well worthwhile.

TABLE I

Results Obtained Using Method A

No. of Jobs	No. of Machines	No. of Prob. Tested	No. Correct	Percent Correct
3	3	57	56	98
4	3	53	45	85
5	3	33	25	76
6	3	12	12	100
7	3	2	2	100
3	4	7	7	100
4	4	7	7	100
5	4	3	2	67
6	4	4	2	50
3	5	6	6	100
4	5	3	2	67
5	5	7	4	57
6	5	7	6	86
3	6	6	6	100
4	6	2	0	0
5	6	4	2	50
6	6	7	3	43
	Total	220	187	85

TABLE II

Results Obtained for Multiple Solutions

No. of Jobs	No. of Machines	No. of Prob. Having Multiple Solutions	No. of Optimal Solutions Generated	
			Enumeration (Method B)	Method A
3	3	10	2	2
4	3	13	2	2
		4	3	2
		2	3	3
		1	4	2
		1	4	4
		1	5	4
5	3	6	2	2
		1	3	2
		2	3	3
		2	4	4
		1	6	3
		1	9	5
		1	18	7
6	3	3	2	2
		1	4	2
		1	4	4
		1	9	6
		1	20	8
		1	28	3
		1	60	8
		1	96	16
7	3	1	34	12

TABLE II (continued)

No. of Jobs	No. of Machines	No. of Prob. Having Multiple Solutions	No. of Optimal Solutions Generated	
			Enumeration (Method B)	Method A
4	4	1	2	1
		2	2	2
		1	6	2
6	4	1	3	2
		1	4	2
3	5	1	2	2
4	5	1	2	1
5	5	1	8	4
6	5	1	2	2
		1	3	2
		1	7	4
		1	21	1
		1	32	4
5	6	1	2	2
6	6	1	6	4
		1	22	4

TABLE III

Comparison of Times Required to Determine Optimal Solution by Method A and by Enumeration of all Possible Sequences (Time in Minutes)

No. of Jobs	No. of Machines	Enumeration (Method B)	Method A	
			Mean	Range
3	3	.025	.050	.025-.067
4	3	.128	.098	.067-.167
5	3	.645	.201	.116-.550
6	3	4.470	.970	.167-2.58
7	3	36.410	.383	.267-.500
3	4	.041	.070	.050-.083
4	4	.158	.090	.067-.116
5	4	.850	.191	.133-.450
3	5	.050	.067	.050-.083
4	5	.168	.108	.100-.133
5	5	1.050	.215	.116-.500
3	6	.055	.088	.067-.133
4	6	.200	.130	.100-.133
5	6	1.283	.236	.200-.300

APPENDIX

Program for Optimal Sequencing, Method A

```
C       DETERMINATION OF OPTIMAL SEQUENCES
C       N JOBS AND M MACHINES
C       ALL JOBS TO BE PROCESSED THROUGH THE MACHINES
C       IN THE SAME ORDER
        DIMENSION A(9,9),B(9,9),T(9),IN(9,50)
        DIMENSION SM(9),SN(9),P(9),R(9),RP(9)
    5   READ 80,N,M,LAST
        READ 81,((B(I,J),J=1,M),I=1,N)
        PRINT 82
        PRINT 83,(I,I=1,M)
        DO 20 I=1,N
   20   PRINT 84,I,(B(I,J),J=1,M)
        MM=M-1
        DO 37 I=1,N
        SM(I)=B(I,1)
        SN(I)=B(I,2)
        DO 37 J=2,MM
        SM(I)=SM(I)+B(I,J)
   37   SN(I)=SN(I)+B(I,J+1)
        II=1
        NN=N
   45   SMIN=SM(1)
        J=1
        SNIN=SN(1)
        K=1
        DO 38 I=2,N
        IF(SM(I)-SMIN)39,40,40
   39   SMIN=SM(I)
        J=I
   40   IF(SN(I)-SNIN)41,38,38
   41   SNIN=SN(I)
        K=I
   38   CONTINUE
        IF(SMIN-SNIN)43,43,42
   42   IN(NN,1)=K
        NN=NN-1
        SM(K)=1.E8
        SN(K)=1.E8
        GO TO 46
   43   IN(II,1)=J
        II=II+1
        SM(J)=1.E8
        SN(J)=1.E8
   46   DO 44 I=1,N
        IF(SM(I)-1.E8)45,44,45
   44   CONTINUE
        PRINT 87
        PRINT 80,(IN(I,1),I=1,N)
```

```
      PRINT 88
      NS=1
      NP=N
      MM=M-1
   15 NSP=2
   16 KSP=NSP-1
      DO 48 I=1,N
      K=IN(I,1)
      DO 48 J=1,M
   48 A(I,J)=B(K,J)
      DO 149 J=1,MM
      P(J)=A(1,J)
  149 R(J)=A(1,J)
      DO 190 J=2,MM
  190 R(J)=R(J)+R(J-1)
      IF(KSP-1)152,153,152
  153 DO 166 J=1,MM
  166 RP(J)=A(2,J)
      DO 191 J=2,MM
  191 RP(J)=RP(J)+RP(J-1)
      GO TO 167
  152 DO 192 J=1,MM
  192 RP(J)=R(J)
      MP=KSP-1
      IF(KSP-2)212,213,212
  212 DO 150 I=2,MP
      P(1)=P(1)+A(I,1)-A(I-1,2)
      IF(R(1)-P(1))179,180,180
  179 R(1)=P(1)
      RP(1)=P(1)
  180 DO 150 J=2,MM
      P(J)=P(J)+A(I,J)-A(I-1,J+1)
      Q=P(J)+R(J-1)
      IF(R(J)-Q)151,150,150
  151 R(J)=Q
      RP(J)=Q
  150 CONTINUE
  213 P(1)=P(1)+A(KSP,1)-A(MP,2)
      Q=P(1)+A(NSP,1)-A(KSP,1)
      IF(R(1)-P(1))154,155,155
  154 R(1)=P(1)
  155 IF(RP(1)-Q)156,158,158
  156 RP(1)=Q
  158 DO 159 J=2,MM
      P(J)=P(J)+A(KSP,J)-A(MP,J+1)
      Q1=P(J)+R(J-1)
      Q=P(J)+A(NSP,J)-A(KSP,J)
      Q2=Q+RP(J-1)
```

```
      IF(R(J)-Q1)201,202,202
  201 R(J)=Q1
  202 IF(RP(J)-Q2)203,159,159
  203 RP(J)=Q2
  159 CONTINUE
  167 P(1)=P(1)+A(NSP,1)-A(KSP,2)
      Q=P(1)+A(KSP,2)-A(NSP,2)
      IF(R(1)-P(1))204,205,205
  204 R(1)=P(1)
  205 IF(RP(1)-Q)206,207,207
  206 RP(1)=Q
  207 DO 208 J=2,MM
      P(J)=P(J)+A(NSP,J)-A(KSP,J+1)
      Q1=P(J)+R(J-1)
      Q=P(J)+A(KSP,J+1)-A(NSP,J+1)
      Q2=Q+RP(J-1)
      IF(R(J)-Q1)209,210,210
  209 R(J)=Q1
  210 IF(RP(J)-Q2)211,208,208
  211 RP(J)=Q2
  208 CONTINUE
      NR=NSP+1
      DO 162 I=NR,N
      P(1)=P(1)+A(I,1)-A(I-1,2)
      IF(R(1)-P(1))175,176,176
  175 R(1)=P(1)
  176 IF(RP(1)-P(1))177,178,178
  177 RP(1)=P(1)
  178 DO 162 J=2,MM
      P(J)=P(J)+A(I,J)-A(I-1,J+1)
      Q1=P(J)+R(J-1)
      Q2=P(J)+RP(J-1)
      IF(R(J)-Q1)163,164,164
  163 R(J)=Q1
  164 IF(RP(J)-Q2)165,162,162
  165 RP(J)=Q2
  162 CONTINUE
      IF(R(MM)-RP(MM))73,74,75
   75 NEX=IN(KSP,1)
      DO 76 I=1,NS
      DO 62 J=1,N
      IF(NEX-IN(J,I))62,63,62
   62 CONTINUE
   63 IN(J,I)=IN(NSP,I)
   76 IN(NSP,I)=NEX
      GO TO 73
   74 NX=NS
      NS=NS+NS
```

```
      NEX=IN(KSP,1)
      DO 77 J=1,NX
      K=NS-J+1
      KK=K-NS/2
      DO 78 I=1,N
   78 IN(I,K)=IN(I,KK)
      DO 60 I=1,N
      IF(NEX-IN(I,K))60,61,60
   60 CONTINUE
   61 IN(I,K)=IN(NSP,K)
   77 IN(NSP,K)=NEX
      NX=NS-1
      DO 56 K=1,NX
      KK=K+1
      DO 56 J=KK,NS
      DO 57 I=1,N
      IF(IN(I,K)-IN(I,J))56,57,56
   57 CONTINUE
      DO 59 I=1,N
   59 IN(I,J)=IN(I,NS)
      NS=NS-1
      NX=NS-1
      K=K-1
   56 I=0
   73 NSP=NSP+1
      IF(NSP-NP)16,16,17
   17 NP=NP-1
      IF(NP-1)15,14,15
   14 DO 12 L=1,NS
      DO 9 I=1,N
    9 IN(I,1)=IN(I,L)
      DO 8 I=1,N
      K=IN(I,1)
      DO 8 J=1,M
    8 A(I,J)=B(K,J)
      T(1)=A(1,1)
      DO 7 I=2,M
    7 T(I)=T(I-1)+A(1,I)
      DO 6 I=2,N
      T(1)=T(1)+A(I,1)
      DO 6 J=2,M
      IF(T(J-1)-T(J))4,4,3
    4 T(J)=T(J)+A(I,J)
      GO TO 6
    3 T(J)=T(J-1)+A(I,J)
    6 CONTINUE
      PRINT 85,T(M)
```

```
12 PRINT 86,(IN(I,1),I=1,N)
   IF(LAST)5,5,2
 2 CALL EXIT
80 FORMAT(9I5)
81 FORMAT(9F7.0)
82 FORMAT(//10X21HPROCESSING TIME, HRS.)
83 FORMAT(5H  JOB3X2HA(I1,1H),9(3X2HA(I1,1H)))
84 FORMAT(I5,9F7.0)
85 FORMAT(15H  ELAPSED TIME F7.0)
86 FORMAT(5X8HSEQUENCE9I5)
87 FORMAT(//18H  INITIAL SEQUENCE)
88 FORMAT(//20H  SOLUTIONS OBTAINED)
   END
```

BIBLIOGRAPHY

1. Sasieni, Maurice, Yaspan, Arthur, and Friedman, Lawrence (1959) Operations Research-Methods and Problems, John Wiley and Sons, Inc., New York, p. 250-269.

2. Akers, Sheldon B., Jr. and Friedman, Joyce (1955) A Non-numerical Approach to Production Scheduling Problems, Opns. Res., Vol. 3, No. 2, p. 429-442.

3. Hardgrave, William W. and Nemhauser, George L. (1963) A Geometric Model and a Graphical Algorithm for a Sequencing Problem, Opns. Res., Vol. 11, No. 6, p. 849-900.

4. Bellman, Richard E. and Dreyfus, Stuart E. (1962) Applied Dynamic Programming, Princeton University Press, New Jersey, p. 142-146.

5. Dudek, Richard A. and Ottis, Tenton Foy, Jr. (1964) Development of M-Stage Decision Rule for Scheduling n Jobs Through m Machines, Opns. Res., Vol. 12, No. 3, p. 471-497.

6. Karush, William (1965) A Counterexample to a Proposed Algorithm for Optimal Sequencing of Jobs, Opns. Res., Vol. 13, No. 2, p. 323-325.

7. Johnson, S. M. (1954) Optimal Two and Three Stage Production Schedules With Set-up Times Included, Nav. Res. Log. Quart., Vol. 1, No. 1, p. 61-68.

VITA

The author was born on November 26, 1939, in Mulvane, Kansas. He received his primary and secondary education in Conway Springs, Kansas. He has received his college education from Kansas State Teachers College in Emporia, Kansas, Kansas State University in Manhattan, Kansas, and The University of Missouri at Rolla, Rolla, Missouri. In January, 1963, he received a Bachelor of Science degree in Education, Majors in Mathematics and Physical Science, from Kansas State Teachers College and was employed as a mathematics and science teacher for the 1963-64 school term in Herington, Kansas.

He has been enrolled in the Graduate School of The University of Missouri at Rolla since June, 1964, and was a graduate assistant in the Computer Science Center from June, 1964, to May, 1965.