THE UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

A MULTIVARIATE COMPARISON OF FLOWSHOP ALGORITHMS

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

degree of

DOCTOR OF PHILOSOPHY

BY

ARTHUR W. SMITH

Norman, Oklahoma

1979

A MULTIVARIATE COMPARISON OF FLOWSHOP ALGORITHMS

APPROVED BY

Edward F. Stafford

[illegible]

Daniel A. Wren

James F. Horrell

DISSERTATION COMMITTEE

ACKNOWLEDGEMENTS

The author extends his thanks to his dissertation committee, composed of Professors E. F. Stafford, D. A. Wren, B. G. Schumacher and J. F. Horrell. Special thanks is extended to Professor E. F. Stafford, the committee chairman, for his continual guidance through all stages of the research investigation and dissertation preparation.

In addition, a number of people generously provided assistance by suggesting, explaining or discussing analytical techniques for this research. The author would like to acknowledge the contributions of those people. Among them were Professors C. Watson, A. Nicewander, C. Liew and W. Greaves, all of the University of Oklahoma; Professor N. Timm of the University of Pittsburgh; Professor S. Halperin of Syracuse University; Professor J. Carlson of the University of Ottawa; and Professor M. Davidson of the University of Rochester. The author also wishes to thank those people whose names were not mentioned but, nevertheless, their contributions are appreciated and highly valued.

Professors J. Carlson and N. Timm kindly provided a copy of their FRMLM computer program which was used to perform multivariate analyses. The author is truly grateful to them for their generosity.

This work was done in connection with graduate study at the University of Oklahoma. The author would also like to express his gratitude to that institution for creating the academic environment in which the skills and knowledge were acquired to perform this study. Thanks are also due to Syracuse University which provided the facilities to continue this research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

A MULTIVARIATE COMPARISON OF FLOWSHOP ALGORITHMS

CHAPTER 1

INTRODUCTION

Using a multivariate technique, this study will examine and compare a number of flowshop heuristic algorithms. This technique permits comparisons to be made on the basis of multiple criteria. In addition, the effect of the problem size on the efficiency of the algorithms is examined. Any interaction effect between the algorithms and the problem sizes is investigated and taken into consideration when interpreting other aspects of this study. The computational efforts that the algorithms require to solve the problems are also compared and reported. It is believed that these types of comparisons provide information which practitioners can utilize in decisions concerning the selection of heuristic algorithms to solve scheduling problems.

Histortical Development

Since ancient times, the need for scheduling tasks and facilities has confronted man. Plato wrote about the advantages of the division of labor, and the Greeks practiced it extensively [30]. Plato mentions as an example, the specialization involved in the production of shoes where some workers became specialists in cutting the leather while others specialized in stitching. If the specialization was also applied to the repair of chariots and horse-drawn vehicles, it would have been necessary to schedule these vehicles through the various work centers in order to return them to productive use as soon as possible.

The arrival of the Industrial Revolution brought about a greater demand for efficiency. Improving the methods of performing the tasks by utilizing the concepts of work and motion study occupied the interest of many of the scientific management pioneers, the most notable of which were Frederick Taylor and Frank Gilbreth. Another pioneer of this era was Henry Gantt, who invented the Gantt chart, which is used as an aid in solving problems. This device is most frequently used in the construction industry and in project scheduling, but can also be utilized in jobshop and flowshop scheduling. Basically, this chart displays the sequence of the activities involved in the project and also represents their durations on a horizontal scale, thus permitting an observer to determine the duration of the entire project as well as the delays and slack embedded within the project. The activities in a flowshop can also be represented with a Gantt chart. A flowshop exists when a batch of jobs are to undergo processing on a number of machines, with the restriction that jobs must be processed in the same technological order, i.e. each job must first be processsed by machine 1, then by machine 2 and so on. The batch assembly line is an example of a flowshop where the jobs are identical. In other words, each automobile, transistor radio, or mass-produced item which leaves the assembly line is identical. In a machine shop, however, the jobs tend not to be identical. The flowshop problem, therefore, is to sequence the batch of jobs through the various machines so as to minimize some measure of performance.

Initially, researchers concentrated on the goal of determining the optimal schedule according to some criterion of performance, which was usually the minimization of makespan [65,76,112]. The most comprehensive manner in which this can be done is to examine all possible schedules. This is called complete enumeration, and its utility diminishes as the number of jobs increases because the number of possible schedules increases astronomically. Thus, the computational effort involved becomes excessive. If there are $\underline{n}$ jobs to be sequenced, then there are $\underline{n}!$ possible schedules to be examined. For example, 4 jobs yield 24 possible schedules, 5 jobs yield 120, 6 jobs yield 720 and 10 jobs yield over 3,600,000 possible schedules. Ten jobs are a rather modest number to be scheduled, but the calculations involved in determining the optimal schedule would occupy a modern high-speed computer for at least thirty minutes, a considerable and costly amount of time. It was therefore necessary to develop techniques which required much less computational effort.

In 1954, Johnson [65] developed an algorithm which reduced computational effort considerably. In fact, the computations required could be done by hand. His algorithm, however, has very limited utility since it was designed to solve problems with two machines and special cases with three machines. Nevertheless, it was a very significant step in reducing computational effort for these limited situations, and it was one of the earlier attempts to use an analytical model in the solution of the problem. Johnson was also among the first to adopt makespan as a measure of performance, an action which was later emulated by the majority of subsequent researchers.

Other approaches soon followed. First, the integer linear programming approach appeared in the late 1950's. Listed among the authors in this field are Wagner [112], Bowman [24] and Manne [76]. Conceptually, this method appears to be very appealing, but results have indeed been very disappointing. In order to formulate an *n* job problem by this method, 4*n* equations are required along with n-squared unknowns. Due to these relatively large numbers of equations and unknowns which require a great many computations, this technique has been somewhat abandoned. Wagner [112] commented as follows:

> As will be evident below, the model (integer linear programming) in its present form is computationally unwieldy except perhaps for situations with a very few machines and a limited number of items; in such cases, a frequently recurring sequencing problem or one involving a considerable financial sum might profitably be solved by the method herein.

Gupta [48] also commented on the limited scope of this approach. He pointed out that Manne's suggestion for generalizing the formulation in order to account for the occasional lateness of jobs would lead to a reduction in the chances of obtaining a feasible solution. Nevertheless, when feasible solutions were obtained, they required less computational effort than did complete enumeration.

The branch-and-bound method was introduced in 1965 in two separate papers, one by Ignall and Schrage [64] and the other by Lomnicki [73]. Since then, several variations of this technique have appeared [13,75,87]. The major differences have been in the bounding procedures, which are designed to eliminate less efficient schedules from consideration. Some of the bounding procedures were studied and compared by Baker [14]. The branch-and-bound method that incorporates a backtracking procedure guarantees optimal solutions. When this latter procedure is excluded, optimal solutions may or may not be obtained. As $\underline{n}$, the number of jobs, and $\underline{m}$, the number of machines, increase, the likelihood of the problem being solved by the branch-and-bound method decreases because an increase in $\underline{n}$ causes the number of branches to grow by a factor of 2^n. This growth eventually begins to overburden the computer's capacity to carry out the number of needed computations within a reasonable period of time, even for moderately-sized problems. So far, the sizes of the problems solved by this technique have been small, generally less than 12 jobs [48,87].

The combinatorial search technique was introduced by Dudek and Teuton [40] in 1964. This technique functions by employing some form of dominance check to eliminate less desirable partial sequences and then examining the remaining small subset for the best partial sequence. Thus, the elimination procedure reduces the search effort and leads to optimal solutions. Since the original Dudek and Teuton article, Smith and Dudek [100], McMahon [74], Bagga and Chakravarti [12], Baker [15], Szwarc [105,106], and Gupta [53,55] have presented other methods. As the number of jobs, $\underline{n}$, increases, the combinatorial method demands a highly disproportionate growth in the computational effort, a fact which has led to the conclusion that its application is not beneficial to problems containing more than nine jobs [55].

The foregoing techniques were effective in obtaining solutions to the flowshop scheduling problem with a lesser computational effort than if all possible schedules were enumerated before selecting the best schedule according to some criterion. As noted earlier, these techniques still required considerable effort as the number of jobs increased beyond ten. As a result, some researchers then turned their attention to heuristic techniques which, by definition, yield "good" or "near optimal" solutions. Using these techniques to obtain a solution saves computational effort. However, the saving is accompanied by a decline in the quality of the solution because the solutions are generally not optimal. The heuristic approach seems to hold the most promise for an economical but efficient solution to the scheduling problem. As a result of the recent emphasis on this solution technique, many heuristic algorithms have appeared in the literature [1,36,37,44,50,54,58,72,84,85]. These algorithms can be placed in two broad categories: those which yield a single solution and those which yield multiple solutions, from which the best is chosen. Chapter 3 will discuss algorithms from both categories in some detail.

A practitioner faced with the problem of scheduling a number of jobs on several machines can therefore solve his problem by selecting an algorithm from among the different classes of algorithms. If he prefers an optimal solution, then he can determine the schedule by such methods as complete enumeration, combinatorial search, integer programming, and branch-and-bound with backtracking. If a less than optimal solution would be acceptable, then a heuristic algorithm or branch-and-bound without backtracking would yield the required solution. In order to select the particular algorithm, the practitioner may need to know which algorithm provides a solution with the least computational effort and what quality of solutions the algorithm provides. Not much research has been done in this area. In the case of the optimizing algorithms, Ashour [8] and Baker [14] have compared a number of algorithms, but their selection of algorithms was by no means exhaustive. Dannenbring [37] has been the only researcher to have compared several heuristic algorithms in the same study. Likewise, his selection was not exhaustive, and many new algorithms have appeared since his selection was made. In addition, when most researchers reported a new algorithm, there was no uniformity in their choices of algorithms with which they compared it [1,7,50,54,58]. Unfortunately, the testing of these new algorithms has been inconsistent, because each new algorithm was tested using a different problem set (or tasks) and on different computers. It is therefore difficult to make concrete statements about the comparative abilities of heuristic algorithms which were not tested in the same study. On the other hand, the research by Ashour, Baker and Dannenbring enables one to make concrete conclusions, because the algorithms included in each study were tested under the same conditions.

Other studies have addressed different issues. Gupta [45,49,57] has used criteria other than makespan in studying the flowshop problem. He has argued that, if economic optimality is desired, makespan may not necessarily be the most appropriate measure of performance. Mellor listed several types of goals which a production department might try to optimize [77]. He also advocated combining the performance measures in order to more adequately reflect these goals.

In a recent study, Borovits and Ein-Dor [23] introduced Kiviat Charts as a means of combining multidimensional criteria to evaluate results derived from employing four scheduling rules to solve jobshop problems. The charts create a visual display of certain types of results and can best be described in the words of the authors, Borovits and Ein-Dor [23]:

> These charts relate various aspects of system performance allowing one to see at a glance whether, in general, the system under consideration is well utilized or not. We believe that this same technique may be of value in evaluating job-shop schedules.

The charts seem to require that the results be expressed in relationship to ideal results and that visual judgements be made in comparing results. The use of monetary transformations of criteria was also done in this study as well as in some studies by Gupta [45,49,57]. Intuitively, this appears to be a very logical approach but, unfortunately, cost data are seldom available.

Justification for This Study

As mentioned in the previous section, much of the research on flowshop scheduling has focused on creating algorithms and on comparing the results of algorithms. The recent trend has been to concentrate on heuristic algorithms. To date, the most comprehensive comparison of heuristic algorithms was done by Dannenbring [37]. His study, however, did not include all of the heuristic algorithms available, and since then, more algorithms that have produced good results have appeared [1,102].

The formulation of the flowshop problem and the development of these algorithms have been done after a number of assumptions and simplifications were introduced. These assumptions and the formulation of the problem will be the topic of Chapter 2. One simplification which has frequently been adopted is the use of a single criterion of performance. In the majority of the studies, including Dannenbring's comprehensive study, the minimization of makespan was used as the criterion of performance. Since a production system would be likely to have multiple goals, the results of the above studies would be of limited use to the managers of such systems.

The Kiviat Chart approach of Borovits and Ein-Dor [23] did not include the minimization of makespan as a criterion of performance, but it produced a technique for combining multidimensional criteria. It, however, requires visual evaluation, which is rather imprecise.

There has been a notable absence of rigorous experimental designs in past studies. Results were generally obtained by simulating the performance of the schedules that the algorithms produced, with the simulations being replicated a number of times. The results were then averaged over the number of replications and comparisons were made with these averaged figures. No attempts were made to attach any statistical significance to the differences between the results for the various heuristic algorithms.

In view of the above statements, it seems reasonable that a study which, using a multivariate approach, compares some of the better heuristic algorithms, permits multiple performance criteria to be used, and permits one to make statements about the statistical significance of the differences in the results would indeed be a useful contribution to the literature in this field of knowledge. This study seeks to develop an analytical technique which will integrate the multiple criteria of performance. A multivariate analysis of variance model is used to permit these criteria to be considered simultaneously and, at the same time, any interaction effects between these criteria can be investigated and taken into account. This technique, therefore, will permit any comparison of algorithms to provide more adequately the information needed by managers. Additionally, statistical statements can be made about the results because of the utilization of an experimental design and a multivariate statistical method.

In this study, a number of heuristic algorithms that gave promising results in former studies are compared. Two of the algorithms were selected from Dannenbring's study since they consistently gave good results, while two others were selected from recent articles on the basis of their promising results. The fifth algorithm was developed by the author for this study. Thus, as a secondary benefit, this study compares the five algorithms using three measures of performance:

1. Makespan
2. Machine Idle Time
3. Job Waiting Time

To the best of this author's knowledge, these algorithms have never undergone such a comparison. Although this is not an exhaustive study of all of the algorithms available, it does compare some of the newer algorithms with the older ones and brings together some which have never been compared.

Dannenbring's research [37] suggested that the size of the problem, which is measured by the combination of the number of jobs and number of machines, had an effect upon the efficiency of the algorithms. This study investigates this effect with the algorithms under consideration. Results from this portion of the study may provide knowledge which would help in selecting the appropriate algorithm for any given task.

The justification for this study can therefore be summarized as follows:

1. It seeks to develop a multivariate model flexible enough to integrate the multiple goals of a production system in order to compare flowshop heuristic algorithms. This comparison would permit the selection of the most appropriate algorithm for scheduling jobs through the system.
2. Using three criteria of performance, it compares five algorithms. Some of these algorithms have never been compared, and those which have been were compared on the basis of a single criterion.
3. It investigates the effect of problem size on the efficiency of the algorithms. This will assist in the selection of the most appropriate algorithm for the task.
4. It investigates the interaction effect between the algorithms and the problem size.
5. It adopts an approach which utilizes a formal experimental design and incorporates statistical techniques in the comparison of the data. Post-analyses are also performed.

Statement of the Problem

With the exception of one study, past research articles have compared the performance of scheduling algorithms on the basis of a single criterion. It has often been mentioned that in order for the scheduling theory to be of use to the practitioner, an attempt should be made to approach the problem realistically [41,48]. Realism has been lacking in comparing the performances of the heuristic algorithms. Most studies have used techniques which require a single measure of performance. This has been convenient but far from realistic because firms are more likely to schedule jobs in order to achieve multiple goals. This study attempts to introduce a more realistic approach to comparing the performance of heuristic algorithms; it develops a multivariate model which would permit evaluations using multiple criteria. It also investigates the effect of the size of the problems on the efficiency of the algorithms. In addition, this study investigates and takes into consideration any interaction between the heuristic and problem sizes. The computational times required by the algorithms to solve particular problems are also compared.

The research is carried out entirely in the laboratory setting. Each algorithm is used to generate a schedule, and the operation of the flowshop, utilizing the schedule, is simulated on the computer. The performance values are recorded and evaluated by a multivariate statistical technique which was selected to compare the algorithms' ability to generate good schedules. The multivariate statistical technique is fully described in Chapter 4, where the full experimental design is presented. Univariate statistical techniques are also used where appropriate, as in the comparison of the heuristic algorithms using a single measure of performance.

The research described here attempts to answer the following questions:

1. Which of the heuristic algorithms being studied produce the most efficient schedules? Efficiency will be measured by the following three criteria of performance:
   a) Minimization of machine idle time
   b) Minimization of job waiting time
   c) Minimization of the length of time between the start of the first job and the finish of the last job.
2. Given the above criteria of performance, which algorithms are most likely to produce optimal schedules?
3. Given a fixed problem size, as measured by the number of jobs and the number of machines, which heuristic algorithm requires the least computational effort, determined by the amount of time required for the computer's central processing unit to perform the calculations?
4. Does the size of the problem affect the efficiency of the heuristic algorithms?
5. Is there an interaction effect between the heuristic algorithms and the problem sizes?

Complete answers to all of the above questions may be elusive, but this study makes diligent efforts to explore them.

Limitations of This Study

No attempt is made to compare all of the algorithms reported in previous studies. Instead, a limited number of the algorithms are selected and compared.

The study is limited to a selected number of problem sizes. It is necessary to limit the number of algorithms and problem sizes because of both the expense involved and the limited computational capabilities of the electronic data processing equipment.

The Organization of The Remainder of This Study

The flowshop scheduling problem is presented and examined in Chapter 2. Here, the problem is formally developed and the assumptions utilized in its formulation are stated and discussed. Chapter 3 consists of a comprehensive survey of the literature relevant to this research. The experimental design and the research analytical tools are presented in Chapter 4. The data are analyzed and the results are discussed in Chapter 5. Chapter 6 consists of both a summary of the results of this investigation and the conclusions derived from the analysis.

CHAPTER 2

THE FLOWSHOP SCHEDULING PROBLEM

This chapter defines and formulates the flowshop problem. The assumptions employed in order to formulate the problem are also specified and discussed. Throughout this chapter, an attempt is made to keep the theory and the discussion as general as possible by using abstract terms. Thus, when the term "job" is mentioned, it refers to an entity on which a number of related tasks are to be performed. An example of a job could be the engine of an airplane, a patient undergoing tests, or the processing of an insurance claim. Likewise, the term "machine" refers to a work station or facility at which the processing is done on the jobs. Such a work station could be a lathe, an x-ray room, or a clerk's desk.

Definition of the Problem

In describing the flowshop problem, it would be useful to think of some hypothetical production center where there are m machines available. These machines are arranged in a particular order and are numbered sequentially. They are so ordered because of the technological requirement that all jobs must undergo initial processing by the first machine, then by the second machine, and so on. The flowshop scheduling problem can therefore be defined as follows [54, p. 12]:

> "Given n jobs to be processed by m machines; the process time of job a on machine m being T_{am} (a=1,2, . . .,n; m=1,2, . . .,M), it is desired to find the order (schedule) in which these n jobs should be processed on the m machines so as to minimize a well defined measure of production cost."

The Assumptions of the Flowshop Problem

On reviewing the literature on the flowshop problem, it was found that a number of assumptions were utilized in order to formulate and solve the problems. The assumptions fell into four categories and can be stated as follows [98, p. 3]:

1. Assumptions regarding the jobs:
   a) All n jobs are available before processing begins.
   b) Each job is to be processed in the same technological order.
   c) Jobs are processed as soon as possible.
   d) Jobs are to finish in the same order in which they started. No passing is to be allowed.

e) A single job cannot be processed simultaneously by more than one machine.

2. Assumptions regarding the machines:

   a) Only one machine of each type is available.

   b) At the most, only one job at a time can be processed on a specific machine.

3. Assumptions regarding processing times:

   a) The processing times of each job are known and are deterministic.

   b) The processing times are independent of sequence.

   c) [The] set-up and transportation time is included in the processing time.

4. Other assumptions:

   a) In-process inventory is allowed.

   b) _n_ and _m_ (the number of jobs and machines, respectively) must be positive finite integers.

An additional assumption, which is quite often employed but was not stated above, requires that, once an operation is started by a machine on a job, it should be continued without interruption to completion. As stated earlier, these assumptions have made possible the formulation and solution of the flowshop scheduling problem, but at the same time there has been a disadvantage in using some of them; they have removed some degree of realism from the problem. For example, consider the first assumption, which states that all of the jobs are available before processing begins. This means that any of the _n_ jobs could be chosen to undergo processing first. Likewise, any of the remaining (_n_-1) jobs could be chosen for the second position, and so on until all of the jobs are sequenced. This assumption makes the flowshop problem static, which is generally only true in the very short run. In an actual situation, jobs would be more likely to arrive in a random dynamic manner than in a batch.

Another assumption concerning the jobs requires a job to be processed as soon as a machine is available or as soon as the job arrives at that particular machine. In practice, this may not necessarily be most economical. It may be possible to delay the processing of a job without delaying the completion of the entire batch. Delaying the processing until there are other jobs to be processed on the same machine may result in a reduction of the operating cost of the machine. For example, if one step in the processing of all the jobs requires heating in a furnace, it might be more economical to delay firing the furnace until enough jobs are available so that they can be processed sequentially, without any time delay between jobs, providing that such a procedure does not increase the overall completion time of the entire batch.

The assumption which restricts jobs to finishing in the same order in which they started prevents the possibility of jobs passing each other. It ensures that the sequence of jobs on each machine is the same, which means that the number of possible schedules are limited to $\underline{n}!$. Without this restriction, the number of possible schedules would have been $(\underline{n}!)^m$, a much larger number.

The stipulation that a single job cannot be processed simultaneously by more than one machine is somewhat unrealistic for most jobs. In actual practice, some jobs can often be segmented into portions that can be processed separately, or more than one machine can be utilized on the job simultaneously. For example, an engine being repaired can be segmented so that the compression system can be maintained at the same time that the ignition system is being reconditioned.

A similar assumption restricts a machine to processing a single job at any one time. While this restriction may be true for a number of machines, like an x-ray machine, it is not necessarily universally true. The modern digital computer is an example of a machine which processes multiple jobs simultaneously. It does not wait until processing is completed on one job before starting the next one. On the contrary, it utilizes the technique called multiprogramming, which permits the machine to interrupt processing of a particular job in order to commence processing on another job.

An additional assumption states that there is only one machine of each type available. This and the previous assumptions serve to limit the complexity of the mathematics that would be needed in formulating and solving the problem.

There are a number of assumptions associated with the processing times. For the sake of convenience, the set-up time incurred in making preparations to process the job is included in the processing time of the job. The time taken to transport the job from the preceding machine is also included in the processing time.

Another assumption states that the processing times are independent of the sequence in which jobs are processed. This assumption is somewhat ludicrous when the preceding assumption is considered. Certainly, it would not be unusual to find that the time taken to dismantle the machine after processing job B and then to set it up in order to process job A would indeed differ from the time taken for these operations if job A followed job C. Thus, the processing time, which incorporates the set-up time, would indeed be different because the set-up times differ with the sequence. This argument also exposes some lack of realism in the third assumption, which states that the processing times of each job are known and are deterministic. This assumption obviously ignores the effect of the sequence.

There are some uncategorized assumptions. First, in-process inventory is allowed. This means that a job can wait in a queue until the machine becomes available to process it. Without this assumption, a schedule would have to be developed taking an additional restriction, no queuing, into consideration. Another uncategorized assumption is the requirement that a machine cannot interrupt processing one job in order to work on another job. This is another unrealistic assumption. Digital computers, operating in a multiprogramming mode, usually process several jobs at the same time by interrupting their processing of one job to process another.

The ideal solution to the flowshop problem would be one developed without the aid of many of these assumptions, especially those which are unrealistic. Unfortunately, models have not yet been developed to represent those situations which exist before simplification by the assumptions. Attempts to relax some of these assumptions have been few and rare. In keeping with the past practice, the flowshop problem will be formulated in the following section with the aid of the assumptions.

The Formulation of the Flowshop Scheduling Problem

When a flowshop heuristic algorithm is applied to the scheduling problem, the expected outcome is a solution which minimizes some measure of performance. This minimization of a performance measure is generally considered as a surrogate for the minimization of production costs. In formulating the problem, three measures of performance are considered. These are the makespan of the schedule, the machine idle time and the job waiting time. As defined earlier, makespan is the length of time which elapses between the start of processing and the completion of processing on the batch of jobs. The machine idle time is a measure of the time a machine is not utilized between the completion of the processing of a job and the arrival of the next job. On the other hand, job waiting time measures the length of time a job waits in a queue for processing to begin on a new machine after processing has been completed on the previous machine. The following notation will be used in the formulation of the problem:

T_{ij} = processing time of job $\underline{i}$ on machine $\underline{j}$

$C(i,j)$ = the completion time of job $\underline{i}$ on machine $\underline{j}$

I_{im} = the length of time machine $\underline{m}$ spends idle waiting for job $\underline{i}$

W_{im} = the length of time job $\underline{i}$ spends in the queue waiting to be processed by machine $\underline{m}$.

Consider a batch of $\underline{n}$ jobs to be processed on $\underline{m}$ machines. If $\underline{s}$ denotes a subset of these jobs formed into a partial sequence of length $\underline{k}$, then this partial sequence can be augmented by a single job $\underline{i}$, which was not included in the partial sequence, to form a new partial sequence, $\underline{si}$, of length ($\underline{k}$+1).

By definition, the start of processing is time zero. The completion time of the partial sequence, $\underline{si}$, on machine $\underline{j}$

can then be determined by the recursive relationship [40,54]:

(2.1) $C(si,j)=\max[C(s,j);C(si,j-1)]+T_{ij}$

where $C(s,0)=C(\theta,j)=0$ for all s and j

if θ is a null sequence.

The general expression above can then be used to calculate the measures of performance as follows:

Makespan

If the last machine can be denoted by $\underline{M}$ and the last job in the sequence by $\underline{N}$, then the makespan can be represented as follows:

(2.2) $C(sN,M)=\max[C(s,M);\ C(sN,M-1)]+T_{NM}$

where $\underline{s}$ is a partial sequence of length $\underline{k}$

and $\underline{k}$ equals $\underline{N}$-1.

The flowshop scheduling problem that uses makespan as its measure of performance seeks to find that schedule which minimizes $C(sN,M)$.

Machine Idle Time

There are times when a machine $\underline{m}$ is idle because there is no job waiting to be processed. The length of the idle time, I_{im}, can be calculated as the difference between the time at which processing is completed on the last job, $\underline{i}$-1, and the time of the arrival of the next job, $\underline{i}$. Since the transportation time of the job $\underline{i}$ is included in the processing time, T_{im}, then the arrival time at a machine can be considered to be the same as the time at which processing on that job was completed on the previous machine, m-1. The machine idle time for machine m while waiting on job i can be calculated as follows:

(2.3) $I_{im}=C(si,m-1)-C(s,m)$

When this is summed over all n jobs, the total machine idle time for machine m is obtained. The sum for the entire schedule is determined by summing the idle times for all of the machines.

Job Waiting Time

When job i waits in a queue at machine m to be processed, the job waiting time, W_{im}, is measured by the difference in time between the finish of the preceding job, i-1, on machine m and the finish of job i on machine m-1. This is expressed as follows:

(2.4) $W_{im}=C(s,m)-C(si,m-1)$

Totaling this value over all jobs on all machines produces the total waiting time for the entire schedule.

The above expressions, (2.2), (2.3) and (2.4), are used to calculate the three measures of performance which are used to evaluate the heuristic algorithms. These algorithms are discussed in the next chapter.

CHAPTER 3

LITERATURE REVIEW

In the first chapter, the historical developments of solving the flowshop problems were traced, beginning with the optimizing techniques and ending with the heuristic methods. This chapter will review the literature available on heuristic flowshop algorithms. Although Johnson's algorithm [65] is not a heuristic technique, this chapter will use it as the starting point because it was one of the first reported efforts to solve the flowshop problem. In addition, a number of algorithms [27,37] have incorporated modified versions of the Johnson algorithm into their solution procedures. The heuristic algorithms reviewed here have been taken both from studies of the individual algorithm and from comparative studies.

Johnson's Algorithm

Perhaps the single most influential article on flowshop scheduling was written by Johnson [65]. In it, he described an optimizing technique which was later called

"Johnson's algorithm." The article was among the first to focus on the flowshop problem, and it was also among the first to use makespan as a measure of performance. Since then, many authors have adopted this measure of performance.

The article described an algorithm which yields optimal solutions for problems with two machines and any number of jobs. The processing times of each job on each machine are utilized to arrive at solutions. Let T_{ij} represent the processing time of job i on machine j (note j = 1 or 2). The jobs are then scheduled by the following rule: job a precedes job b when

$$\min \{T_{a1}, T_{b2}\} \leq \min \{T_{a2}, T_{b1}\}.$$

In actual practice, the processing times for all jobs on all machines are examined and the job with the smallest time selected. If the smallest time is on machine 1, the job is placed in the earliest available position; if the time occurs on machine 2, it is placed in the latest available position.

Two of the main advantages of this solution technique are its simplicity and its guarantee of an optimal solution. The solution can be reached manually. Johnson provided theoretical proof for the technique and extended it to special cases of the three machine problem.

Heuristic Algorithms

Palmer [85] developed the Slope Index Technique, one of the first heuristic algorithms for the flowshop problem.

According to this method, a slope index is calculated for each job, and then these indices are ranked in a decreasing order. The ranked list of jobs is used as the job schedule for the flowshop. Palmer states that the basis of his algorithm is assigning priority to jobs which have "the strongest tendency to progress from short times to long times in sequence of processes [85, p. 102]." The slope index for each job i is calculated with the following functional expression:

(3.1) $S_i = 1/2[-(m-1)T_{i1} - (m-3)T_{i2} . . . + (m-3)T_{i,m-1} + (m-1)T_{im}]$

where m is the number of machines or processes

and T_{ij} is the processing time for job i on machine j.

This slope index method requires only a moderate computational effort and could easily be executed manually. Palmer tested his algorithm on 7 small problems, ranging in size from 4 jobs by 3 machines to 9 jobs by 3 machines. The processing times for these problems were the same as those used by Storey and Wagner [103] who solved their problems using an integer programming method. Palmer compared the solutions obtained by his slope index method with the optimum solution obtained by using makespan as the measure of performance. His methods yielded solutions which had makespan values ranging from 0% to 8.6% higher than the optimal solutions. In one case, the slope index method actually produced an optimal solution. Palmer also reported his results in terms of the number of job interchanges in the schedule in order to arrive at the optimal schedule. These ranged from 0 to 10. Furthermore, he investigated the schedules yielded by ordering the slope indices in the reverse pattern, arranging them in an increasing order. The results these schedules yielded were very much inferior to the previous results.

Page [84] developed three flowshop heuristic algorithms which were designed on the assumption that there is some analogy between the scheduling problem and the sorting problem in data processing. Thus, each algorithm is based on one of the following sorting techniques: merging, pairing, and exchanging. In applying the latter technique, exchanging, either individual jobs or groups of jobs are exchanged. Page used makespan as the measure of performance when testing the three algorithms he developed. The sizes of the problems he solved with his algorithms varied from 4 jobs by 3 machines to 32 jobs by 8 machines.

An examination of the results obtained from the solution of these problems indicated that the algorithm based on merging produced the best results. Makespan values for this algorithm exceed the lower bound values for the same problems by percentage deviations varying from 2% to 20%. The algorithm based on pairing yielded the worst results with percentage deviations varying from 4.4% to 28.5%. On the other hand, this algorithm consumed the least computer time in performing the computations necessary to arrive at a schedule. Merging and exchanging required approximately the same amount of time. Hence, the merging algorithm is preferable to the exchange algorithm because its percentage deviation results are superior. The choice between the merging and pairing algorithms depends upon the manager's preference for either accurate results or the least computational effort.

Ashour introduced the decomposition method of job scheduling in 1967 [6] and reported a modified version of this technique in 1970 [7]. In brief, this method partitions the batch of jobs into subgroups, generally two, and the jobs in each group are sequenced by an optimizing technique. The two sequenced groups are then recombined to form a single schedule. The division into two groups reduces the amount of computational effort. For example, determining the optimal sequence for a single set of 10 jobs by the complete enumeration technique would entail the examination of 10! or 3.6 million different sequences or permutations of the jobs. On the other hand, when the 10 jobs are divided into groups, the number of sequences to be examined is reduced to 10!/(5!5!) or 252 different sequences.

In tests of this technique, the number of jobs solved varied from 6 to 40 while the number of machines varied from 3 to 10. Problems with 6 or less jobs were also subjected to complete enumeration of all sequences in order to obtain optimal solutions, while those with 7 or more jobs were subjected to partial enumeration.

Ashour discovered that as the number of jobs in each subgroup increases, the number of potential schedules decreases. He cited the case of 6 jobs in a single group that yielded 720 different schedules, whereas the same jobs, divided into 2 groups of 3 jobs each, produced 20 arrangements. Further subdivision into 3 groups of 2 jobs each increased the number of arrangements to 90. He also found that as the number of jobs in each subgroup increases, the range of schedule makespan values decreases, the maximum schedule makespan value decreases, and the relative frequency of the shortest schedule time increases. Thus, when the number of jobs in each subgroup increases, better schedules result. In addition, he found that the computational effort required to obtain a solution decreases with an increase in the size of the subgroups. It therefore appears that partitioning the jobs into two subgroups would tend to yield better results and, at the same time, require minimal computational effort.

Ashour compared the results obtained from solving 18 problems when the jobs were divided into two subgroups with solutions obtained by partial enumeration. He found that the decomposition approach yielded the better results. Ashour also claimed that this technique was superior to the rounded linear programming method of Giglio and Wagner [42].

Perhaps the most attractive feature of this technique is the subdivision of a large number of jobs into smaller groups, which can then, using much less computational effort, be sequenced by optimizing techniques. Gupta and Maykut [58] took this technique one step further and arrived at the heuristic decomposition technique.

The heuristic decomposition technique, introduced in 1973, is based on a modification of Ashour's decomposition approach and the job-pairing algorithm. The jobs are partitioned into two subgroups. One subgroup is scheduled by a technique which produces optimal schedules, while the remaining subgroup is sequenced by a heuristic job-pairing technique. The subgroups may contain an unequal number of jobs. The choice of the size of the subgroup to be scheduled by the optimizing technique depends upon the amount of computational effort which can be afforded in obtaining a solution. For a large number of jobs, it is likely that the size of this subgroup would be smaller than the subgroup sequenced by the job-pairing algorithm, because most optimizing techniques require considerable computational effort for subgroup sizes of 10 or greater.

The job-pairing algorithm develops a schedule by adding jobs, one at a time, to a partially-formed schedule. The job selected for addition to the partial schedule is the one which has the greatest potential to minimize the idle time of the last machine in the process. The job-pairing procedure yields a partial schedule that is considered a synthetic job by the heuristic algorithm. In the overall schedule of all jobs, this synthetic job precedes the partial schedule determined by the optimizing technique. Gupta and Maykut determined this placement of the synthetic job because the job-pairing algorithm operates by attempting to minimize the machine idle and processing time for the first pair of jobs on the last machine. The minimization of idle time on the last machine automatically results in the minimization of makespan [45].

The heuristic decomposition algorithm was tested on 360 problems, using a Univac 1108 computer and processing times that were randomly generated from a uniform distribution with a range from 0 to 99. The sizes of the problems were small, varying from 6 jobs by 3 machines to 12 jobs by 3 machines. The results were reported in efficiency terms. Efficiency is defined as the ratio of the optimum makespan value to the value obtained from the solution by the algorithm. The results were compared with those reported in Ashour's decomposition study [7]. This is not entirely a direct comparison because Ashour obtained his processing times from a uniform distribution ranging from 1 to 30. Compressing the range in this manner reduces the span of possible error [58].

Over the range of problems considered, the heuristic decomposition technique produced slightly better results when efficiency was used as the measure of performance. It had superior average efficiencies over 4 of the 9 problem sizes, while Ashour's decomposition technique excelled over one problem size, 7 jobs by 4 machines. Both heuristic algorithms had equivalent results over the remaining four problem sizes. It should be noted that all of the efficiencies were good, the worst being .90. The heuristic decomposition method required considerably less computing time in order to arrive at schedules. On the average, it required less than 1/100 of the time required by Ashour's method. For example, the largest problem, 12 jobs by 3 machines, required .0769 and 11.56 seconds for the respective algorithms. The job-pairing algorithm saves a significant amount of time because it needs considerably less time than the optimizing technique to schedule one of the subgroups.

Since the heuristic decomposition technique appears to produce solutions of high quality while consuming very little computational time on the Univac 1108 computer, it has been selected as one of the heuristic algorithms to be treated by the multivariate analysis of variance method in this research study.

Campbell, Dudek and Smith [27] developed an algorithm which has become a standard for comparison as demonstrated by its inclusion in several studies since its initial publication [1,37,54]. In the remainder of this research study, this heuristic algorithm is referred to as the CDS algorithm.

The CDS algorithm converts the single m-machine problem to (m-1) artificial 2-machine problems. This is achieved by calculating new processing times, V_{ij}, from the original processing times, T_{ij}, with the aid of the following equations:

(3.2) $$V^k_{i1} = \sum_{j=1}^{k} T_{ij}$$

(3.3) $$V^k_{i2} = \sum_{j=1}^{k} T_{i,m-j+1}$$

where V_{ij} = the derived processing time for job i on machine j

T_{ij} = the original processing time for job i on machine j

k = the number of the problem (k=1,2, . . ., m-1)

Thus, for each artificial problem, the processing time for job i on machine 1, V^k_{i1}, is determined by summing the original processing times for job i on the first k machines. Likewise, the processing time for job i on the second machine, V^k_{i2}, is determined by summing the processing times for job i on the last k machines. The (m-1) artificial problems are then solved individually with the Johnson algorithm, yielding (m-1) solutions. In the end, the best solution is selected.

Campbell, Dudek and Smith tested their algorithm over a wide range of problems, a total of 340, and compared the makespan results with the solution obtained by Palmer's slope index heuristic algorithm. This comparison was done with problems of both small and large sizes. Processing times were randomly obtained from a rectangular distribution with a range of 1 to 99.

The small problems ranged from 3 jobs by 4 machines to 6 jobs by 6 machines. A sample size of 20 problems was used for each problem size. The optimal makespan solutions for the problems were also determined, and the percentage deviation of the heuristic algorithm solution values were calculated. For all small problem sizes, the deviations for the CDS algorithm were smaller, ranging from .12% to 2.56% with an average of 1.38%. On the other hand, the corresponding figures for Palmer's slope index algorithm were 2.68%, 6.52% and 4.58%.

These heuristic algorithms were also used to solve 10 large problems varying in size from 20 jobs by 20 machines to 60 jobs by 30 machines. It was not feasible to determine optimal solutions, so the results were reported in terms of makespan values and the percentage improvement of the CDS algorithm's results over the other algorithm's results. In all cases but one, the CDS algorithm had lower makespan values with percentages varying from 0.84% to 10.60%. In the single case when the slope index algorithm performed better, the problem was a 20 job by 20 machine problem and the percentage improvement was -.34%.

The CDS algorithm required more computer time to arrive at solutions. This is not surprising because Palmer's slope index method produced a single sequence, and the CDS algorithm produced and evaluated (m-1) sequences. The CDS algorithm is selected for inclusion in the multivariate study described later in this research. It is selected because of the excellent results it produced in all of the studies in which it has appeared.

Gupta [50] designed a functional heuristic algorithm which is somewhat similar to Palmer's slope index algorithm. He described this algorithm as an extension of Page's analogy of scheduling and sorting. It is claimed that computations can be done manually for reasonably large problems. A similar claim is made for the slope index algorithm.

Gupta observed that both Page [84] and Bakshi and Arora [16] showed that the Johnson algorithm [65] can be represented by a functional expression, which is associated with each job independent of the other jobs. When this functional expression is evaluated for each job and the resulting values sorted in ascending order, a schedule is derived by substituting the corresponding jobs for the sorted values. This expression, functionally equivalent to the Johnson algorithm, is limited to the two-machine problem and special cases of the three-machine problem. Thus, Gupta extended this approach to the general m-machine problem by developing a new functional expression:

(3.4) $$f(i) = \frac{A}{\underset{1 \leq m \leq (M-1)}{\text{minimum}} (T_{im} + T_{i,m+1})}$$

where: $A = 1$ if $T_{iM} \leq T_{i1}$

$= -1$ otherwise,

T_{im} = the processing time of job i on machine m,

M = the last machine in the technological order.

Gupta tested his functional heuristic algorithm by comparing it with Palmer's slope index heuristic algorithm. This comparison occurred over 243 problems, ranging in size from 4 jobs by ~~4~~ machines to 100 jobs by 60 machines. Problems with 8 jobs or less were considered small, and optimal solutions were determined for these problems. The processing times used were generated from a rectangular distribution with a range from 0 to 999.

The results for the small problems were reported in terms of the percentage deviations of the algorithm makespan value from the optimal makespan value. The functional heuristic algorithm had better results (smaller percentage deviations) than the slope index heuristic algorithm for 157 of 195 problems solved. The latter algorithm was superior on 27 occasions, and there were 11 ties. The average percentage deviations of the functional heuristic algorithm ranged from 3.4% for smaller 4 jobs by 4 machines problems to 10.6% for the 8 jobs by 8 machines problems. The range for the slope index algorithm varied from 7.6% to 22.0%.

The functional heuristic algorithm maintained its superiority with the large problems. The results of the large problems were reported as a ratio, calculated by dividing the makespan value for the functional heuristic algorithm by the makespan value of the slope index algorithm. The average value of the ratio for each problem size was less than 1, thus confirming that the functional heuristic algorithm had the smaller makespan value. Of the 48 problems solved, the slope index heuristic algorithm had the better solution for only 4 problems.

The times required for computing schedules by these algorithms were very small indeed, and, in absolute terms, differences were practically negligible. For example, on the Univac 1108 computer, the 100 job problems required an average of .1490 and .1670 seconds for the functional and slope index heuristic algorithms respectively.

Although Gupta's functional heuristic algorithm produced better results than Palmer's slope index algorithm over the wide range of problems solved, the average percentage deviations were somewhat high for both algorithms. Unfortunately, a direct comparison cannot be made with other studies because the processing times were taken from a distribution with an unusually large range of 0 to 999.

In another study, Gupta [54] introduced three heuristic algorithms: Minit, Micot and Minimax. The author stated that these were products of "several simplifications and approximations of the combinatorial approach to flowshop scheduling problems [54, p. 13]." All three algorithms were developed with the objective of minimizing the maximum flowtime or makespan values.

The Minit algorithm develops a schedule by adding a single job at a time to a partial schedule. The job is selected from among the unscheduled jobs on the basis of its ability to minimize the idle time generated on the last (M^{th}) machine by the augmented partial schedule. Thus, if σ is a partial schedule and jobs a and b are competing to be added to the partial schedule, job a will be chosen for augmentation if:

$$Q(\sigma a,M) < Q(\sigma b,M)$$

where: $Q(\sigma a,M)$ is the idle time associated with the augmented schedule, σa, on the last machine M.

The initial partial schedule is formed from the pair of jobs that generates the least idle time on the last machine.

The Micot heuristic algorithm is very similar to the Minit heuristic algorithm, but they differ in that the Micot heuristic algorithm seeks to minimize completion times on the last machine. Hence, the augmented partial schedule, σa, is considered more desirable than σb if:

$$C(\sigma a,M) < C(\sigma b,M)$$

where: $C(\sigma a,M)$ is the completion time of the augmented partial schedule, σa , on the last machine, M.

The Minimax heuristic algorithm bears some resemblance to the Johnson algorithm. It determines schedules by examining the processing times for all jobs on all machines and assigns the job that needs the least processing time to the earliest available sequence position. Conversely, the job that needs the greatest processing time can be assigned to the latest available position. Thus, the schedule can be developed from both ends.

Using 1200 problems ranging in size from 4 jobs by 3 machines to 7 jobs by 20 machines, these three algorithms were then compared among themselves and with the Campbell, Dudek and Smith (CDS) heuristic algorithm. The processing times were generated randomly from a rectangular distribution with values ranging from 0 to 99. The quality of the solutions was reported in efficiency terms as the ratio of the value of the measure of performance for the optimal solution to the value for the solution determined by the algorithm. The two measures of performance used were mean flowtime and maximum flowtime (or makespan).

Under the makespan criterion, the results of the three proposed heuristic algorithms were inferior to those of the CDS algorithm. The Minimax heuristic algorithm produced the worst results, with an average ratio for a problem size as low as .85. On the other hand, the Minit heuristic algorithm produced the best results of the proposed algorithms, with average ratios ranging from .90 to .97. The ratios for the CDS algorithm ranged from .95 to .99. As the size of the problems increased, the quality of the solutions obtained by all 4 algorithms decreased.

When the mean flowtime criterion is considered, the performances of the three proposed heuristic algorithms were better than that of the CDS algorithm. The Minit heuristic algorithm had consistently high ratios, generally above .95, closely followed by the Micot heuristic algorithm, whose solution qualities varied from .93 to .98. The results for the Minimax heuristic algorithm varied from .90 to .97, while those for the CDS algorithm fell to as low as .88. The rankings do not reflect any success or failure of the deliberate design embodied in these algorithms, for none of the four algorithms were developed with the minimization of mean flowtime as a goal.

When the computational times required by the Univac 1108 computer to produce a schedule are considered, the CDS algorithm consumed the most time while the Minimix heuristic algorithm utilized the least time. The time required increases with both the number of jobs and the number of machines for all four algorithms. In absolute terms, the time requirements were small and should not influence the choice of algorithms for problems of the small magnitudes considered.

A heuristic algorithm which has recently appeared in the literature was developed by Aggarwal and Stafford [1]. It was designed to minimize the makespan value of the resulting schedule in a two-stage procedure. In the initial stage, a tentative schedule is developed by a heuristic rule that determines the jobs' positions in the sequence according to the values of their processing times and the machine locations. At the next stage, jobs in neighboring positions are temporarily interchanged to determine if this action leads to an improvement in the makespan values. Beneficial interchanges are made permanent. If the changes are not beneficial, the jobs return to the status they had before the interchange.

In regards to developing the initial tentative sequence, the authors stated:

> The logic of this algorithm is based on three considerations: (1) machines located near the beginning of the machine-sequence should process jobs in the increasing order of their processing times; (2) machines located near the end of the machine-sequence should process jobs in the decreasing order of their processing times; and, (3) different job sequences for each machine can arise from the above two conditions [1, p. 238].

In addition, the authors suggested a rule to be used for resolving the conflict between two or more jobs competing for the same position in the sequence. This rule selects the job that forms the partial sequence which has the minimum makespan value.

This heuristic algorithm, which will subsequently be called the AS algorithm, was used to solve 200 small problems and 500 larger problems. The sizes of the small problems ranged from 3 jobs by 3 machines to 8 jobs by 8 machines, while those for the large problems ranged from 10 jobs by 10 machines to 100 jobs by 50 machines. Processing times were randomly generated from a uniform distribution ranging from 1 to 99.

In the case of the small problems, the AS algorithm determined optimal solutions for 55% of the problems attempted. The average percentage deviations of the algorithm's makespan values from the optimum values were calculated; these percentages proved to vary from 0% for the 3 jobs by 5 machines problems to 6.24% for the 8 jobs by 7 machines problems. The overall average was 3.64%. These results were compared with those published in the report of the CDS algorithm [27], which also solved problems of similar sizes and employed randomly selected processing times from a distribution of the same range. On examining the problem sizes the two algorithms had in common, the maximum percentage deviations for both algorithms were the same (5.8%), but the CDS algorithm had the lower percentage deviation on 4 of the 7 problem sizes. However, it was mentioned that the AS algorithm provided nearly .1% to 1% better results for almost 50% of the problems, and that as the problem sizes increase, the AS algorithm provides better results [1]. The average percentage deviations over the seven common problem sizes were 2.7% and 2.1% for the AS and CDS algorithms, respectively.

The AS algorithm was clearly superior in terms of the time required on the computer for the algorithm to determine a schedule. Although processing was done on different computers--an IBM 360/67 for the AS algorithm and an IBM 7040 for the CDS algorithm--the authors converted the CDS algorithm times by taking into account the fact that the former computer is approximately 12.3 times as fast as the latter computer. As a comparison, the AS algorithm consumed 1.8 seconds in solving the 60 jobs by 7 machines problem, whereas the CDS algorithm needed 8.8 converted seconds. The largest problem solved by the AS algorithm was 100 jobs by 50 machines, and that required less than 32 seconds. From the results, it appears that the computational times required by the AS algorithm increases with the number of jobs and the number of machines. The times required by the algorithms were very reasonable, and it is believed that the majority of the time is consumed in that portion of the algorithm which interchanges neighboring jobs to check for better solutions. According to the authors, the maximum improvement resulting from this interchange effort never exceeded 4.2%.

Due to the closeness of its results with the CDS algorithm's performance, its ability to solve large problems, and its very modest time requirements for computations on the computer, the AS algorithm is selected for inclusion in the multivariate research study.

Comprehensive Studies on Flowshop Algorithms

The most complete study comparing heuristic algorithms was done by Dannenbring [37]. It included a number of the heuristic algorithms described earlier in this chapter. These algorithms were the CDS algorithm, Palmer's slope index algorithm, and Page's merging, pairing, individual job exchange, and group job exchange algorithms. Dannenbring also developed three heuristic algorithms, which were included in the study. To complete the group of algorithms studied, he included the random search heuristic algorithm and a linear branch and bound procedure that does not fall in the category of heuristic algorithms. A description of Dannenbring's algorithms appears in the next section.

Dannenbring's Heuristic Algorithms

The three algorithms developed by Dannenbring are related in that the second algorithm is developed from the first, and the third is developed from the second by adding improvement routines. Naming them in order of increasing sophistication with their abbreviated names following in parentheses, they are the rapid access (RA), the rapid access with close order search (RACS), and the rapid access with extensive search (RAES) heuristic algorithms. A description of each algorithm follows.

The rapid access (RA) heuristic algorithm. This algorithm is similar to the CDS algorithm in that an m-machine problem is converted to a pseudo 2-machine problem. The RA algorithm is different because it produces a single 2-machine problem which provides a solution quickly and easily. Also, in obtaining the new processing times for the artificial 2-machine problem, it assigns weights to the original processing times as follows:

(3.5) $$P_{i1} = \sum_{j=1}^{m} (m-j+1)T_{ij}$$

(3.6) $$P_{i2} = \sum_{j=1}^{m} (j)T_{ij}$$

where T_{ij} = the processing time for job i on machine j for the original problem

P_{ij} = the processing time for job i on machine j for the pseudo problem.

The new pseudo problem is solved using the Johnson algorithm.

<u>Rapid access with close order search (RACS).</u> Starting with the solution developed by the RA algorithm, an improvement routine is added which involves a search for a better solution. This routine interchanges adjacent jobs. The resulting schedule obtained from the transposition of a single pair of adjacent jobs is called a neighbor. Thus, if there are n jobs in a sequence, (n-1) neighbors will result from transposing all pairs of adjacent jobs. Each neighbor is examined for improvement in the measure of performance and the best neighbor is selected as the schedule.

<u>Rapid access with extensive search (RAES).</u> Continuing from where the RACS algorithm terminated, the RAES uses the best neighbor to generate new neighbors. A search is made among these neighbors for the neighbor which yields

the greatest improvement. The process is repeated until the newest neighbors fail to provide improved solutions.

Other Algorithms

The random sampling heuristic algorithm. This technique, abbreviated the RS algorithm, develops schedules by randomly selecting jobs for the various positions in the schedule. Its greatest advantage is that it develops a schedule inexpensively.

The linear branch and bound procedure. This technique belongs to the class of implicit enumeration techniques. Its inclusion in the comparison is useful because it provides a direct comparison across classes of solution techniques. It schedules by a branching enumeration approach, whereby the least attractive partial sequences are curtailed by the use of lower bounds. One of the properties which distinguishes between branch and bound techniques is the method of obtaining bounds. The bounding approach used in the procedure was developed by McMahon and Burton [14,75].

Testing Procedures

The heuristic algorithms were tested over 1580 problems, with sizes ranging from 3 jobs by 3 machines to 50 jobs by 50 machines. Processing times were randomly generated from a uniform distribution, with values ranging from 0 to 99.

Many measures of evaluation were used. The measure that has the greatest similarity with other studies is called the relative error (r). It is calculated with the following expression:

(3.7) $$r = 100\,[1-(MS/MS_{st})]$$

where MS = the makespan of the heuristic solution

MS_{st} = the makespan of the evaluation standard.

It was noted that optimal solutions were determined for 1509 test problems and estimates of the optimal solutions for the remaining 71 problems were calculated by procedures described in [36].

Results

The results of the small and large problems were reported separately. Problems with 3, 4, 5 and 6 jobs were considered small while those with 7, 10, 25 and 50 jobs were labeled large. The relative error results of the small algorithms are displayed in Table 3.1. The abbreviated names used in Table 3.1, but not mentioned previously are:

M	Page's merging algorithm
GE	Page's group job exchange algorithm
IE	Page's individual job exchange algorithm
P	Page's pairing algorithm
SI	Palmer's slope index algorithm

TABLE 3.1

RELATIVE ERRORS FOR SMALL PROBLEMS

Heuristic Algorithm	Relative Error (r)
RAES	0.64%
RACS	1.30%
CDS	1.73%
M	1.74%
RS	2.03%
GE	2.72%
RA	3.65%
LBB	3.82%
SI	3.98%
P	4.38%
IE	4.99%
Average for all algorithms	2.82%

Source: Dannenbring [37].

From the relative error results, the RAES algorithm provided the best performance with a relative error of only .64%. This algorithm is one of the three designed by Dannenbring. The relative errors for his other algorithms are 1.30% and 3.65%. It therefore appears that the improvement routines added to the fundamental RA algorithm improved it from a mediocre position to the best. Again, the CDS algorithm proved to be a good performer. The results agreed with a previous study, which found that the CDS algorithm is superior to the slope index algorithm (SI) [27]. They also confirmed that Page's merging algorithm (M) is better than his other algorithms. In Page's study [84], the pairing algorithm (P) had the worst results, but in

Dannenbring's study it is slightly better than the worst algorithm, Page's individual job exchange (IE). It is also noted that the linear branch and bound procedures (LBB) did not fare well.

The results for the large problems are displayed in Table 3.2. Because optimal solutions for 71 of the larger problems were estimated, the figures reported are estimated relative errors.

TABLE 3.2

ESTIMATED RELATIVE ERRORS FOR LARGE PROBLEMS

Heuristic Algorithm	Estimated Relative Errors
RAES	1.58%
RS	3.17%
CDS	4.11%
RACS	4.26%
M	4.68%
GE	5.68%
SI	6.18%
RA	6.61%
LBB	8.19%
P	8.40%
IE	9.19%

Source: Dannenbring [37].

An examination of Table 3.2 reveals that the relative errors for the large problems are approximately twice the magnitude of those for the small problems. This is true probably because the samples remained the same or decreased as the problem sizes increased. When the latter occurs, the diversity of possible solutions increases. If a relatively small sample is taken, then the mean values of the sample could likely be distorted from the optimum values by outlying solutions.

It is also noteworthy that many of the heuristic algorithms are in approximately the same positions in the rankings. Most notably, the RAES algorithm, the CDS algorithm, the P algorithm and the IE algorithm have retained their former positions. There are few or no changes in the rankings of the other algorithms, with the exception that the RS algorithm has virtually exchanged positions with the RACS algorithm. This suggests that for large problem sizes, the RACS' single pass improvement routine is insufficient to obtain an excellent solution. Multiple passes are needed. Their presence in the RAES algorithm is responsible for the improvement in performance beyond that displayed by the RACS algorithm. It is difficult to explain the sharp improvement in the ranking of the RS algorithm. Dannenbring [37] suggests that it may be due to the manner in which the sample size parameter was selected.

Based on the results for both large and small problems, it appears that those algorithms that contain improvement routines (RAES, RACS) or those that produce multiple solutions from which the best is selected (CDS, RS) did better than those algorithms that produced a single solution (RA, SI, P and LBB). An exception is the merging algorithm (M) which produces a single solution. It performed better than the GE and IE algorithms, which produced multiple solutions.

Computations for the algorithms were done on an IBM 360/91 computer and the computation times were reported. The times averaged over all problems are displayed in Table 3.3.

TABLE 3.3

COMPUTATION TIMES FOR ALL PROBLEMS

Heuristic Algorithm	Average Time(sec)
SI	.008
RA	.020
P	.048
M	.073
RACS	.097
CDS	.162
GE	.306
IE	.453
RAES	2.130
RS	9.778
LBB	15.293

Source: Dannenbring [37].

With the exception of the LBB algorithm, those algorithms which produced a single schedule consumed the least time per problem. The RAES algorithm, the best algorithm tested, required considerably more time than all of the algorithms except the RS and LBB algorithms. The considerable increase is due to the use of the multiple pass improvement routine.

From this study, it appears that the RAES and CDS algorithms are the best performers in terms of solution quality and consistency. The RAES algorithm is therefore selected for inclusion in the multivariate study on this basis. The CDS algorithm has already been selected.

Other Comparative Studies

There were two other comparative studies of flowshop scheduling techniques. Unlike Dannenbring's study, they did not emphasize heuristic algorithms. One of the studies authored by Baker [14], did not include a single heuristic algorithm.

Ashour's comparative evaluation. This was the first study to compare flowshop algorithms that are varied in their fundamental approaches to obtaining a solution. With the exception of two algorithms, it can be said that the algorithms belong to different classes of solution techniques. The algorithms compared fall in the classes of switch and check (or combinatorial search) [40,100], branch-and-bound with and without backtracking [11,73], rounded linear programming [42] and modified decomposition [6,7]. The latter technique is a heuristic algorithm and was described earlier in this chapter. Ashour also classified the techniques into two categories: optimal-producing and suboptimal-producing. These names describe the type of solutions yielded by the algorithms. The heuristic algorithm is suboptimal-producing and was joined in this category by the branch-and-bound without backtracking.

The algorithms were tested over 550 problems, ranging in size from 6 jobs by 3 machines to 12 jobs by 3 machines. It should be noted that the number of machines varied from 3 to 5. The processing times were randomly generated from a uniform distribution with a range of 1 to 30. Computations were done on an IBM 360/50 computer. The results were reported in terms of efficiencies. The quotient of the makespan value of the optimal solution and the value for the algorithm's solution is defined as the efficiency. The results revealed that both the modified decomposition and the branch-and-bound without backtracking algorithms produced solutions with similar efficiencies. Expressing the number of optimal solutions produced as a percentage of the number of problems attempted, the percentages for the algorithms were 43% and 45%, respectively. The modified decomposition algorithm, however, required about 5 times as much computer time to arrive at a solution. This was actually less than the time required by the optimal-producing algorithms.

It would not be appropriate to compare the results in this study with the results in Dannenbring's study because the linear branch-and-bound algorithms in each study used different bounding procedures.

Baker's study. This comparative study did not include any heuristic algorithms. It concentrated on controlled-enumeration techniques for determining minimum makespan values. These consist of branch-and-bound and elimination methods. The branch-and-bound method views a schedule development as a tree-like process. It uses lower bounds to select a branch to pursue, and at the same time, curtails the enumeration of other partial sequences. Since there are several ways of arriving at a lower bound, Baker compared the results derived from the various bounding procedures.

In the case of the elimination techniques, a schedule is developed by adding a job at a time to a partial schedule. At all stages of development of this schedule, certain dominant subsets of jobs can be selected from all possible subsets by applying rules that eliminate some of the subsets. Only the dominant subsets are enumerated, therefore this elimination technique reduces computational effort.

The results indicate that the branch-and-bound techniques are more efficient than the elimination method. It was found that a composite of two bounds produced better schedules than a single bound. The study also confirmed the extreme inefficiency of the elimination strategy in solving problems of more than 9 jobs, as reported in [55].

Measures of Performance

Most of the research mentioned in this chapter used makespan as the measure of performance. A number of authors [13,37,41,44,45,77] have suggested investigating other criteria of performance. On the other hand, Manne [48,76] defended the use of makespan by arguing that it is likely to be correlated with dollar costs. In other words, he visualizes makespan as a surrogate for production costs. Gupta [48] disputed this suggestion and, with the help of an example, showed that Manne's argument is not always correct. In addition, he states that:

> The use of makespan as a criterion of optimality considers the effect of idle times on the last machine only. This implies that the makespan criterion does not consider the importance of intermediate machines [45, p. 298].

Thus, if the last machine happens to be inexpensive to own and operate, while the intermediate machine is costly, the makespan criterion would not appear to be appropriate because it minimizes the idle time on the inexpensive machine rather than on the more costly intermediate machine. Therefore, justifying the use of makespan as a surrogate for production costs depends upon the strength of the relationships between them.

Elmaghraby [41] commented that the use of a single criterion of performance in the selection of the best schedule is a simplification of managerial practices in operating production systems. A list of possible system goals which managers might seek to satisfy by good sequencing of jobs was enumerated by both Elmaghraby [41] and Mellor [77], who credited Beenhaker for its origin. Included in these lists are the following criteria:

<u>Relative to the facility (including men)</u>

The minimization of idle facility investment
The minimization of facility set-up costs
The day-to-day stability of the work force

Minimum materials handling costs
Maximum utilization of manpower
Maximum facility utilization
General flexibility
Reserve capacity for rush orders

<u>Relative to the product</u>
Minimum in-process inventory
Adherence to promised shipping dates
Maximum output (production rate)
Minimum raw material inventories
Minimum finished product inventories
Minimum investment inventories
Shortest makespan for certain products
Minimum obsolescence and deterioration of products

The above is not an exhaustive list, but it represents many of the factors that occupy the interest of management. Mellor [77] added that a sequencing system with the ability to combine these factors operationally is yet to be devised.

In the research field, Gupta and Dudek [57] attempted to combine a few of the factors by a technique which utilizes opportunity costs. In this process, operating cost, job waiting cost, machine idle cost, and the penalty cost of late jobs are calculated and summed to give the total opportunity cost of the schedule. The criterion of performance in this case is the minimization of total opportunity cost. Thus, the total opportunity cost can be expressed as follows:

$$(3.8) \quad TC(S) = \sum_{i=1}^{n} [\sum_{m=1}^{M} (h_{im}s_{kim}+y_{im}w_{im}+X_{im}r_m)+p_i d_i]$$

where h_{im} = setup cost per unit time for job i at machine m
s_{kim} = setup time of job i at machine m if job k precedes job i in schedule S
y_{im} = waiting time for job i at machine m

before processing starts

w_{im} = waiting cost per unit time for job i at machine m

X_{im} = idle time of machine m before job i arrives for processing

r_m = rate of return per unit time for machine m

p_i = penalty cost per unit time for job i

d_i = lateness of job i.

In the above expression, there are a number of cost coefficients. In his research, Gupta randomly generated the values of these cost coefficients. Thus, the opportunity costs values are highly dependent upon the generated cost coefficients.

It was also indicated that by selecting certain values for the parameter in the total opportunity cost equation, an expression which represents the cost associated with makespan can be obtained.

Gupta and Dudek specified eight criteria of performance. These generally consisted of the component costs of the total opportunity cost equation, combinations of these components, and makespan.

A comparison of the criteria of performance was done by randomly generating processing times for 180 small problems, with 4 to 6 jobs and 4 to 6 machines, and 10 large problems with 10 to 40 jobs and 10 to 40 machines. The processing times were obtained from a uniform distribution varying from 0 to 999. The small problems were solved by complete enumeration to obtain the optimal schedule according to each criterion of performance. In the case of the large problems, the solution by complete enumeration was not done in order to avoid the excessive computational effort. Instead, schedules were generated by random sampling through Monte Carlo simulation and the best solution according to each criterion was selected. The opportunity cost C_{io} of the schedule associated with each criterion of performance, i, was then determined. Let the opportunity cost for the schedule determined by the total opportunity cost criterion be C_{10}; the percentage deviation of the opportunity cost for the ith criterion can then be calculated by the following expression:

$$(3.9) \quad d = \frac{C_{io}-C_{10}}{C_{io}} \times 100; \quad i=1,2, \ldots 8.$$

It was found that the criterion defined by the combination of job waiting cost and penalty cost came closest to the performance of the total opportunity cost criterion. Next in order was the criterion formed by the combination of job waiting cost and machine idle cost. Alarmingly, makespan was ranked sixth, below penalty cost and job waiting cost. Thus, according to this research, makespan is not a good surrogate for total opportunity cost.

The logic of the total opportunity cost approach is sound, but the use of the generated costs does not permit one to determine fully the consequences for an actual production system because the relative values of the generated costs may not be the same as the relative values of the actual costs. For instance, in this analysis it appears that the relative values of the costs permit job waiting costs and penalty costs to be the nearest suboptimal criterion. With another set of cost coefficients, the best suboptimal criterion may be different. It may even be makespan!

Summary

The literature available on heuristic algorithms was reviewed in this chapter. The articles that introduced the algorithms were considered, as well as subsequent reports in which they appear. A few studies devoted to comparing algorithms were also reviewed. From the studies examined, a number of heuristic algorithms were recognized for their good performances. Dannenbring's RAES algorithm was the most outstanding while the CDS algorithm was a consistent performer. A number of the algorithms described in the review are selected for inclusion in the multivariate analysis of variance study.

The articles on criteria of performance were also examined. It appears that although many research studies have adopted the minimization of makespan as their criterion of performance, many authors are arguing that it may not truly reflect what management practices. There were suggestions that other criteria should be explored. It was also suggested that managers use multiple criteria. Therefore, there is need to use multiple criteria in research studies [48].

CHAPTER 4

RESEARCH METHODOLOGY

This chapter describes the experimental design which was used in performing the research. The research was done entirely in a laboratory environment. Considerable use was made of the computer for performing simulations and analyzing data. As in previous studies involving both the jobshop and the flowshop problems, simulation was used to generate data for analysis because large numbers of real-life problems are not available to academicians.

A sound experimental design is a necessary foundation for meaningful scientific inquiry. In this study the experimental design was divided into two categories of planning: strategic and tactical. In strategic planning, the experiment was designed to provide the necessary data. Issues such as the specification of the factors (independent variables) to be studied were considered here. The performance measures (dependent variables) to be observed were also identified at this stage of planning. Tactical planning involved establishing the details for the performance of the experiments. These included the size of the samples and the number of replications of the experiment.

Strategic Planning

This investigation was limited to the following factors:

1. Heuristic algorithms
2. Problem sizes

These factors were selected to conform, to some degree, with previous studies, which were reported in the literature [8,14,37] Also considered were other factors, including the subdivision of the second factor into two separate factors: the number of jobs and the number of machines. As mentioned in the first chapter, problem size, by definition, is measured by the combination of these two components.

Factors and Levels

Heuristic algorithms. Each of the five levels specified for this factor is an algorithm designed to solve heuristically the flowshop scheduling problem described in Chapter 2. Four of the five algorithms have already appeared in the literature while the fifth, RGES, was designed especially for this study, and is described in Appendix A. The algorithms were selected on the basis of their good performances in previous studies. The names of the five algorithms are as follows:

1. Random generation with extensive search (RGES)
2. Heuristic decomposition (HD)
3. Rapid access with extensive search (RAES)
4. Campbell, Dudek and Smith (CDS)
5. Aggarwal and Stafford (AS)

The Gupta and Maykut study reported that the HD algorithm produced better results than the pure decomposition method [58]. The HD algorithm was therefore selected on this basis as well as for the uniqueness of its approach. The RAES algorithm was selected for its superior performance in solving both small and large problems in Dannenbring's comprehensive study [37] of several heuristic algorithms. Although the CDS heuristic algorithm also appeared in the Dannenbring study [37] and was excelled by RAES, it was still chosen because it performed well in that study and in others [1,27,54] where it has been used as a standard for comparison. A recent study compared the AS heuristic algorithm with the CDS heuristic algorithm [1]. From the results, it was difficult to ascertain which performed better, except in terms of computational effort, where the AS heuristic algorithm was reported more economical.

To demonstrate the technique of each heuristic algorithm, a schedule was developed manually for a small problem. This is displayed in Appendix B.

Problem sizes. In this investigation, problems with ten or fewer jobs and ten or fewer machines were considered small in size. Most of the past research has concentrated on problems in this class. However, it is believed that larger problems would tend to resemble those faced by managers in realistic situations, especially with respect to the number of jobs. Therefore, both large and small scale problems were considered in this research.

Fifteen levels of this factor were selected for the small problems. These levels were obtained by determining all combinations of 4, 6, 8, 9 and 10 jobs with 4, 7 and 10 machines. Thus, the smallest problem size consisted of 4 jobs and 4 machines while the largest contained 10 jobs and 10 machines.

The large problems had 20 levels obtained from all combinations of 20, 40, 60, 80 and 100 jobs with 15, 30, 45 and 60 machines.

It was reported in Dannenbring's article [37] that the relative errors of the heuristic algorithms increased with the number of jobs when the number of machines was held constant, and also with the number of machines when the jobs were held constant. It would therefore be reasonable to expect the efficiency of a heuristic algorithm to vary with the problem sizes since problem size is measured by the combination of jobs and machines. Hence, problem size was studied as a factor.

Measures of Performance

In developing a schedule, its designer usually approaches the task with a number of goals in mind. Mellor [77], Elmaghraby [41] and other writers [13,16] have identified many of these goals. Associated with these goals are measures of performance, which help to determine objectively the degree to which these goals are being achieved. In the majority of past research studies, scheduling was done to satisfy a single goal: the minimization of makespan. As mentioned earlier, Gupta [45] argued that the minimization of makespan did not necessarily minimize opportunity costs, and that a combination of criteria of performance produced results which were nearly optimal.

Despite these arguments, the minimization of makespan was retained as one of the criteria of performance because it has traditionally been used and because examining a Gantt chart of even a simple flowshop problem suggests that minimizing makespan also minimizes overall machine idle time on the flowshop line. It was simply assumed that one of the criteria of performance is the fastest completion of processing on a batch of jobs.

It was also assumed that managers would like to keep their machines utilized as fully as possible since more utilization means more production, while machine idle time means no production. Therefore, the minimization of machine idle time was used as a criterion of performance.

The Gupta and Dudek study [57] indicated that the combination of job waiting cost and machine idle time cost was the second best suboptimal combination with respect to the minimization of total opportunity cost. Taking this into consideration, job waiting time was chosen as the third measure of performance. While the job is waiting, costs are incurred since it can be considered inventory in process. Gupta and Maykut [58] demonstrated that there is some degree of inverse proportionality between job waiting time and machine idle time. Also, according to Gupta [45], minimizing makespan is equivalent to minimizing idle time on the last machine. Thus, there is some correlation between all three measures of performance.

Multivariate analytical models are capable of taking these correlations into consideration without providing distorted results [114]. On the contrary, if each measure of performance is taken individually, distorted results would be obtained because univariate analysis cannot take correlation between response variables into account.

Job lateness was also considered, but rejected, as a measure of performance. It is believed that, in practice, due dates are determined in an arbitrary manner. In the absence of any empirical data on the estimating behavior of the individual who determines the due date for the delivery of the processed job, it would be difficult to arrive at a meaningful due date by any of the possible alternative methods. Since the due date, in combination with the schedule time, determines whether a job is late, the validity of the estimated job lateness would be questionable. As a result, it was decided to abandon the idea of using job lateness as a measure of performance.

TABLE 4.1

SMALL PROBLEMS

Problem Size	Heuristic Algorithm				
	1	2	3	4	5
4x4	30 Vectors of M,I,W				
4x7 . . .					
10x10					

In summary, there are two factors: heuristic algorithms and problem sizes. The heuristic algorithm factor has five levels while the problem size factor has 15 levels for the small problems and 20 levels for the large problems. There are three measures of performance, also called response variables; the values of these variables appear as a vector in each cell. The sample size (number of replications) is 30, thus yielding 30 vectors of performance measures in each cell. An explanation of the choice of the number of replications is given in the next section, which describes tactical planning. The foregoing is represented in Table 4.1.

Tactical Planning

Campbell, Dudek and Smith [27] stated that the uniform distribution provided challenging problems, and other studies [1,37,54,58] have also used this distribution for generating processing times. Adopting this approach, the processing times for the problems in this research were randomly generated from a uniform distribution, ranging from 0 to 99, inclusive. This range was chosen to provide some measurable differences in the values of the performance measures for the various algorithms.

A Lehmer-type random number generator was included in the Fortran code preceeding the programmed heuristic algorithms in order to generate the processing times for each problem. Each of the algorithms was then used to solve this problem. A vector of observations was obtained from the solution by each heuristic algorithm. Each vector was placed in a cell associated with the heuristic algorithm. The five vectors were considered repeated measures on the same subject because they resulted from solving the same problem. This process was repeated until 30 vectors of observations were obtained for each cell. The matrix of processing times for consecutive replications within a single cell was generated using consecutive strings of random numbers from the random number stream. For each experimental group of subjects, a new random number seed was used, thus ensuring that processing times were randomly selected. (The random number seeds are recorded in Appendix C.)

In the experiment, each heuristic algorithm solved the same set of problems as the other algorithms. Thus, repeated measures were obtained on the same subject. This repeated measures approach has been the standard practice in past research studies and was adopted here to conform with past procedures. This means that the performance measures for a particular heuristic algorithm will be correlated with measures for other algorithms, thus creating dependence between cells. On the other hand, the error variance is reduced by removing one source of variation [69].

Multiple responses exist because data were obtained on three variables from each solution. This experimental design can therefore be described as a multivariate repeated measures, multi-response design.

Sample Size

The number of problems (replications) of a particular size is called the sample size. The sample size was fixed at 30 so that it may be assumed that the distributions of both the sums and means of the observations approximate a normal distribution. This follows from the central limit theorem, which suggests a sample size of 25 or more observations to allow the assumption of normality when little is known about the distribution of the observations [31]. Fixing the sample size at thirty also ensured that cell sizes for the multivariate model were equal. This equality tended to minimize the adverse effects of violating the assumptions of equal variance-covariance matrices for the multivariate analysis of variance model [69,109].

Source of Data

Data were obtained by using the computer to simulate the operation of the flowshop. The simulation model of the flowshop was programmed for the computer in Fortran. The jobs were sequenced through the flowshop simulation model in accordance with the schedule developed by the heuristic algorithm while the values of the response variables were calculated by a subroutine. These values are recorded by the computer and returned as the output of the simulation process.

The length of time required by the computerized heuristic algorithm to arrive at a schedule was measured and recorded as data. The measurements were performed by a subroutine that utilizes the internal clock of the computer.

Optimum makespan values of the measures of performance were determined for the small problems by an implicit enumeration method [1]. These values were compared with those obtained for the algorithms in order to determine the percentage deviation of the heuristic algorithm values.

The Multivariate General Linear Model

The multivariate general linear model was employed in this multivariate analysis of variance of the multi-response repeated measures and in the testing of the hypotheses using the p-variate vectors of observations. Considerable use was made of the theoretical framework which appears in Timm [108 ,109] and Carlson and Timm [28].

Assuming that each p-variate vector of observations associated with a subject is multi-normally distributed with common unknown variance-covariance, Σ, the entire collection of N observations on the p dependent variables may be represented by the following expression:

(4.1) $Y = XB + Eo$

with expectation,

(4.2) $E(Y) = XB$

and variance,

(4.3) $V(Y) = I_n \otimes \Sigma$

where the symbol $\otimes$ is the Kronecker or direct product of two matrices.

In the above expressions,

Y is a NxP data matrix,

X is a Nxq design matrix of known constants, with linearly independent columns,

B is an unknown qxp parameter matrix of population means,

Eo is a Nxp random error matrix,

I_n is an identity matrix,

q is the number of conditions.

The unknown parameter vectors in the matrix B were determined by the solution of the normal equation:

(4.4) $X' X B = X'Y$

The estimates were designated $\hat{B}$. The unknown elements, σ_{ij}, of the variance-covariance matrix, Σ, were obtained from the sum of the squares and cross products (SSP) matrix due to error. This matrix was derived using the expression:

(4.5) $Se = Y'Y - Y' X \hat{B}$

Testing Linear Hypotheses

Using the multivariate linear approach, null hypotheses can be expressed in the form:

(4.6) $Ho : CBA = \Gamma$

where C is a $\nu_h$xq matrix of rank $\nu_h \leq q$

A is a pxt known real matrix of rank $t \leq p$

Γ is $\nu_h$xt matrix of known constants

ν_h is the number of degrees of freedom associated with the hypotheses

ν_e is the number of degrees of freedom associated with the error.

Before testing the hypothesis, computation of the sum of squares and products matrices for the hypothesis, Qh, and for the error, Qe, was done with the following expressions:

(4.7) $$Qh = (C\hat{B}A - \Gamma)'(C(X'X)^{-1}C')^{-1}(C\hat{B}A - \Gamma)$$

(4.8) $$Qe = A'Y'[I-X(X'X)^{-1}X']YA$$

Determining the truth of the null hypothesis was done with a number of techniques which incorporate Qh and Qe. They all utilized the roots of one of the following characteristic equations:

(4.9) $$|Qh - \lambda Qe| = 0$$

(4.10) $$|Qe - v(Qe + Qh)| = 0$$

(4.11) $$|Qh - \theta(Qh + Qe)| = 0$$

Jones [66] stated that it is necessary for Qe to be nonsingular in order to obtain solutions of the characteristic equations. The roots of equation (4.9) are $\lambda_1, \lambda_2, \ldots, \lambda_s$, where they are ordered from largest to smallest and $s = \min(\nu_h, t)$.

Ho was tested with Wilks' criterion [109], which is calculated from the following equation:

(4.12) $$\Lambda = \frac{|Qe|}{|Qe+Qh|} = \prod_{i=1}^{s}(1 + \lambda_i)^{-1}$$

where Λ denotes Wilks' criterion and $U^{\alpha}(t,\nu_h,\nu_e)$ is the lower percentage point of the U distribution.

Ho was rejected at significance level α, if

(4.13) $$\Lambda < U^{\alpha}(t,\nu_h,\nu_e)$$

The value of Wilks' criterion can be converted to an F-statistic by using an expression developed by Rao and reported in 34, p. 227.

Another test of Ho involved Pillai's trace criterion [28], which was derived from the following expression:

(4.14) $$V^{(s)} = Tr[Qh(Qh+Qe)^{-1}] = \sum_{i=1}^{s} \theta i = \sum_{i=1}^{s} \frac{\lambda_i}{1+\lambda_i}$$

Using a significance level of α, Ho was rejected if $V^{(s)} > V^{\alpha}(s,m,n)$. In the preceding expression, $m = (|\nu_h - t| - 1)/2$, $n = (\nu_e - t - 1)/2$, $s = \min(\nu_h, t)$, and $V^{\alpha}(s,m,n)$ is the critical constant taken from the upper percentage points of the V distribution [109].

The Hotteling-Lawley Trace criterion [28] was also used to test Ho. The formula for this criterion is

(4.15) $$U^{(s)} = Tr(QhQe^{-1}) = \sum_{i=1}^{s} \lambda_i$$

Ho was rejected when $U^{(s)} > Uo^{\alpha}(s,m,n)$. A table of the distribution of the critical constant $Uo^{\alpha}(s,m,n)$ is also provided in Timm [109].

Roy's largest root criterion [28] can also be used to test Ho. The criterion, θ, was determined by the following expression:

(4.16) $$\theta = \lambda_1/(1+\lambda_1)$$

The hypothesis was rejected if $\theta > \theta^{\alpha}(s,m,n)$. Critical values were obtained from the distribution for the largest root, which is provided in Timm [109].

When s = 1, all criteria are the same. Beyond that value of s, there is a divergence. There is no clear indication of which criterion is better. According to Harris [61], the majority of texts and computer programs report the Wilks' criterion for significance tests. Among the reasons given is the ease with which the value of Λ can be converted to an approximate F-value, for which well developed and extensive tables are readily available. Tables for the other criteria are not as thorough and ubiquitous. Jones [66] stated that the power functions of the criteria may depend upon the values of all the population roots. He stated that when more than one root is large, Wilks' criterion has its greatest power, while at the same time, Roy's largest root criterion is at its minimum power. On the other hand, the distribution of the roots of the trace criteria has no effect on their power. He also mentioned that the advantage of the largest root criterion is its ability to provide simultaneous confidence bounds for several important parameters. According to Jones, one of the virtues of the trace criterion is that knowing the values of the characteristic equations' separate roots is not necessary for performing tests. This is because the sum of the roots, $\Sigma\lambda_i$, is also equal to the sum of the diagonal elements of matrix, $QhQe^{-1}$.

The Hypotheses

The research hypotheses developed for this study sought to investigate the comparative abilities of the various heuristic algorithms in providing schedules. The hypotheses were designed to facilitate testing by inferential statistical methods. The hypotheses are stated as follows:

When multiple criteria of performance are used:

1. There is no interaction effect between the heuristic algorithms and the size of the problems.
2. The heuristic algorithms determine equally efficient schedules.
3. There is no difference between the results yielded by the various problem sizes.

Using a single criterion of performance:

4. Given a fixed problem size, the five heuristic algorithms require an equal amount of computational effort, which is measured by the amount of computer time needed to perform the processing.
5. Adopting the minimization of makespan as the criterion of performance, the five heuristic algorithms are equally efficient.
6. Adopting the minimization of idle time as the criterion of performance, the five algorithms are equally efficient.

7. Adopting the minimization of job waiting time as the criterion of performance, the five algorithms are equally efficient.

Non-statistical test:

8. For small problems, the heuristic algorithms produce an equal number of optimal schedules.

Analytical Procedures

As an initial step, the data were summarized by calculating the mean values of the response variables in each cell. The format of these means appear in Table 4.2. The mean values are graphed with a separate graph for each dependent variable. In the case of the small problems where the optimal value for the dependent variable, makespan, has been determined, the average percentage deviation between the values determined by the heuristic algorithm and the optimal values is reported for each problem size and heuristic algorithm. For both the small and the large problems, the average ratio of the value of the dependent variable for each algorithm to the same value for a particular algorithm, RAES, is reported.

Because there were multiple response variables and repeated measures on each subject, the multivariate analysis of variance technique was employed. This approach was preferred rather than univariate analysis with each dependent variable because the latter technique, unlike the former, does not take into consideration any correlation

TABLE 4.2

MEANS FOR REPEATED MEASURES DATA

Performance Measures		Makespan					Job Waiting Time					Machine Idle Time				
Heuristic Algorithms		1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
Problem Sizes	P_1	μ_{11}	μ_{12}	μ_{13}	μ_{14}	μ_{15}	μ_{16}	μ_{17}	μ_{18}	μ_{19}	$\mu_{1,10}$	$\mu_{1,11}$	$\mu_{1,12}$	$\mu_{1,13}$	$\mu_{1,14}$	$\mu_{1,15}$
	• • • •															
	P_k	μ_{k1}	μ_{k2}	μ_{k3}	μ_{k4}	μ_{k5}	μ_{k6}	μ_{k7}	μ_{k8}	μ_{k9}	$\mu_{k,10}$	$\mu_{k,11}$	$\mu_{k,12}$	$\mu_{k,13}$	$\mu_{k,14}$	$\mu_{k,15}$
	• • • •															
	P_w*	μ_{w1}	μ_{w2}	μ_{w3}	μ_{w4}	μ_{w5}	μ_{w6}	μ_{w7}	μ_{w8}	μ_{w9}	$\mu_{w,10}$	$\mu_{w,11}$	$\mu_{w,12}$	$\mu_{w,13}$	$\mu_{w,14}$	$\mu_{w,15}$

*w equals 15 and 20 for the small and large problems, respectively.

between the dependent variables [60,61,91]. There is also the danger of compounding the Type I error when multiple univariate tests are done, because each univariate test is designed to yield a significant result α X 100% of the time when the null hypothesis is true [61]. Furthermore, the assumptions for multivariate analysis of variance are not as demanding as those for univariate analysis [108].

Profile analysis, an approach often used to analyze multivariate repeated measures data in other studies [21,32,35,61,82,108,109], was adopted for this study. In order to accommodate multiple dependent variables in the profile analysis framework, new dependent variables were created by forming them from the combination of each former dependent variable and each level of the factor, heuristic algorithm. Thus, there were 15 new dependent variables and each column in Table 4.2 contains information for a particular new variable. Hence, the entry μ_{11} represents the mean value of the 30 observations in the cell for problem size 1 (4 jobs X 4 machines) and for the new dependent variable 1, which is formed from the former dependent variable, makespan, and the first heuristic algorithm, RGES.

Testing of the Hypotheses

As stated earlier in this chapter's section on testing linear hypotheses, accepting or rejecting hypotheses depended upon the derived values of certain test criteria.

This section discusses the procedures for testing the hypotheses.

The test for interaction. The first hypothesis sought to determine whether or not there is a significant interaction between problem size and heuristic algorithms. It performed this determination by checking for parallelism of the profiles. This check was done by developing transformations on the data in Table 4.2, and performing one-way multivariate analysis of variance on the transformations with the problem size as the only factor. Thus, the algebraic representation of the first hypothesis being tested by profile analysis is:

$$Ho1 = \begin{bmatrix} \mu_{11} - \mu_{13} \\ \mu_{12} - \mu_{13} \\ \mu_{14} - \mu_{13} \\ \mu_{15} - \mu_{13} \\ \mu_{16} - \mu_{18} \\ \mu_{17} - \mu_{18} \\ \mu_{19} - \mu_{18} \\ \mu_{1,10} - \mu_{18} \\ \mu_{1,11} - \mu_{1,13} \\ \mu_{1,12} - \mu_{1,13} \\ \mu_{1,14} - \mu_{1,13} \\ \mu_{1,15} - \mu_{1,13} \end{bmatrix} = \ldots = \begin{bmatrix} \mu_{81} - \mu_{83} \\ \mu_{82} - \mu_{83} \\ \mu_{84} - \mu_{83} \\ \mu_{85} - \mu_{83} \\ \mu_{86} - \mu_{88} \\ \mu_{87} - \mu_{88} \\ \mu_{89} - \mu_{88} \\ \mu_{8,10} - \mu_{88} \\ \mu_{8,11} - \mu_{8,13} \\ \mu_{8,12} - \mu_{8,13} \\ \mu_{8,14} - \mu_{8,13} \\ \mu_{8,15} - \mu_{8,13} \end{bmatrix} = \ldots = \begin{bmatrix} \mu_{15,1} - \mu_{15,3} \\ \mu_{15,2} - \mu_{15,3} \\ \mu_{15,4} - \mu_{15,3} \\ \mu_{15,5} - \mu_{15,3} \\ \mu_{15,6} - \mu_{15,8} \\ \mu_{15,7} - \mu_{15,8} \\ \mu_{15,9} - \mu_{15,8} \\ \mu_{15,10} - \mu_{15,8} \\ \mu_{15,11} - \mu_{15,13} \\ \mu_{15,12} - \mu_{15,13} \\ \mu_{15,14} - \mu_{15,13} \\ \mu_{15,15} - \mu_{15,13} \end{bmatrix}$$

In order to test this hypothesis using the multivariate general linear model, it was necessary to specify the C, A and Γ matrices for the expression, Ho: $CBA = \Gamma$. As already indicated, the B matrix is a matrix of parameters which is represented by the matrix of means in Table 4.2. For the small problem, the dimension of this matrix was 15x15 while it was 20x15 for the large problems. The row dimension represents the number of levels for the problem size factor while the column dimension represents the number of new dependent variables.

The C matrix contains the contrasts. It is specified below in a partitioned form:

$$C = \left| I_k \,\vdots\, -\underline{1} \right|$$

where $k = q-1 = \nu_h$ (the hypothesis degrees of freedom)

I_k is a kxk identity matrix

$-\underline{1}$ is a column vector of minus ones

q = number of levels of the problem size factor.

The post matrix A, which enabled the development of hypotheses among different response parameters [78], was partitioned as follows:

$$A = \left[\begin{array}{c:c:c} A1 & 0 & 0 \\ \hdashline 0 & A1 & 0 \\ \hdashline 0 & 0 & A1 \end{array}\right]$$

where 0 is a 5x4 matrix of zeros

and A1 is the 5x4 matrix that is specified below.

$$A1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & -1 & -1 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The above matrix was designed to facilitate comparisons between the RAES algorithm and all of the other heuristic algorithms.

The Γ matrix consisted of a kx12 matrix of zeros, where k equals ν_h, the hypothesis degrees of freedom.

Thus, in testing for interaction, the dimensions of the matrices were:

$$\text{Ho1:} \quad \underset{(q-1)xq}{C} \quad \underset{qxp}{B} \quad \underset{px12}{A} = \underset{kx12}{\Gamma}$$

where $q-1 = \nu_h$ (equals 14 and 19 for the small and large problems, respectively)

and p = number of new dependent variables(i.e. 15)

<u>Testing for the differences between heuristic algorithms</u>. If parallelism was not tenable, the hypothesis which tests differences between heuristic algorithms was as follows:

$$Ho2 = \begin{bmatrix} \mu_{11} \\ \mu_{21} \\ \mu_{31} \\ \mu_{41} \\ \vdots \\ \mu_{15,1} \\ \mu_{1,6} \\ \mu_{2,6} \\ \vdots \\ \mu_{15,6} \\ \mu_{1,11} \\ \mu_{2,11} \\ \vdots \\ \mu_{15,11} \end{bmatrix} = \cdots = \begin{bmatrix} \mu_{13} \\ \mu_{23} \\ \mu_{33} \\ \mu_{43} \\ \vdots \\ \mu_{15,2} \\ \mu_{1,8} \\ \mu_{2,8} \\ \vdots \\ \mu_{15,8} \\ \mu_{1,13} \\ \mu_{2,13} \\ \vdots \\ \mu_{15,13} \end{bmatrix} = \cdots = \begin{bmatrix} \mu_{15} \\ \mu_{25} \\ \mu_{35} \\ \mu_{45} \\ \vdots \\ \mu_{15,5} \\ \mu_{1,10} \\ \mu_{2,10} \\ \vdots \\ \mu_{15,10} \\ \mu_{1,15} \\ \mu_{2,15} \\ \vdots \\ \mu_{15,15} \end{bmatrix}$$

The matrices were of the form:

C is I_q, an identity matrix

A,B are the same matrices used for the test of parallelism

Γ is a qx12 matrix of zeros

where q = number of levels of the factor.

If parallelism was tenable, the matrices were as follows:

$$\underset{(1xq)}{C} = (1/q, 1/q, \ . \ \ . \ \ ., 1/q)$$

A,B are the same matrices used in the test of parallelism.

Γ is a 1x12 matrix (or a vector) of zeros.

In algebraic form, the hypothesis is expressed as:

$$\text{Ho2:} \begin{bmatrix} \sum_{i=1}^{q} \mu_{i1}/q \\ \\ \sum_{i=1}^{q} \mu_{i6}/q \\ \\ \sum_{i=1}^{q} \mu_{i,11}/q \end{bmatrix} = \ . \ . \ . = \begin{bmatrix} \sum_{i=1}^{q} \mu_{i3}/q \\ \\ \sum_{i=1}^{q} \mu_{i8}/q \\ \\ \sum_{i=1}^{q} \mu_{i,13}/q \end{bmatrix} = \ . \ . \ . = \begin{bmatrix} \sum_{i=1}^{q} \mu_{i5}/q \\ \\ \sum_{i=1}^{q} \mu_{i,10}/q \\ \\ \sum_{i=1}^{q} \mu_{i,15}/q \end{bmatrix}$$

Testing differences between problem sizes. When parallelism was not tenable, the parametric form of this hypothesis was as follows:

$$
Ho3 = \begin{bmatrix} \mu_{11} \\ \mu_{12} \\ \mu_{13} \\ \mu_{14} \\ \mu_{15} \\ \mu_{16} \\ \mu_{17} \\ \mu_{18} \\ \mu_{19} \\ \mu_{1,10} \\ \mu_{1,11} \\ \mu_{1,12} \\ \mu_{1,13} \\ \mu_{1,14} \\ \mu_{1,15} \end{bmatrix} = \ldots = \begin{bmatrix} \mu_{81} \\ \mu_{82} \\ \mu_{83} \\ \mu_{84} \\ \mu_{85} \\ \mu_{86} \\ \mu_{87} \\ \mu_{88} \\ \mu_{89} \\ \mu_{8,10} \\ \mu_{6,11} \\ \mu_{8,12} \\ \mu_{8,13} \\ \mu_{8,14} \\ \mu_{8,15} \end{bmatrix} = \ldots = \begin{bmatrix} \mu_{15,1} \\ \mu_{15,2} \\ \mu_{15,3} \\ \mu_{15,4} \\ \mu_{15,5} \\ \mu_{15,6} \\ \mu_{15,7} \\ \mu_{15,8} \\ \mu_{15,9} \\ \mu_{15,10} \\ \mu_{15,11} \\ \mu_{15,12} \\ \mu_{15,13} \\ \mu_{15,14} \\ \mu_{15,15} \end{bmatrix}
$$

The following matrices were required for this test:

B,C are the same matrices as those used in the test of parallelism

A is an I_{15} identity matrix

Γ is a $(q-1)\times15$ matrix of zeros.

When parallelism was tenable, the matrices were as follows:

B,C are the same matrices as above

Γ is a $(q-1)\times3$ matrix of zeros

A is a 15×3 matrix which is partitioned as follows:

$$A = \begin{bmatrix} \underline{V1} & \underline{0} & \underline{0} \\ \underline{0} & \underline{V1} & \underline{0} \\ \underline{0} & \underline{0} & \underline{V1} \end{bmatrix} .$$

where $\underline{V1}$ is a column vector of 5 elements, each being 1/5

and $\underline{0}$ is a column vector of 5 elements, each being zero.

This hypothesis is represented in algebraic form as:

$$\text{Ho3:} \quad \begin{bmatrix} \sum_{j=1}^{5} \mu_{1j}/5 \\ \sum_{j=6}^{10} \mu_{1j}/5 \\ \sum_{j=11}^{15} \mu_{1j}/5 \end{bmatrix} = \ldots = \begin{bmatrix} \sum_{j=1}^{5} \mu_{kj}/5 \\ \sum_{j=6}^{10} \mu_{kj}/5 \\ \sum_{j=11}^{15} \mu_{kj}/5 \end{bmatrix} = \ldots = \begin{bmatrix} \sum_{j=1}^{5} \mu_{15,j}/5 \\ \sum_{j=6}^{10} \mu_{15,j}/5 \\ \sum_{j=11}^{15} \mu_{15,j}/5 \end{bmatrix}$$

The computations for the above three hypotheses were done by the Full Rank Multivariate Linear Model (FRMLM) program which was reported by Carlson and Timm [28]. The output from this program includes Wilks' criterion, the multivariate F-statistic, and the trace criteria. Univariate results, which include the univariate F-statistic, are also provided.

The remaining hypotheses. In testing the fourth hypothesis, another analysis of variance was done. In this case, only one dependent variable, the CPU time required for arriving at schedules, was used. It is possible to use univariate analysis of variance if the assumptions of compound symmetry of the variance-covariance matrix are met. Otherwise, it is safer to use multivariate analysis of variance since this latter technique does not require the assumption. In either case, an F-statistic was obtained and the hypothesis of equal computational effort for all heuristic algorithms was rejected if the F-statistic yielded from the sums of the squares for the hypotheses and errors was greater than the critical F-ratio for the significance level specified. The F-test was preferred because of its robustness with respect to the required assumptions.

The fifth, sixth and seventh hypotheses were also based on a single response variable, and all of the comments made on the previous hypothesis with respect to the selection of a univariate or multivariate approach applied to each of these hypotheses. Again, the F-statistic was used to accept or reject each hypothesis.

The final hypothesis, which states that the heuristic algorithms yield an equal number of optimal schedules, was tested in a non-statistical manner by counting and comparing the actual number of optimal schedules.

In the above tests, an initial significance level of .05 was chosen. On those occasions when it appeared that another level would produce more useful results, the new level was adopted and further investigation pursued.

Post Hoc Tests

If the hypothesis concerning the equality of the heuristic algorithms was rejected, further analyses were done to determine which algorithm was responsible for this lack of equality. It was also possible that a subset, rather than the total number of dependent variables, may have contributed to this lack of equality. A number of writers [60,61,66,107,109] have suggested post hoc tests when multivariate hypotheses are rejected. Timm's discriminant analysis approach [109] was used in this research.

Summary

This chapter outlined the research methodology utilized in performing the investigation. It specified the factors and their levels, and it identified the measures of performance: makespan, machine idle time and job waiting time. It also explained how data were obtained as an output of computer simulation. The various analytical and testing techniques were specified as well as the hypotheses which were tested. The criteria required for the acceptance or rejection of hypotheses were also indicated.

CHAPTER 5

ANALYSIS OF DATA AND RESULTS

This chapter reports and discusses the experimental investigation on the heuristic algorithms. The initial section provides a summary of the results. This is followed by a discussion of the data generated by the simulation model of the flowshop and then by a statistical analysis of the same data. The results of post hoc analyses are described. Throughout the chapter the results are interpreted and evaluated.

Summary of the Results

The results obtained from the testing of the hypotheses are as follows:

A. When the three measures of performance are considered simultaneously:

   1. There is a significant interaction between the problem sizes and heuristic algorithms which means that some algorithms were more efficient at a particular problem size.

2. The five heuristic algorithms produce significantly different results
3. The observations on the three performance measures are significantly different for the various problem sizes.

B. The computational efforts, as measured by the CPU time, are significantly different for the heuristic algorithms.

C. When each performance measure is considered separately, the data generated by the heuristic algorithms are significantly different except in one batch of comparisons.

D. The heuristic algorithms do not produce an equal number of optimal solutions. In fact, there is a great disparity among the numbers produced.

Description of Data Generated

All of the data were generated using the IBM 370/158 computer of the Computer Center, the University of Oklahoma. A total of 450 small problems and 600 large problems were solved by each heuristic algorithm. The experiments were designed so that the heuristic algorithms solved the same problems. The data derived from these solutions included a schedule and the CPU time required by the heuristic algorithm to arrive at a solution. The resulting schedule was then evaluated by the simulation model of the flowshop which yielded values of the performance measures. Thirty problems were solved for each problem size and the results were averaged over all of the problems. The raw data on the performance measures and CPU time are not reported here since it is estimated that there are 9000 items for the small problems, and 12,000 for the large problems. This would involve dozens of pages of tables and would be too voluminous. Instead, the mean values for each cell are tabulated and reported in Appendices D and E for the small and large problems, respectively. The means for the small problems are plotted on graphs in Figures 5.1, 5.2 and 5.3, with each figure representing a performance measure. The corresponding graphs for the large problems appear in Figures 5.4, 5.5 and 5.6. There are several graphs in each figure so each line is identified by a letter. These letters are related to the various problem sizes in Tables 5.1 and 5.2 for the small and large jobs, respectively.

Although the graphs of the means of the makespan values for the small problems reveal that the RAES heuristic algorithm produced the lowest values, this heuristic algorithm did not have a monopoly on these figures. At the smallest problem sizes, the RGES algorithm matched its results, and the values yielded by the AS and CDS algorithms were also comparable. The closeness of the results of the AS and CDS algorithms and the superior results of the RAES algorithm confirmed the findings reported in earlier studies [1,37]. An examination of the larger problems revealed that

TABLE 5.1

CODES FOR THE SMALL PROBLEMS

Problem Size	Letter Code	Numerical Code	Problem Size	Letter Code	Numerical Code
4x4	A	1	8x7	H	8
4x7	B	2	8x10	I	9
4x10	C	3	9x4	J	10
6x4	D	4	9x7	K	11
6x7	E	5	9x10	L	12
6x10	F	6	10x4	M	13
8x4	G	7	10x7	N	14
			10x10	O	15

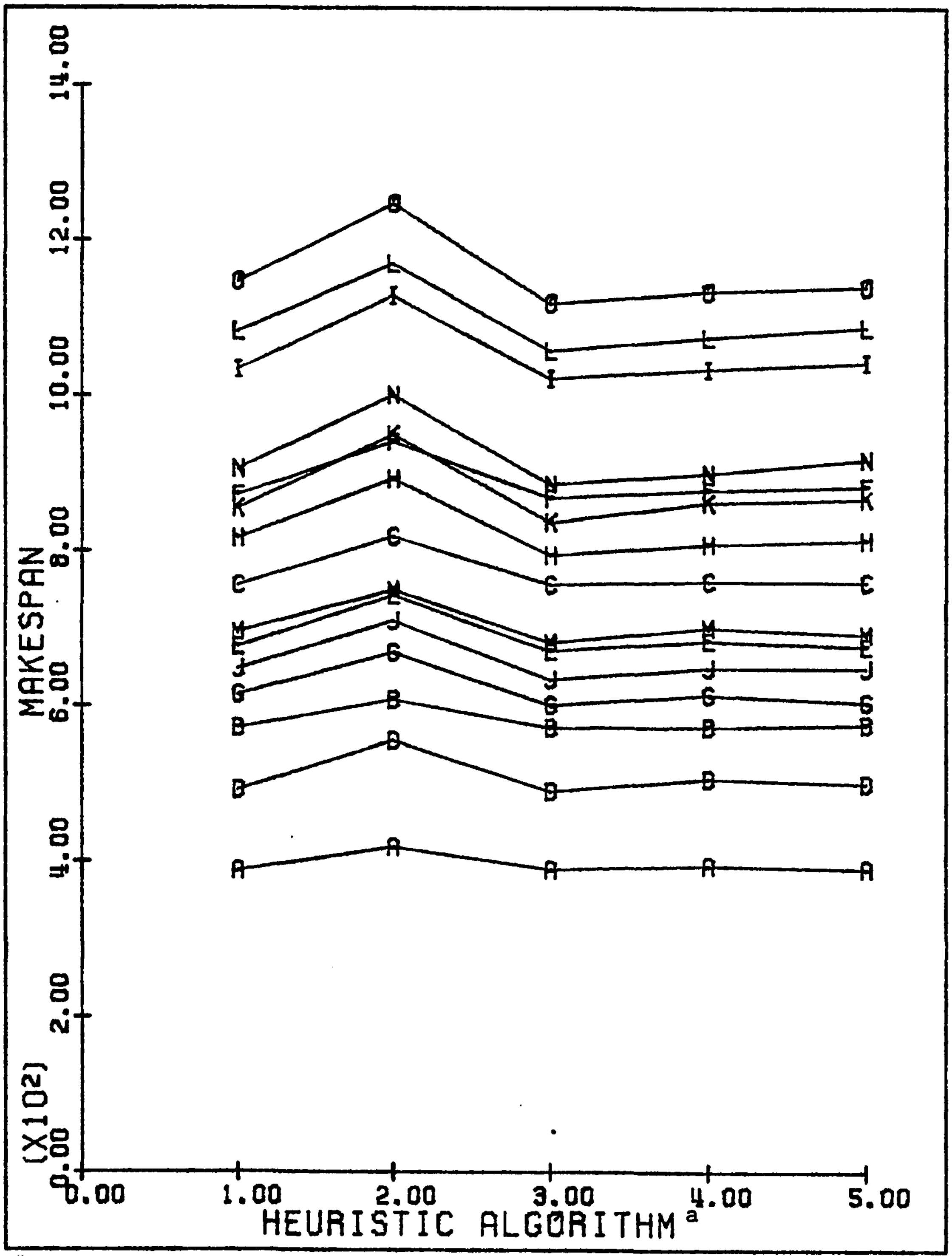
TABLE 5.2

CODES FOR THE LARGE PROBLEMS

Problem Size	Letter Code	Numerical Code	Problem Size	Letter Code	Numerical Code
20x15	A	16	60x45	K	26
20x30	B	17	60x60	L	27
20x45	C	18	80x15	M	28
20x60	D	19	80x30	N	29
40x15	E	20	80x45	O	30
40x30	F	21	80x60	P	31
40x45	G	22	100x15	Q	32
40x60	H	23	100x30	R	33
60x15	I	24	100x45	S	34
60x30	J	25	100x60	T	35

FIGURE 5.1

MAKESPAN MEANS, SMALL PROBLEMS

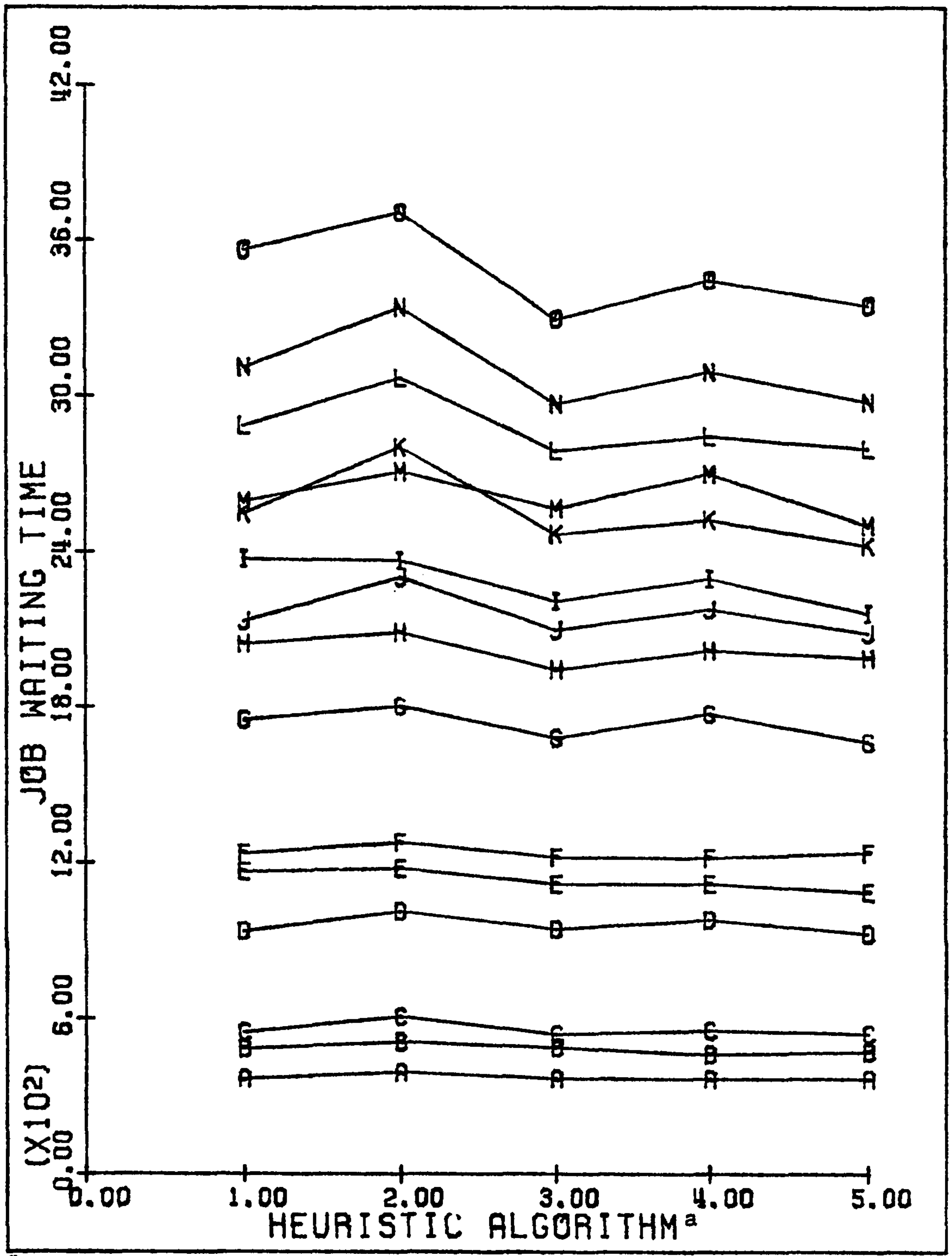


*See Table 5.1 for interpretation of letter codes

^{a}1=RGES; 2=HD; 3=RAES; 4=CDS;5=AS

FIGURE 5.2

JOB WAITING TIME MEANS, SMALL PROBLEMS

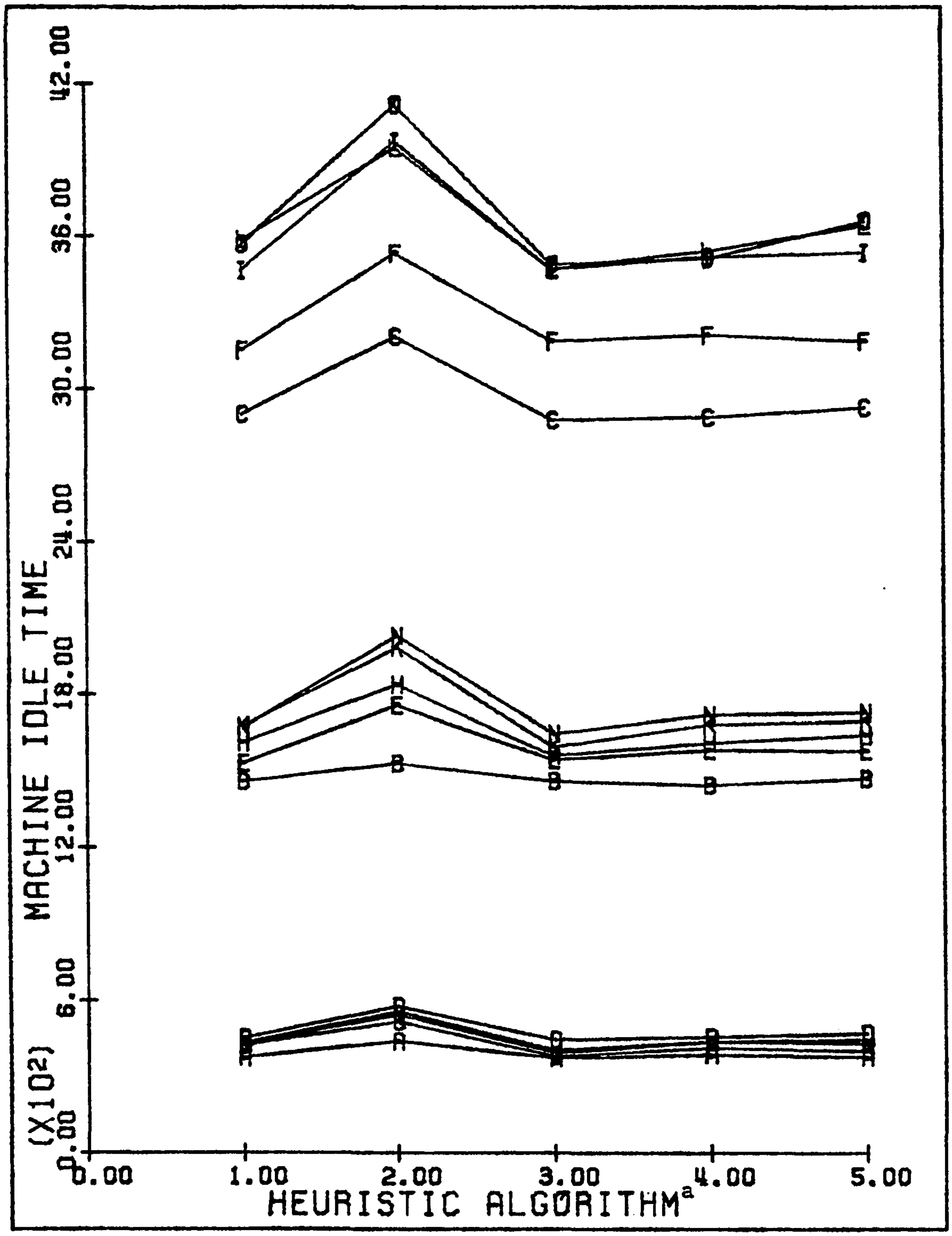


*See Table 5.1 for interpretation of letter codes

^{a}1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.3

MACHINE IDLE TIME MEANS; SMALL PROBLEMS

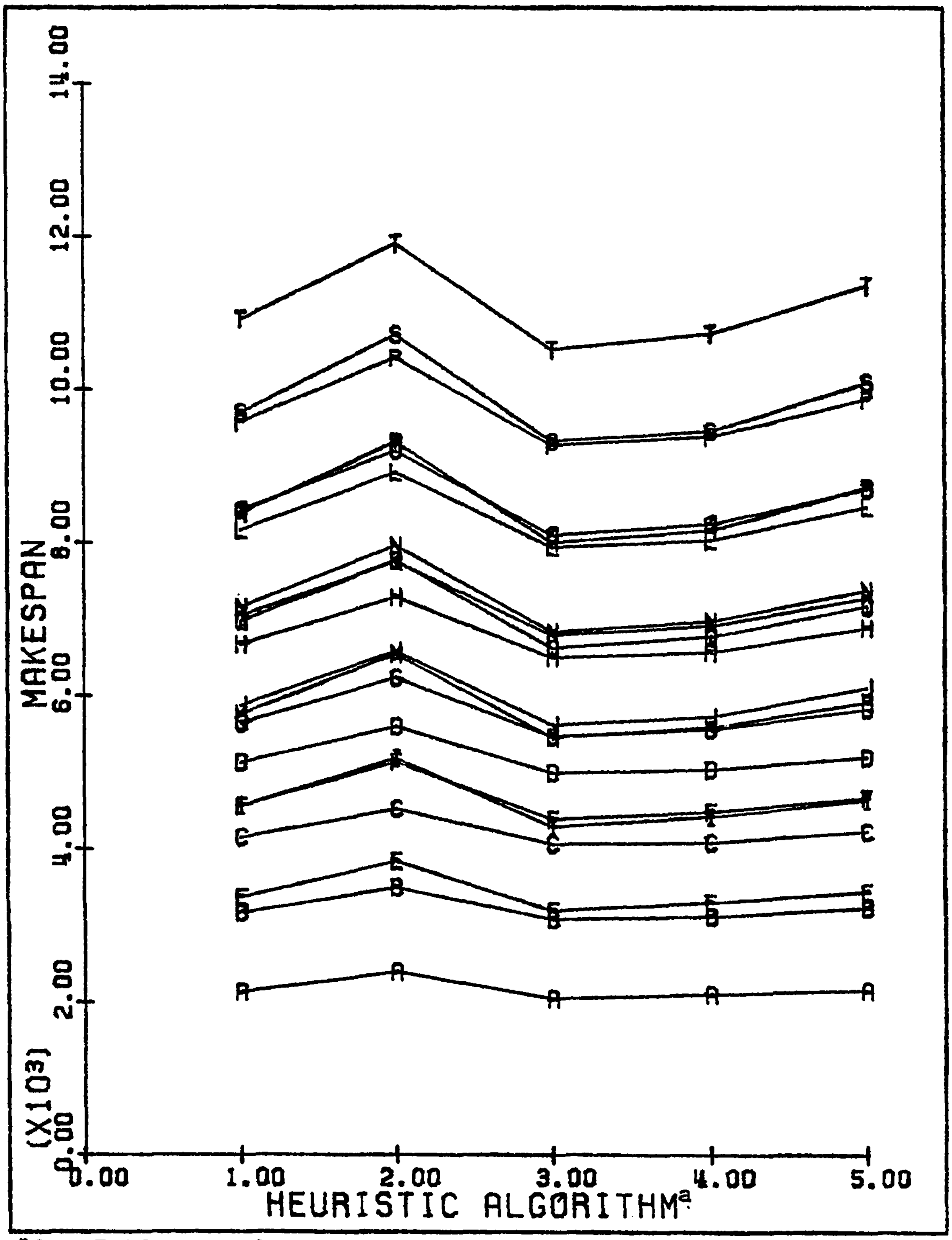


*See Table 5.1 for interpretation of letter codes

ª1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.4

MAKESPAN MEANS, LARGE PROBLEMS

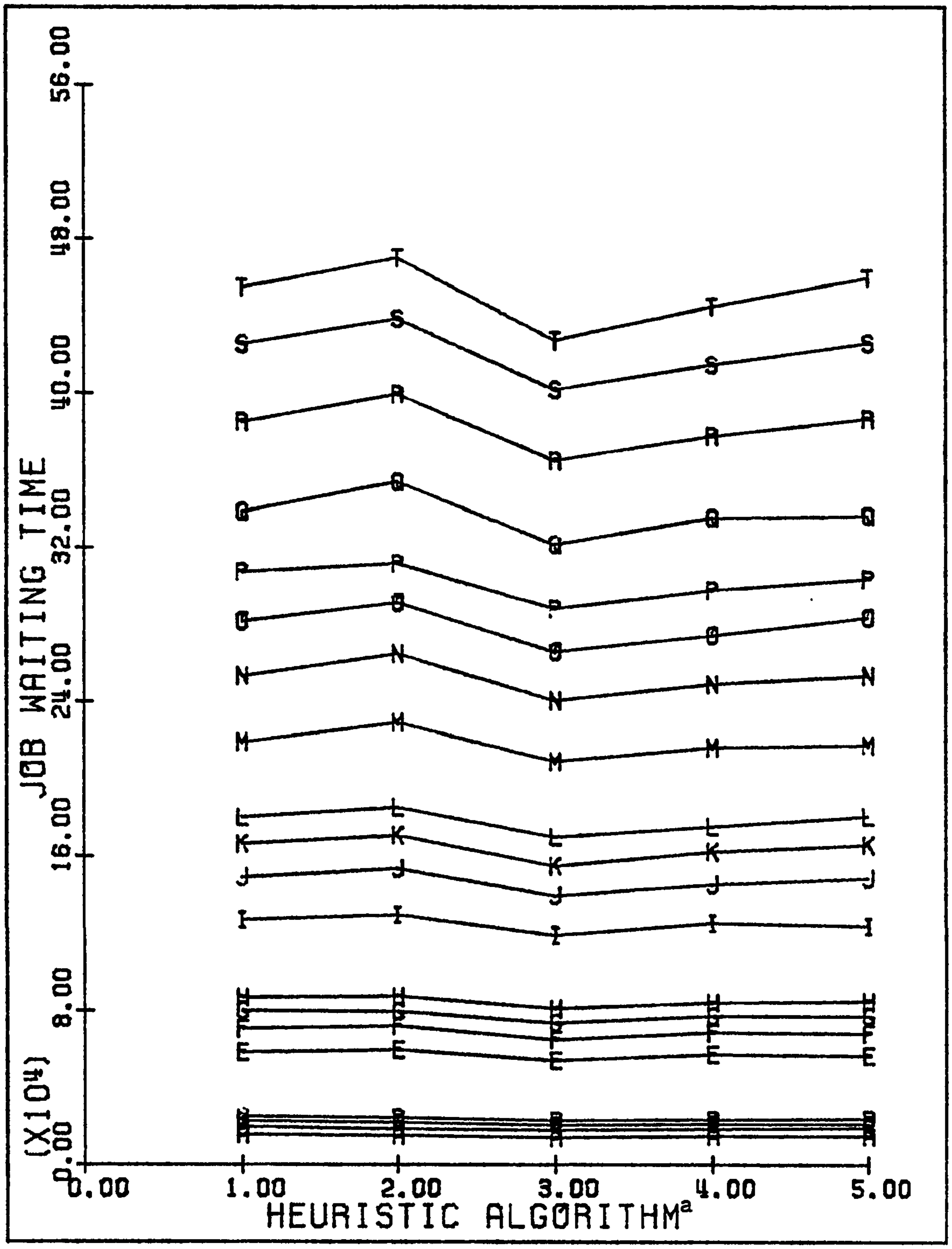


*See Table 5.2 for interpretation of letter codes

[a]1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.5

JOB WAITING TIME MEANS, LARGE PROBLEMS

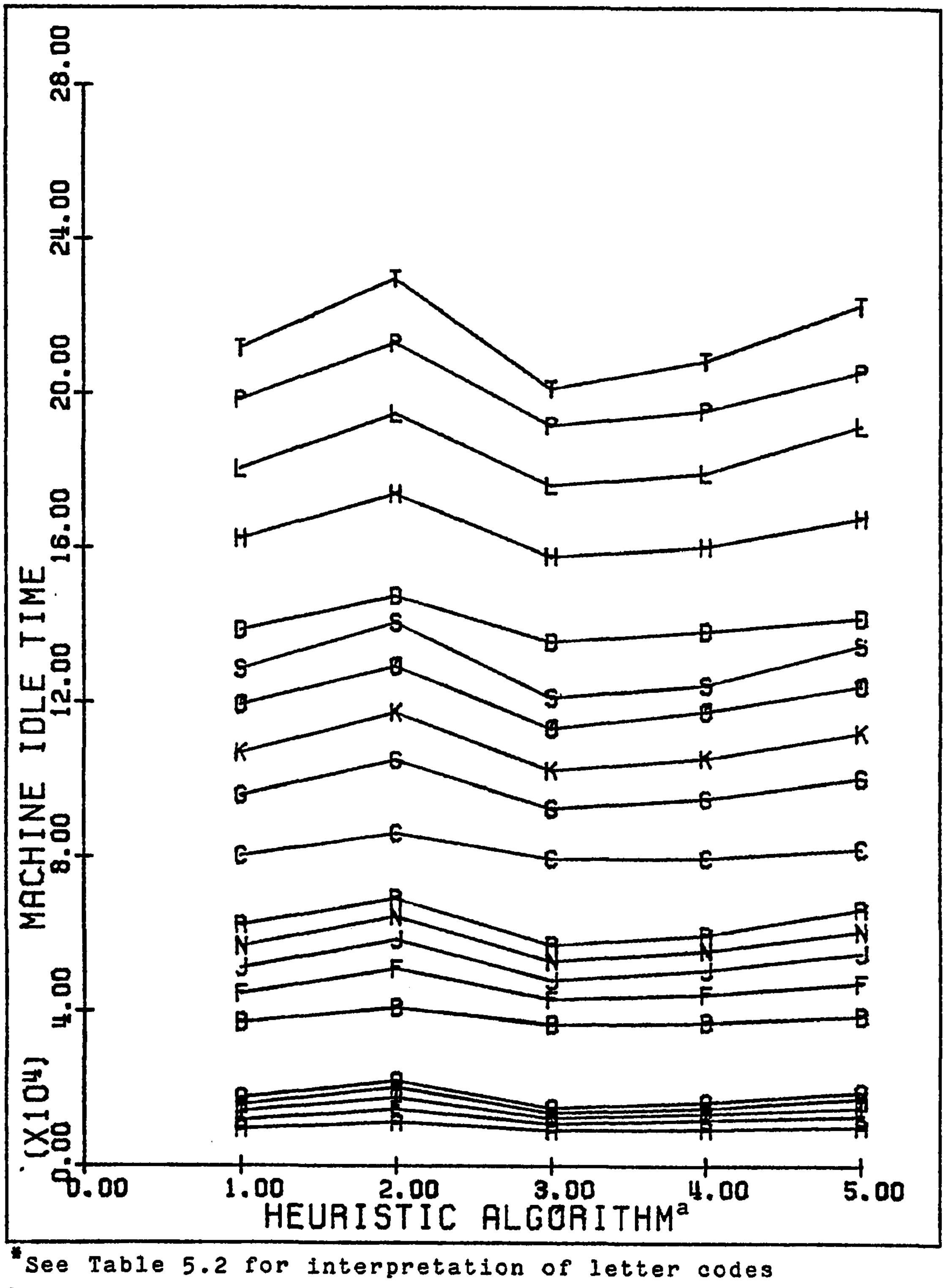


*See Table 5.2 for interpretation of letter codes

[a] 1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.6

MACHINE IDLE TIME MEANS, LARGE PROBLEMS



*See Table 5.2 for interpretation of letter codes

ᵃ1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

the RAES algorithm produced the lowest makespan values. In general, the values produced by the CDS algorithm were not much higher and were followed, in order of magnitude, by the values for the RGES, AS and HD algorithms. At all problem sizes, the HD algorithm produced the worst results.

When machine idle time is considered as the criterion, a plot of the means for the small problems indicates that, in general, the RAES algorithm produced the best results. On a relatively small number of problem sizes, the RGES, CDS and AS algorithms took turns at producing the lowest values for this criterion. In the case of the large problems, the RAES algorithm yielded the best results at all problem sizes. With the exception of the HD algorithm and, on occasions, the RGES and AS algorithms, the values produced for this measure of performance were very similar in magnitude.

The machine idle times seem to vary much more significantly with the number of machines than with the number of jobs. This is demonstrated in Figure 5.3 by the cluster of lines for various problem sizes when the number of machines were held constant and the number of jobs varied. For the small jobs, the machine idle times for the 4x4, 6x4, 8x4, 9x4 and 10x4 problems varied by a minimum of 77 units for the RGES algorithm. On the other hand, the values for the 4x4, 4x7 and 4x10 problem sizes varied by as much as

2,526 units for the same algorithm, thus illustrating the very strong influence of the number of machines.

The results for the job waiting time criterion displayed different patterns. The CDS algorithm tended to produce results for the small problems that were definitely worse than those for the RAES and AS algorithms. Its results were, on the average, about equal to those produced by the RGES algorithm. On the other hand, there was a close similarity between RAES' and AS' results. In the case of the large jobs, the RAES algorithm once more produced the best job waiting times. The next best results were yielded by the CDS algorithm while the RGES and AS algorithms had observations that were approximately the same. In its characteristic style, the values of the performance measure for the HD algorithm were inferior to those of the other algorithms.

The job waiting times increased as the number of jobs increased. A similar pattern was observed with an increase in the number of machines. The variation of the job waiting times with the number of jobs was more pronounced than with the number of machines.

In general, the RAES algorithm produced the best results for all three measures of performance over all of the problem sizes. When these values for the RAES algorithm were divided by the corresponding values of the same performance measure for another algorithm, the ratios obtained were found to be usually less than one. The average values of the ratios are displayed in Tables 5.3 and 5.4. This again illustrates that the RAES algorithm yielded the lower values for the performance measures.

The results for the small jobs are also expressed in terms of percentage deviation of the makespan values from the optimal values. They appear in Table 5.5. The optimal values were determined using an implicit enumeration technique [1]. No attempt was made to produce the optimal values for large problems because the implicit enumeration algorithms needed, on the average, 31.5 minutes to solve a 10 jobs x 10 machines problem. This is a considerable quantity of time when it is compared with a mean value of 0.039 seconds required by the CDS algorithm, which needs the least time. Clearly, even if sufficient funds were available, the necessary CPU time to solve even one 100x60 problem for its optimal solution would not be available on any academic or commercial computer; even if this time were available, the solution would not be known for several centuries hence.

The deviations again demonstrate the superior performance of the RAES algorithm. The RGES algorithm also produced good results with the smaller problem sizes. It actually produced optimal solutions for all thirty replications of the 4 jobs x 4 machines problems. This is explained by the fact that RGES randomly generates 25 different schedules, and one of the optimal schedules was

TABLE 5.3

RATIOS AND MEAN CPU TIMES FOR SMALL PROBLEMS

Number of Jobs	Number of Machines	Heuristic Number[a]	Average Ratios[b] Makespan	Machine Idle Time	Job Waiting Time	Average CPU Time[c]
4	4	1	1.001	.997	1.004	.041
		2	.930	.854	.962	.011
		3	N/A	N/A	N/A	.018
		4	.987	.971	1.002	.009
		5	.995	.991	1.012	.013
4	7	1	1.000	1.000	1.012	.052
		2	.946	.965	1.022	.017
		3	N/A	N/A	N/A	.024
		4	.999	1.010	1.066	.015
		5	.994	.993	1.046	.015
4	10	1	1.001	.993	1.009	.061
		2	.925	.899	.927	.019
		3	N/A	N/A	N/A	.030
		4	.995	.997	.995	.020
		5	.995	.985	1.000	.017
6	4	1	.995	.983	1.020	.067
		2	.885	.771	.963	.026
		3	N/A	N/A	N/A	.033
		4	.969	.984	.979	.011
		5	.981	.963	1.030	.018
6	7	1	.992	1.011	.971	.083
		2	.905	.884	.965	.026
		3	N/A	N/A	N/A	.046
		4	.983	.978	1.005	.016
		5	.993	.980	1.032	.023
6	10	1	.995	1.010	.994	.114
		2	.924	.907	.979	.036
		3	N/A	N/A	N/A	.063
		4	.989	.992	1.012	.023
		5	.983	1.002	.997	.027

[a] 1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

[b] Ratio of the performance measure for the particular heuristic to the similar value for the RAES heuristic.

[c] The CPU time is measured in seconds.

TABLE 5.3--Continued

Number of Jobs	Number of Machines	Heuristic Number[a]	Average Ratios[b] Makespan	Machine Idle Time	Job Waiting Time	Average CPU Time[c]
8	4	1	.977	.904	.969	.099
		2	.906	.758	.945	.040
		3	N/A	N/A	N/A	.051
		4	.980	.931	.958	.012
		5	.996	.983	1.022	.024
8	7	1	.972	.969	.956	.137
		2	.892	.858	.951	.047
		3	N/A	N/A	N/A	.071
		4	.982	.967	.968	.019
		5	.975	.954	.983	.032
8	10	1	.990	1.001	.943	.186
		2	.908	.877	.947	.054
		3	N/A	N/A	N/A	.119
		4	.989	.988	.970	.027
		5	.981	.985	1.026	.038
9	4	1	.979	.940	.989	.113
		2	.896	.743	.931	.083
		3	N/A	N/A	N/A	.057
		4	.978	.938	.971	.013
		5	.978	.929	1.011	.027
9	7	1	.979	.948	.972	.160
		2	.884	.809	.902	.111
		3	N/A	N/A	N/A	.091
		4	.971	.947	.982	.021
		5	.967	.937	1.022	.037
9	10	1	.977	.971	.975	.210
		2	.907	.883	.927	.120
		3	N/A	N/A	N/A	.129
		4	.984	.982	.984	.030
		5	.972	.956	1.003	.047
10	4	1	.979	.910	.990	.138
		2	.913	.736	.954	.099
		3	N/A	N/A	N/A	.066
		4	.973	.880	.952	.014
		5	.986	.899	1.025	.032

TABLE 5.3--Continued

Number of Jobs	Number of Machines	Heuristic Number[a]	Average Ratios[b] Makespan	Machine Idle Time	Job Waiting Time	Average CPU Time[c]
10	7	1	.978	.990	.959	.206
		2	.888	.815	.908	.121
		3	N/A	N/A	N/A	.109
		4	.984	.961	.962	.023
		5	.965	.955	1.010	.043
10	10	1	.976	.980	.936	.262
		2	.899	.850	.904	.144
		3	N/A	N/A	N/A	.146
		4	.987	.997	.958	.039
		5	.982	.956	.987	.067

TABLE 5.4

RATIOS AND MEAN CPU TIMES FOR LARGE PROBLEMS

Number of Jobs	Number of Machines	Heuristic Number[a]	Average Ratios[b] Makespan	Machine Idle Time	Job Waiting Time	Average CPU Time[c]
20	15	1	.960	.951	.900	1.781
		2	.855	.810	.929	.603
		3	N/A	N/A	N/A	1.352
		4	.974	.964	.947	.165
		5	.949	.911	.948	.335
20	30	1	.975	.986	.911	3.369
		2	.882	.895	.961	1.095
		3	N/A	N/A	N/A	2.977
		4	.988	.988	.962	.444
		5	.954	.944	.928	.554
20	45	1	.980	.992	.906	5.176
		2	.899	.924	.938	1.563
		3	N/A	N/A	N/A	4.380
		4	.995	.999	.973	.861
		5	.961	.970	.972	.814
20	60	1	.974	.978	.917	5.942
		2	.892	.920	.941	2.073
		3	N/A	N/A	N/A	6.915
		4	.992	.980	.980	1.406
		5	.960	.957	.961	1.094
40	15	1	.948	.915	.927	9.578
		2	.831	.740	.907	3.482
		3	N/A	N/A	N/A	9.184
		4	.966	.915	.945	.441
		5	.928	.848	.956	1.433
40	30	1	.962	.964	.916	17.927
		2	.854	.842	.901	6.496
		3	N/A	N/A	N/A	18.184
		4	.978	.970	.945	1.146
		5	.939	.909	.952	2.488

[a] 1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

[b] Ratio of the performance measure for the particular heuristic to the similar value for RAES heuristic.

[c] The CPU time is measured in seconds.

TABLE 5.4--Continued

Number of Jobs	Number of Machines	Heuristic Number[a]	Average Ratios[b] Makespan	Average Ratios[b] Machine Idle Time	Average Ratios[b] Job Waiting Time	Average CPU Time[c]
40	45	1	.969	.964	.921	27.017
		2	.878	.881	.927	9.458
		3	N/A	N/A	N/A	26.749
		4	.983	.973	.953	2.065
		5	.938	.923	.954	3.376
40	60	1	.974	.972	.934	36.178
		2	.892	.907	.929	12.257
		3	N/A	N/A	N/A	37.327
		4	.989	.985	.966	3.371
		5	.944	.941	.955	4.263
60	15	1	.940	.864	.936	24.311
		2	.827	.697	.918	10.788
		3	N/A	N/A	N/A	24.865
		4	.970	.909	.950	.877
		5	.924	.819	.961	3.244
60	30	1	.956	.936	.937	52.691
		2	.855	.820	.909	20.023
		3	N/A	N/A	N/A	55.581
		4	.981	.949	.960	2.160
		5	.920	.872	.939	6.297
60	45	1	.964	.958	.931	76.002
		2	.876	.874	.908	29.686
		3	N/A	N/A	N/A	86.619
		4	.982	.971	.955	3.913
		5	.933	.913	.935	9.159
60	60	1	.973	.977	.943	104.285
		2	.890	.904	.917	38.808
		3	N/A	N/A	N/A	114.137
		4	.989	.983	.969	5.976
		5	.937	.920	.941	10.475
80	15	1	.947	.863	.954	47.398
		2	.834	.675	.912	24.904
		3	N/A	N/A	N/A	51.503
		4	.976	.919	.966	1.394
		5	.922	.788	.961	5.806

TABLE 5.4--Continued

Number of Jobs	Number of Machines	Heuristic Number[a]	Average Ratios[b] Makespan	Machine Idle Time	Job Waiting Time	Average CPU Time[c]
80	30	1	.955	.930	.950	103.242
		2	.858	.822	.909	46.890
		3	N/A	N/A	N/A	121.925
		4	.980	.953	.966	3.399
		5	.926	.873	.949	11.102
80	45	1	.961	.950	.943	160.388
		2	.880	.876	.912	68.271
		3	N/A	N/A	N/A	176.084
		4	.982	.963	.969	5.965
		5	.930	.913	.938	16.259
80	60	1	.969	.967	.938	219.561
		2	.891	.900	.925	90.025
		3	N/A	N/A	N/A	219.251
		4	.987	.981	.968	9.073
		5	.939	.933	.950	22.246
100	15	1	.949	.850	.949	81.067
		2	.853	.683	.908	46.514
		3	N/A	N/A	N/A	82.166
		4	.978	.909	.959	2.112
		5	.923	.787	.956	8.869
100	30	1	.955	.918	.948	180.171
		2	.858	.825	.914	87.580
		3	N/A	N/A	N/A	191.545
		4	.980	.957	.967	5.032
		5	.917	.861	.945	17.825
100	45	1	.962	.943	.944	292.479
		2	.871	.862	.916	127.931
		3	N/A	N/A	N/A	319.051
		4	.985	.974	.969	8.584
		5	.924	.900	.943	26.525
100	60	1	.964	.950	.940	384.466
		2	.884	.876	.910	169.243
		3	N/A	N/A	N/A	461.304
		4	.981	.965	.961	12.876
		5	.926	.903	.930	37.001

TABLE 5.5

AVERAGE DEVIATIONS AND OPTIMAL SOLUTIONS FOR THE SMALL PROBLEMS

		4 Machines		7 Machines		10 Machines	
Number of Jobs	Heuristic[a] Number	Optimal Solutions	Average % Deviation	Optimal Solutions	Average % Deviation	Optimal Solutions	Average % Deviation
4	1	30	0.00	29	0.12	28	0.28
	2	5	9.41	10	5.95	2	9.51
	3	27	0.13	29	0.16	27	0.36
	4	20	1.54	27	0.29	23	0.93
	5	24	0.62	24	0.81	20	0.84
6	1	18	1.03	14	1.60	10	1.62
	2	1	13.96	1	12.86	0	11.35
	3	22	0.51	19	0.70	15	1.13
	4	11	3.96	9	2.57	8	2.27
	5	12	2.55	16	1.49	9	2.98
8	1	4	4.67	1	4.94	2	3.38
	2	1	16.49	0	16.27	0	15.03
	3	14	2.14	10	1.92	6	2.26
	4	8	4.39	2	3.85	1	3.43
	5	10	2.69	2	4.60	2	4.31
9	1	1	4.67	1	4.39	3	5.11
	2	0	17.03	0	16.94	0	14.02
	3	8	2.32	6	2.14	3	2.60
	4	3	4.77	1	5.23	3	4.28
	5	3	4.82	3	5.77	0	5.66
10	1	2	3.60	0	5.74	1	6.52
	2	0	14.62	0	18.57	0	17.03
	3	13	1.36	3	3.28	1	3.85
	4	8	4.29	2	5.06	0	5.26
	5	8	2.98	0	7.20	0	5.86

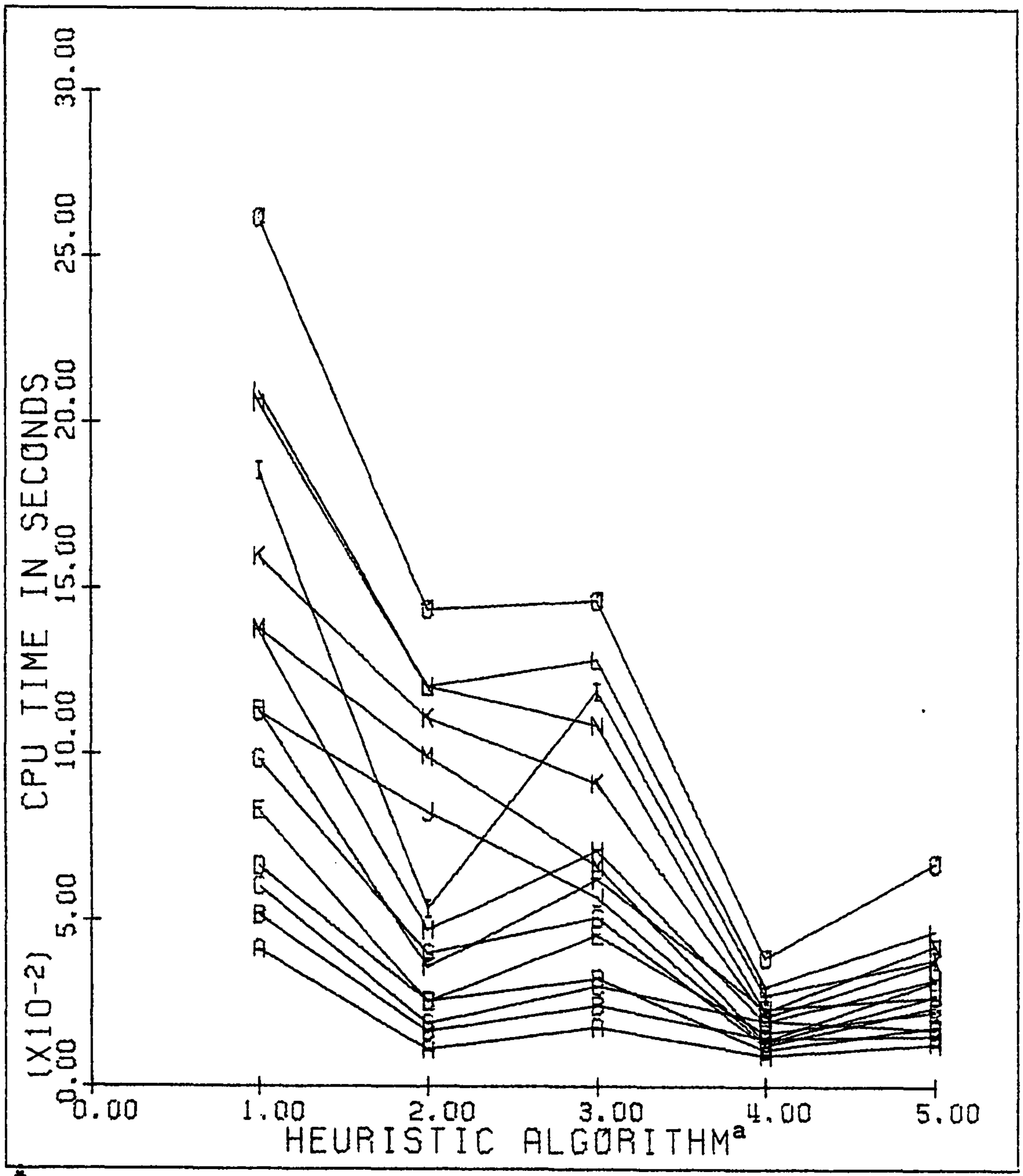
[a]1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

among them. The chances of this occurring are quite substantial since there are only 4! or 24 possible schedules for a 4 jobs flowshop problem. The largest deviations are associated with the HD algorithm.

The mean CPU times are plotted in Figures 5.7 and 5.8 using linear scales. Due to the concentration of the lines at the smaller CPU times, these means are again plotted in Figures 5.9 and 5.10 using logarithmic scales. The plots indicate that there is a large variation in the CPU times required by the heuristic algorithms. For the small problems, the CDS algorithm needed the least amount of time to arrive at schedules while the RGES algorithm needed the most time. The AS algorithm was also very economical with time. The large amount of time required by the RGES algorithm is due to its examination of several schedules. In the initial phase, 25 schedules were randomly generated and evaluated. This consumed a fair amount of time. The best of the 25 schedules was selected, and attempts were made to improve the schedule by interchanging neighboring jobs. This pair exchange technique required a considerable quantity of CPU time. At the other extreme, the CDS algorithm, which needed the least time, converts the problem into (m-1) two-machine problems which individually require very little time to be solved. The most attractive heuristic algorithm, RAES, consumed a relatively large amount of time because of the pair exchange routine employed in its second phase.

FIGURE 5.7

CPU TIME MEANS, SMALL PROBLEMS

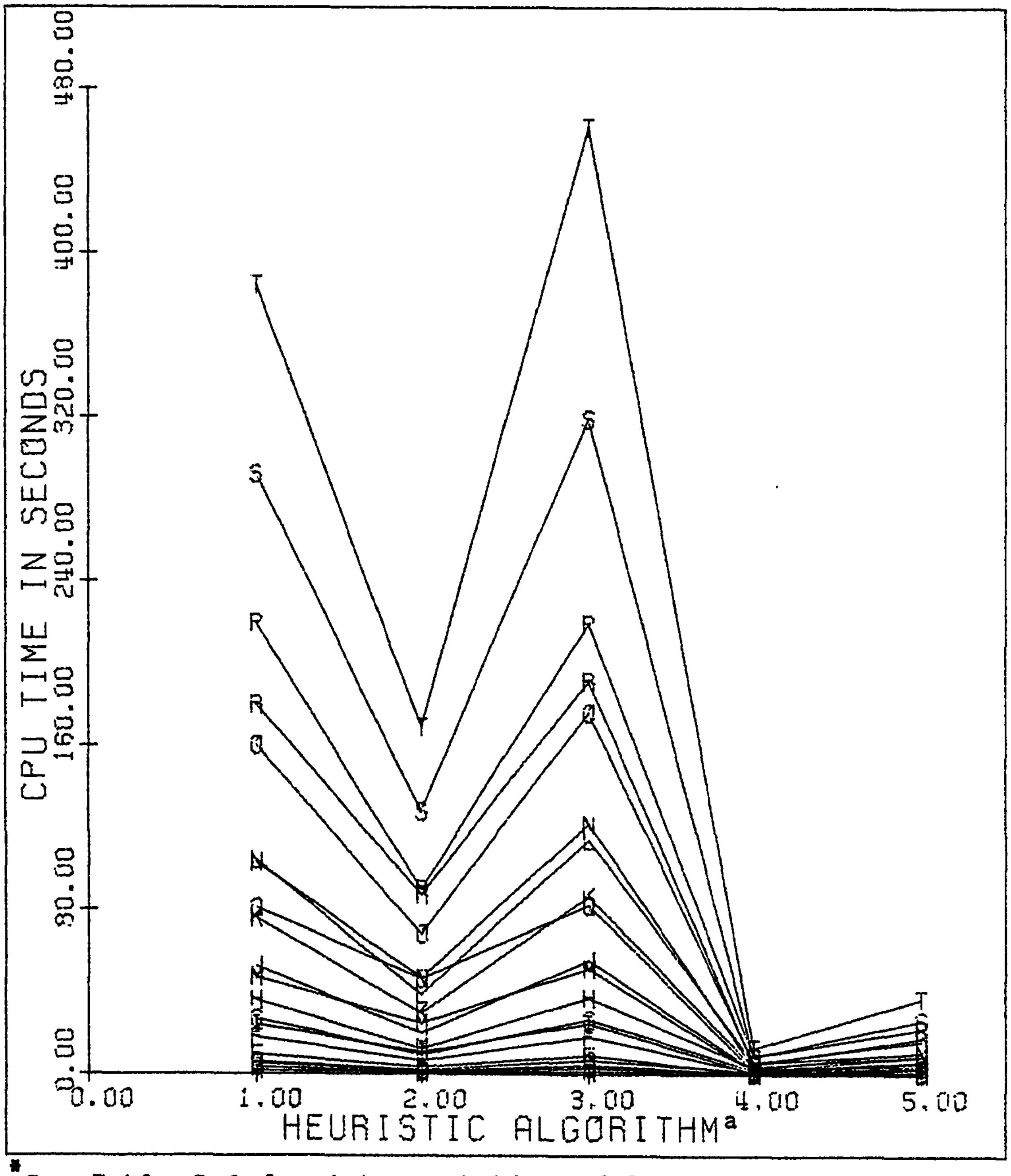


*See Table 5.1 for interpretation of letter codes

[a] 1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.8

CPU TIME MEANS, LARGE PROBLEMS

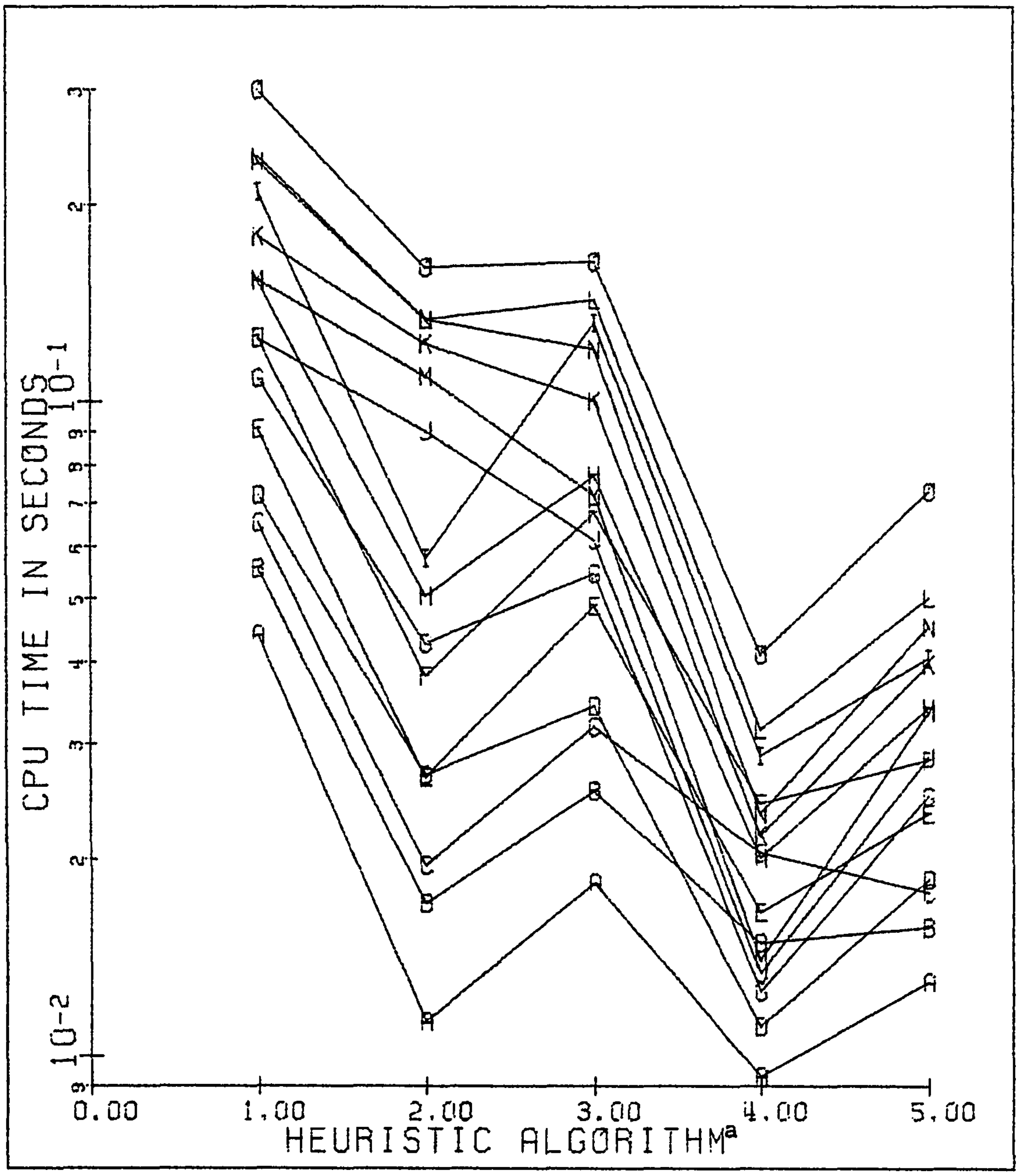


*See Table 5.2 for interpretation of letter codes

^{a}1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.9

CPU TIME MEANS, SMALL PROBLEMS

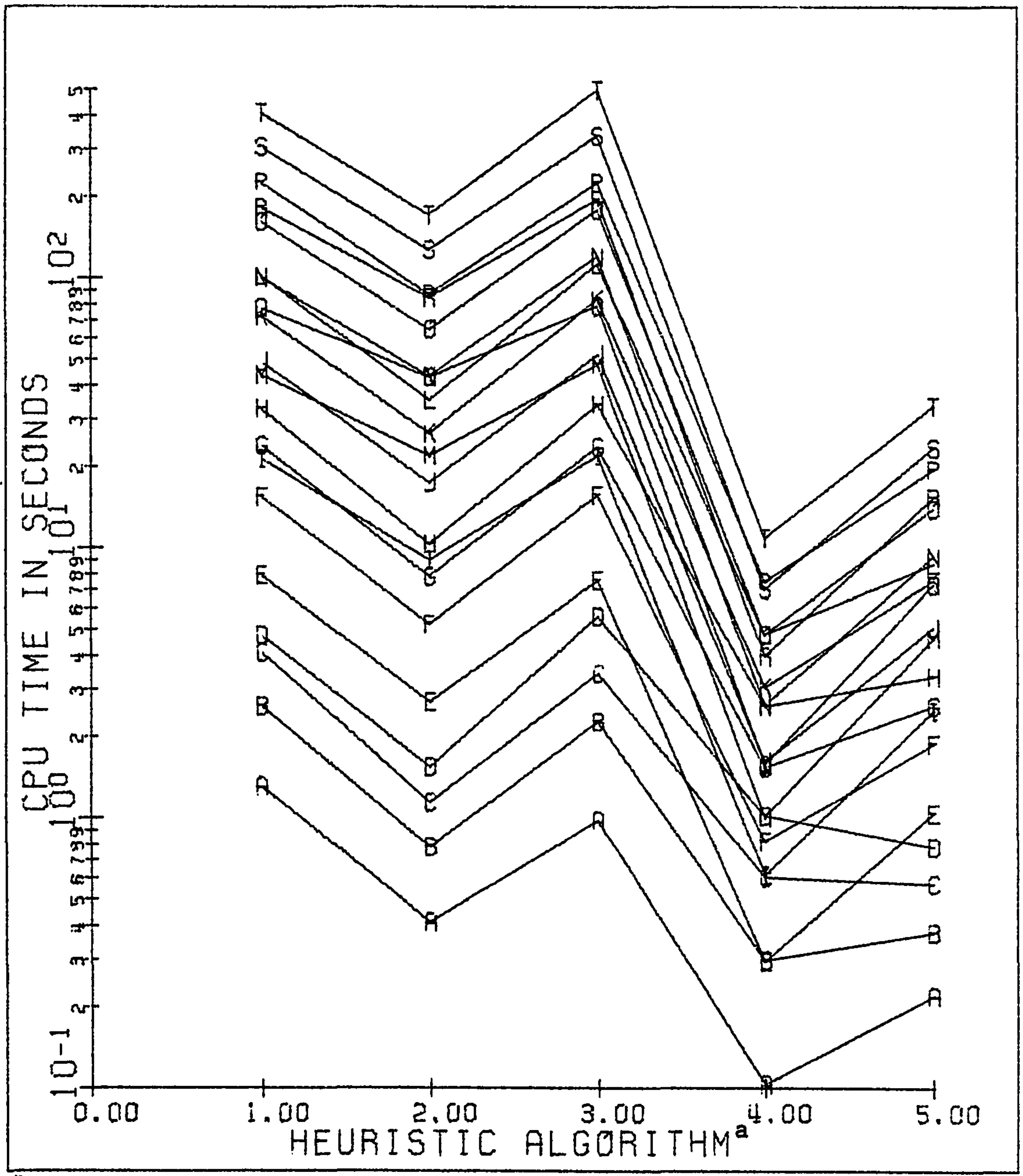


*See Table 5.1 for interpretation of letter codes

[a]1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.10

CPU TIME MEANS, LARGE PROBLEMS



*See Table 5.2 for interpretation of letter codes

^a 1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

Although the RGES algorithm was the largest consumer of time in solving the small jobs, its time needs were considerably less than that of the implicit enumeration technique which produced the optimal values of the measures of performance. As an example, this latter technique required, on the average, 31.5 minutes to solve the 10x10 problem while the RGES algorithm required 0.262 seconds.

When the large problems were considered, a new pattern emerged. The RAES heuristic algorithm replaced RGES as the largest consumer of time. In fact, this algorithm appears to be increasing its demand for CPU time at an increasing rate as compared with the other algorithms. The same can be said for the RGES algorithm, but to a smaller degree. It also appears that an increase in the number of machines demands a correspondingly higher increase in CPU time than is needed for an equivalent increase in the number of jobs.

The results of regression analyses, relating the CPU time to the number of jobs and number of machines, will be reported for both the small and large problems later in this chapter.

The Influence of Algorithm Design on Performance

The differences in the results of the heuristic algorithms are due mainly to their designs. The RGES algorithm provided its best results at the smallest problem sizes because the chances of including the better schedules in the 25 randomly generated schedules are greatest at the smallest problem sizes where the number of possible schedules is small.

The HD algorithm produced the worst results. It is believed that the job-pairing procedure is responsible for this performance. As shown by Gupta and Maykut [58], this procedure builds schedules by selecting jobs which minimize the idle time of the last machine. This algorithm thus creates much job waiting time because it delays jobs which do not minimize the last machine's idle time. The jobs which were delayed eventually contributed to the machine idle time when they were processed. The causes of the machine idle time also contribute to an increase in the makespan value.

The RAES algorithm creates artificial two-machine problems from m-machine problems by calculating new processing times which are weighted sums of the original processing times. Johnson's algorithm [65] is then applied to the artificial two-machine problem to arrive at a schedule which was subjected to an improvement routine. Although Johnson's algorithm was specifically designed to give optimal makespan solutions to the two-machine problem with actual processing times, it also gave fairly good results in the initial phase of the RAES algorithm using these pseudo processing times. The CDS algorithm employed a similar process of forming artificial two-machine problems and solving them with Johnson's algorithm. The major differences between these two algorithms are that the CDS algorithm produced multiple schedules from which the best is chosen and the RAES algorithm has an improvement routine included. Both of these algorithms produced the best results. Thus, it would be reasonable to conclude that the features they had in common contributed to their good performances and the improvement routine enabled the RAES algorithm to surpass the CDS algorithm.

The underlying logic of the AS algorithm is the scheduling of jobs on machines early in the sequence in increasing order of their processing times, and vice versa for machines late in the sequence. This algorithm also incorporates an improvement routine [1]. The scheduling of jobs according to the increasing order of their processing times is intuitively ideal for minimizing the makespan value.

All of these heuristic algorithms were designed to minimize the makespan values. Nothing can be said about their abilities to minimize the other measures of performance because minimum values of these measures were not calculated. Any comments would therefore be limited to relative comparisons between the values produced by the algorithms.

Multivariate Analysis

The data generated from the flowshop simulations were subjected to a number of multivariate tests. These included multivariate analysis of variance, multiple discriminant analysis and multivariate regression analysis. Of primary importance were the multivariate analysis of variance tests since they served as a means of evaluating a number of the hypotheses.

Multivariate Tests of the Hypotheses

From a review of the observations, it appears that the RAES algorithm produced the smallest values for each performance measure, which were considered the best results. Furthermore, Dannenbring [37] found that RAES yielded the best results in his earlier experiments. Consequently, the tests of the first three hypotheses were performed in such a manner that comparisons were made between the RAES algorithm and the other algorithms. This was accomplished in the design of the matrices which contained the contrasts.

The first three hypotheses were tested with the aid of the FRMLM computer program written by Carlson and Timm [28]. A description of the results of these tests follows.

<u>The interaction hypothesis (Hypothesis 1)</u>. This hypothesis postulated that there is no interaction effect between the problem sizes and the heuristic algorithms. Using the F-statistics obtained from the data collected, this hypothesis was rejected for both the small and large problems. The F-statistics were derived from the Wilks' Lambda criterion using the conversion technique discussed in Chapter 4. These statistics, as well as others derived from the data, appear in Tables 5.6 and 5.7, for the small and large problems, respectively. Both F-statistics indicate

TABLE 5.6

TESTS OF SIGNIFICANCE, SMALL PROBLEMS

	Effect		
	Interaction	Heuristic	Problem Size
Wilks' Lambda	.391	.136	.001
parameters	12,14,435	12,15,435	15,14,435
F approximation	2.511	5.259	20.359
degrees of freedom	168,3914	180,4024	210,4364
probability	p<.0001	p<.0001	p<.0001
Lawley-Hotelling Trace	1.081	3.386	49.901
parameters (s,m,n)	12,.5,211	12,1,211	14,0,209
F approximation	2.716	7.941	99.361
degrees of freedom	168,5066	180,5066	210,5854
probability	p<.0001	p<.0001	p<.0001
Pillai's Trace	.830	1.374	2.475
parameters (s,m,n)	12,.5,211	12,1,211	14,0,209
F approximation	2.245	3.646	5.999
degrees of freedom	168,5220	180,5220	210,6062
probability	p<.0001	p<.0001	p<.0001

TABLE 5.7

TESTS OF SIGNIFICANCE, LARGE PROBLEMS

	Effect		
	Interaction	Heuristic	Problem Size
Wilks' Lambda	.023	.002	.000
parameters	12,19,580	12,20,580	15,19,580
F approximation	11.402	21.046	92.686
degrees of freedom	228,5831	240,5913	285,6734
probability	$p<.001$	$p<.001$	$p<.001$
Lawley-Hotelling Trace	8.267	43.242	3311.842
parameters (s,m,n)	12,3,284	12,3.5,284	15,1.5,282
F approximation	20.601	102.369	6555.510
degrees of freedom	228,6818	240,6818	285,8462
probability	$p<.001$	$p<.001$	$p<.001$
Pillai's Trace	2.191	2.705	3.557
parameters (s,m,n)	12,3,284	12,3.5,284	15,1.5,282
F approximation	6.689	8.279	9.244
degrees of freedom	228,6972	240,6972	285,8700
probability	$p<.001$	$p<.001$	$p<.001$

that there is significant interaction at the .001 level. F-statistics were also derived from the Lawley-Hotelling trace and the Pillai trace criteria using the conversion equations provided on p. 143 of [18]. These derivations were necessary because tables for the critical values of both criteria were not readily available for the ranges of the parameters employed. These F-statistics also indicate that there is significant interaction between the two factors at the same level of significance.

This significant interaction suggests that some heuristic algorithms were more efficient at certain problem sizes. In terms of profile analysis, these results state that parallel profiles do not exist. A superficial examination of the plots of the means of the dependent variables (or performance measures) for the small problems revealed that the lines connecting the points for the various problem sizes are roughly parallel. When the graphs in Figure 5.1 were thoroughly scrutinized, it was discovered that, for the makespan means, the line labelled F (which represents problem size, 6 jobs x 10 machines) intersects the line labelled K (which represents problem size, 8 jobs x 7 machines). On further examination, it was observed that the line labelled A is not parallel to the line labelled D; neither is the line labelled B parallel to the line labelled G. On the graphs of the job waiting means in Figure 5.2, the lines labelled K and M intersect. Also, it is clear that lines I and J are not parallel. The lines labelled I and L intersect on the graphs of the machine idle time means in Figure 5.3. The line labelled B is clearly not parallel to the neighboring lines labelled E, H, K and N. The examples mentioned are not an exhaustive account of the lines which are not parallel. There are many other examples. The deviations from parallelism are, however, very small and could only be detected by keen observation or by a very sensitive test. The multivariate analysis of variance test for interaction detected the lack of parallelism. The high sensitivity of this test was due to the large number of degrees of freedom which were derived from the number of levels in the factors and the number of observations.

There was also a significant interaction effect for the large problems. The multivariate F-statistic obtained for the test was significant at the .001 level. The graphs of the means of the performance measures (or dependent variables) exhibited less variation than those for the small problems. The intersection of lines occurred only on the graphs of the makespan means in Figure 5.4. Nevertheless, the test with its large number of degrees of freedom was sensitive enough to identify the lack of parallelism.

A significant interaction demands that caution must be exercised in testing the remaining hypotheses. A repeated univariate approach should be cautiously interpreted because this technique does not take into consideration the fact that the factors are confounded. A multivariate approach has therefore been adopted for tests of differences in heuristics, and in the problem sizes. This approach was described in Chapter 4.

The test of differences in heuristic algorithms (Hypothesis 2). This hypothesis postulated that the heuristic algorithms produce schedules which are equally efficient. The results for the small jobs rejected this hypothesis. They yielded F-statistics, derived from all three criteria, which were significant at the .001 level. Similar results were obtained for the large jobs. These results for the small and large problems are displayed in Tables 5.6 and 5.7, respectively.

From the graphs of the means of the dependent variables, it is clear that there is no equality of performances among the heuristic algorithms. The HD algorithm clearly produced results which were inferior to those of other algorithms. Its observations on the dependent variables were always the largest. The lines joining the means of the observations for the various algorithms have wavy patterns. This suggests that the average values varied among the algorithms. A horizontal line would indicate perfect equality. At this stage, it seems that in addition to the HD algorithm, both the AS and RGES algorithms contributed to the variation. With a few exceptions, it appears that the results for the RAES and CDS algorithms were quite close. A thorough identification of the factors causing this lack of equality will be done in the post hoc tests.

Testing of the differences in the problem size (Hypothesis 3). As anticipated, there were significant differences in the levels of this factor. Again, due to the limited range of the tables of the various distributions available, the test of significance was limited to the derived F-statistics. The respective values of the F-statistics were significant with a probability value of less than .001. These results for both the small and large problems are displayed in Tables 5.6 and 5.7, respectively.

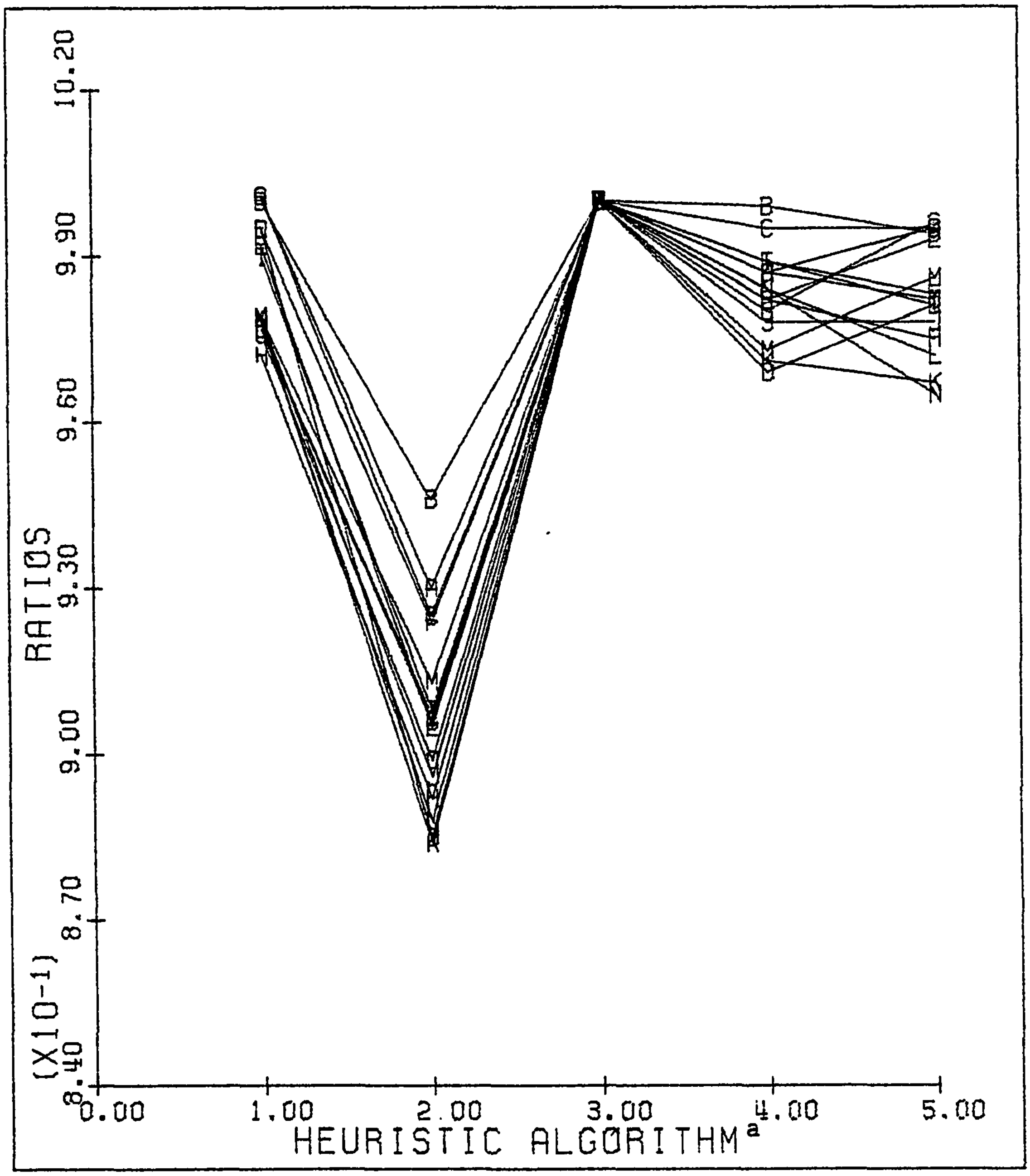
The plots of the means for each dependent variable at each problem size are particularly deceptive when used to assist in any interpretation of the differences in results. This difficulty is due to the scale which was selected to accommodate all of the problems, defined as being large, in the same figure. For example, the ratio of the job waiting time value for the RAES algorithm to the same value for the RGES algorithm at the 20 jobs x 15 machines problem size is .900. An examination of the graph reveals that the line joining the values for these algorithms appears to be a horizontal straight line which erroneously suggests that the algorithms have equal job waiting times. Hence, the ratios in Tables 5.3 and 5.4 would be more reliable than the graphs of the mean values in providing information. From these tables, it appears that at the smallest problem sizes, the ratios are high. As the problem sizes become larger, the ratio becomes smaller, indicating that the efficiency of the other algorithms declined with respect to the efficiency of the RAES algorithm. These ratios are plotted in Figures 5.11 to 5.16.

Comparing the computational times (Hypothesis 4). This hypothesis states that the heuristic algorithms need an equal amount of computational effort to arrive at schedules. In this research study, the CPU time required to perform the computations was used as a measure of the computational effort. This hypothesis was rejected for both the small and large problems. The analysis of the data was done using the MANOVA program distributed by the CLYDE computing services [4], and an experimental design described by Nicewander [82] for repeated measures with one dependent variable. Although only one dependent variable is involved in this analysis, a multivariate approach is taken since the use of a univariate approach requires that the assumption of compound symmetry be met [108]. The values of the resulting F-statistics for both the large and small problems led to the rejection of the hypothesis since they exceeded the critical values for a significance level of .001.

When the tabulated average CPU times in Tables 5.3 and 5.4 and the graphs of them in Figures 5.7 and 5.8 were examined, it was very clear that the CPU times required by

FIGURE 5.11

MAKESPAN RATIOS, SMALL PROBLEMS

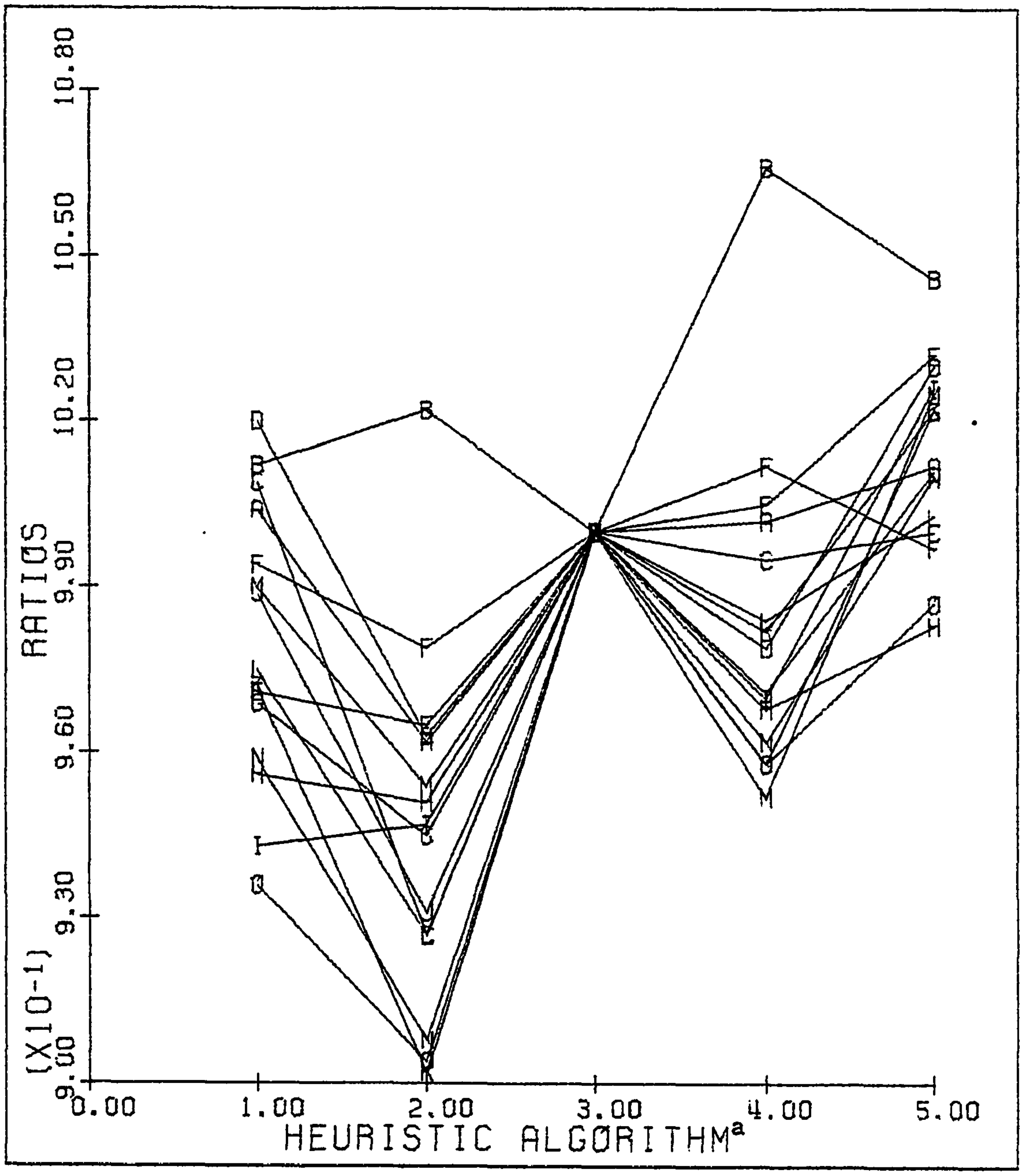


*See Table 5.1 for interpretation of letter codes

^a 1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.12

JOB WAITING TIME RATIOS, SMALL PROBLEMS

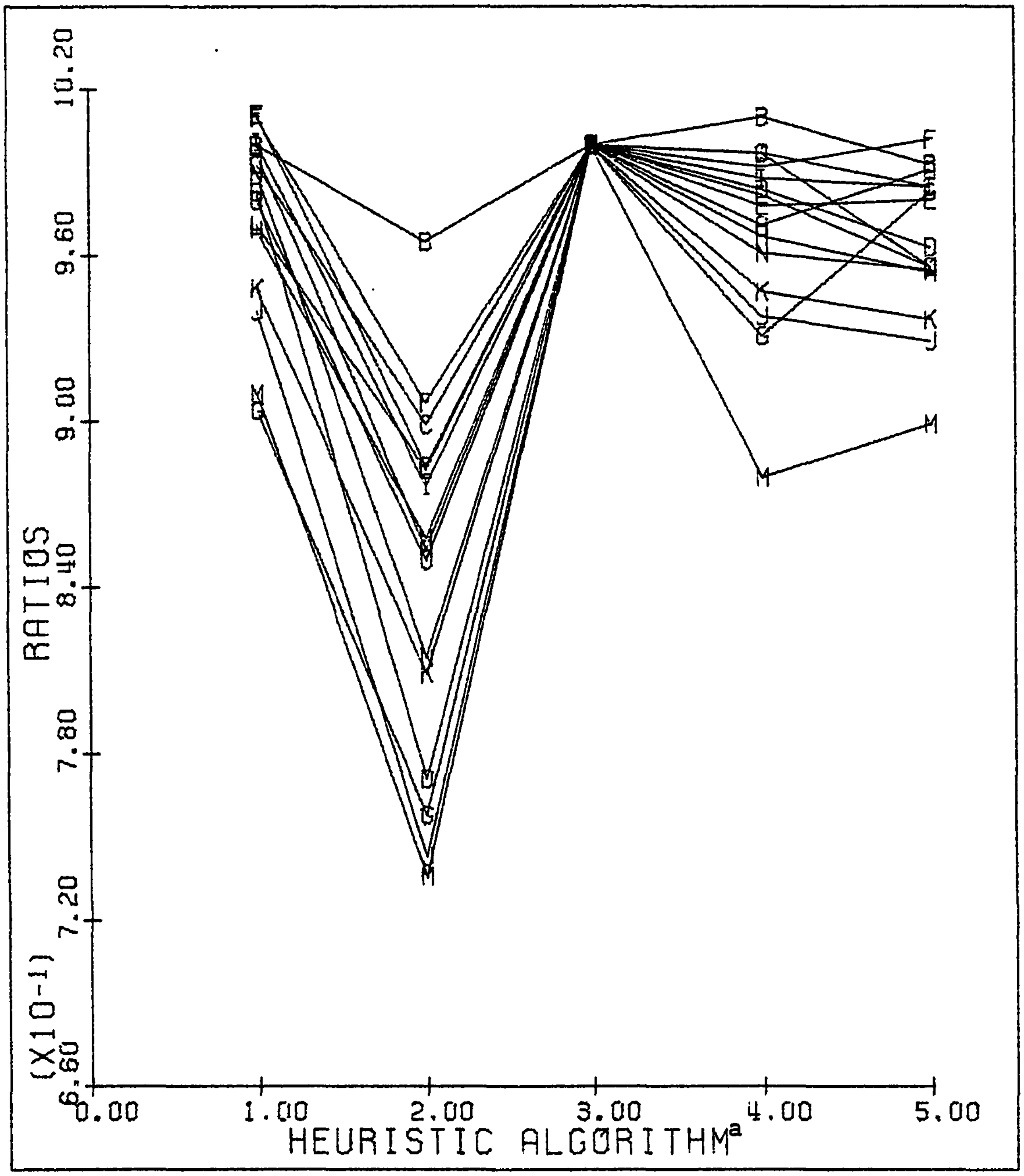


*See Table 5.1 for interpretation of letter codes

[a] 1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.13

MACHINE IDLE TIME RATIOS, SMALL PROBLEMS

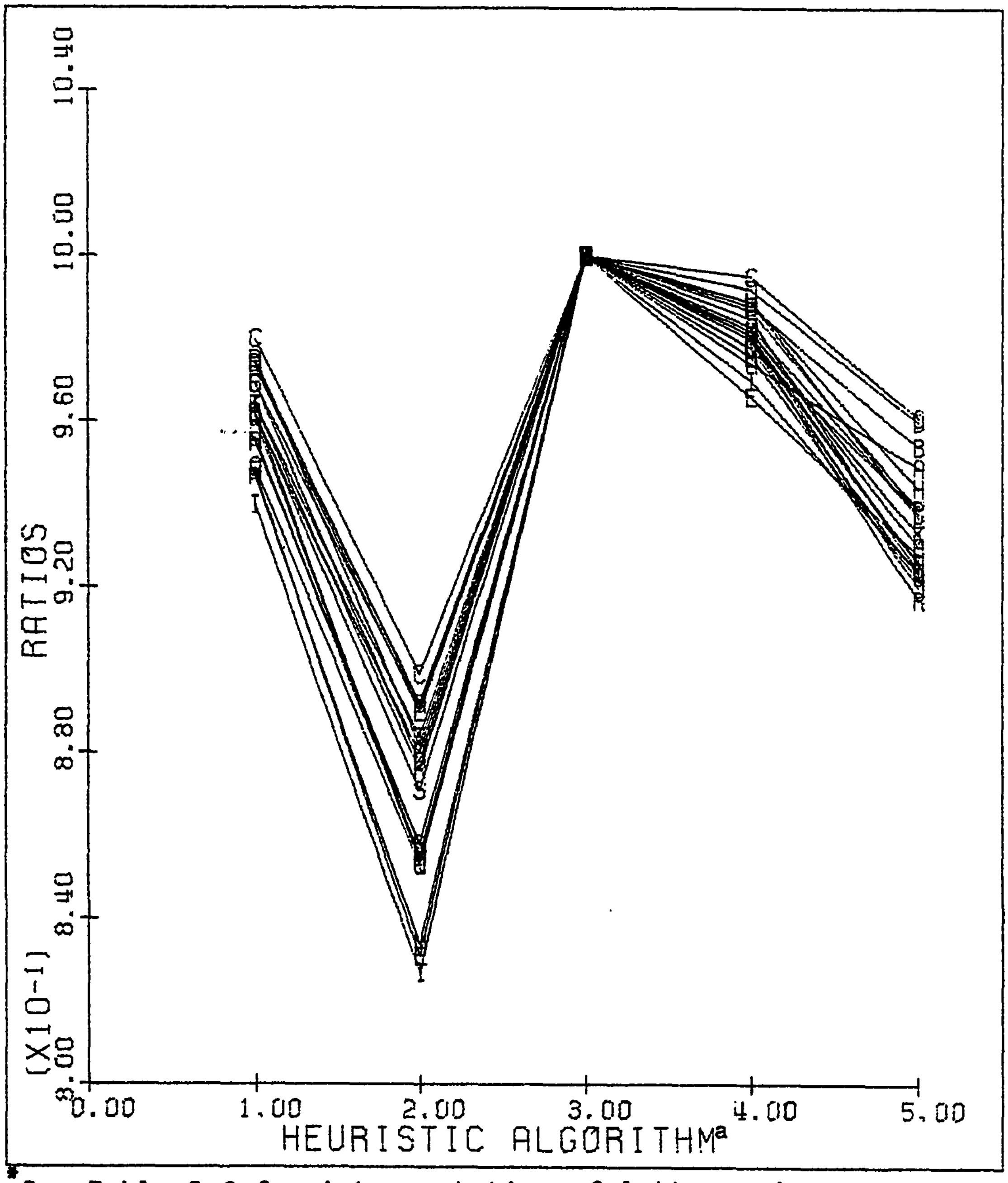


*See Table 5.1 for interpretation of letter codes

[a]1-RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.14

MAKESPAN RATIOS, LARGE PROBLEMS

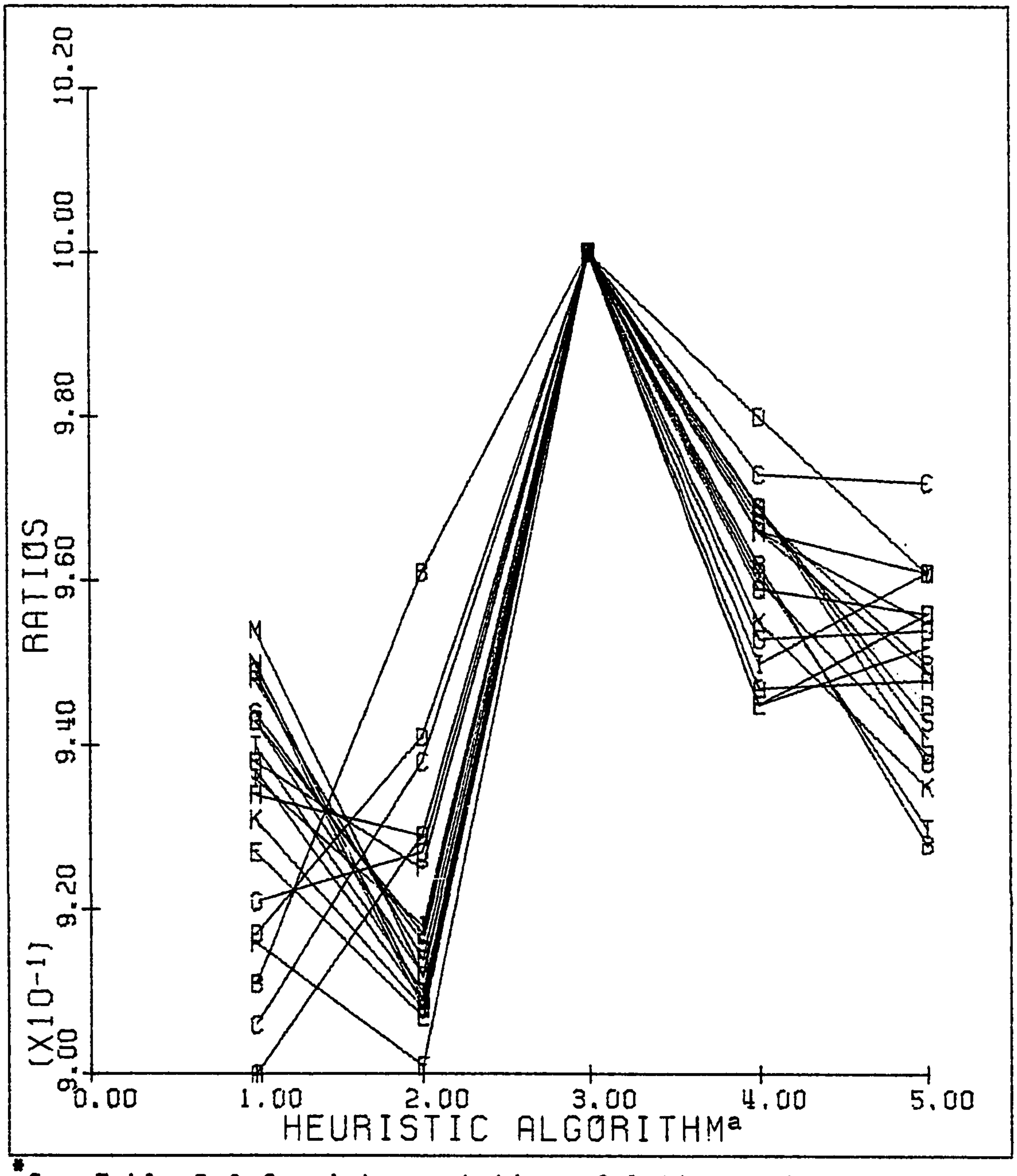


*See Table 5.2 for interpretation of letter codes

[a]1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.15

JOB WAITING TIME RATIOS, LARGE PROBLEMS

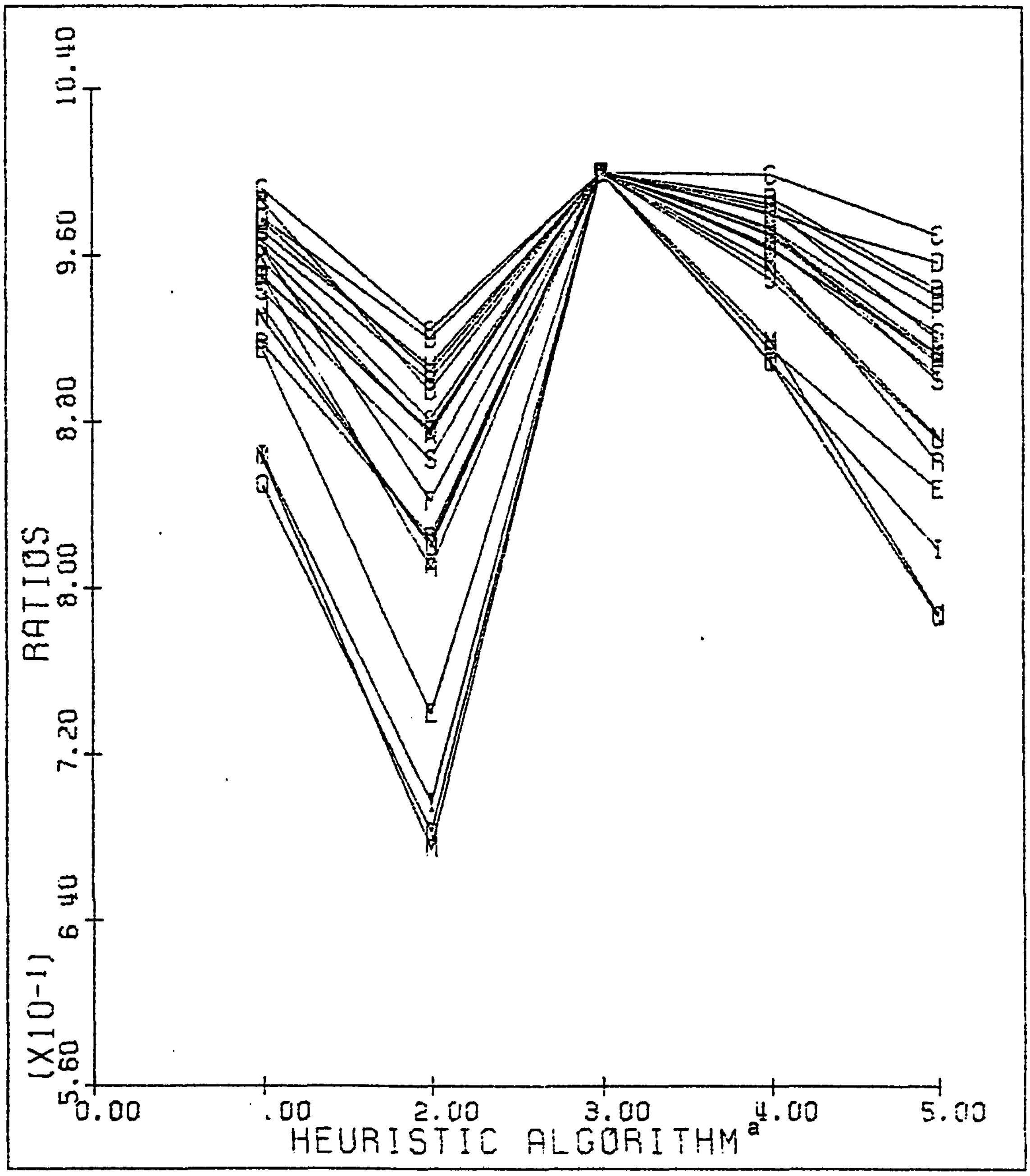


*See Table 5.2 for interpretation of letter codes

a1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

FIGURE 5.16

MACHINE IDLE TIME RATIOS, LARGE PROBLEMS



[*]See Table 5.2 for interpretation of letter codes

[a]1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

the algorithms were definitely different. As the problem sizes grew larger, the differences increased--especially between the algorithms which needed the most and the least CPU time.

Univariate Comparison of the Heuristic Algorithms

In this section, the heuristic algorithms are compared, taking each dependent variable individually. Univariate F-statistics, provided by the FRMLM program [28] as a by-product, are used to perform these comparisons. The comparisons take the form of comparing each heuristic algorithm with the RAES algorithm. Thus, four comparisons are done and an F-statistic is reported for each one.

Univariate Tests of The Hypotheses

Makespan as the criterion of performance (Hypothesis 5). From the results in Tables 5.8 and 5.9, it can be concluded that each heuristic algorithm produced makespan values which were significantly larger than those produced by the RAES algorithm.

TABLE 5.8

MAKESPAN COMPARISONS, SMALL PROBLEMS

Comparing RAES With	F-Value	p-Value	Degrees of Freedom
RGES	5.718	P<.0001	15,435
HD	45.707	P<.0001	15,435
CDS	5.225	P<.0001	15,435
AS	5.637	P<.0001	15,435

TABLE 5.9

MAKESPAN COMPARISONS, LARGE PROBLEMS

Comparing RAES With	F-Value	p-Value	Degrees of Freedom
RGES	150.8	P<.0001	20,580
HD	904.0	P<.0001	20,580
CDS	52.0	P<.0001	20,580
AS	315.8	P<.0001	20,580

These results agree with the findings which were reported by Dannenbring [37] in those cases where the two studies have comparisons in common. The patterns in the graphs of the makespan means also confirm these findings, but with a lesser degree of certainty. This is especially true when making a distinction between the results for the CDS and RAES algorithms.

<u>Machine idle time as a criterion of performance (Hypothesis 6)</u>. The results from the test of this hypothesis are displayed in Table 5.10 and 5.11 for the small and large problems, respectively.

TABLE 5.10

MACHINE IDLE TIME COMPARISONS, SMALL PROBLEMS

Comparing RAES With	F-Value	p-Value	Degrees of Freedom
RGES	2.793	.0004	15,435
HD	37.874	P<.0001	15,435
CDS	2.624	.0008	15,435
AS	6.005	P<.0001	15,435

TABLE 5.11

MACHINE IDLE TIME COMPARISONS, LARGE PROBLEMS

Comparing RAES With	F-Value	p-Value	Degrees of Freedom
RGES	40.4	P<.0001	20,580
HD	278.6	P<.0001	20,580
CDS	22.7	P<.0001	20,580
AS	150.2	P<.0001	20,580

They reveal that the other heuristic algorithms produced machine idle times that were significantly larger than those of RAES for both the small and large problems. The probabilities that these differences occurred by chance are extremely small, the largest being .0008. These results substantiated the information that was concluded from the plots of the machine idle time means, which did not produce horizontal lines. This evidence was also supplied in a different form by the ratios between the RAES algorithm and the other algorithms, which are recorded in Tables 5.3 and 5.4 and plotted in Figures 5.13 and 5.16. In general, very few of the ratios were close to 1.0.

<u>Job waiting time as the criterion of performance (Hypothesis 7)</u>. The results of this analysis appear in Tables 5.12 and 5.13 for the small and large problems, respectively.

TABLE 5.12

JOB WAITING COMPARISONS, SMALL PROBLEMS

Comparing RAES With.	F-Value	p-Value	Degrees of Freedom
RAES	4.719	P<.0001	15,435
HD	8.054	P<.0001	15,435
CDS	4.543	P<.0001	15,435
AS	0.759	P=.7238	15,435

TABLE 5.13

JOB WAITING COMPARISONS, LARGE PROBLEMS

Comparing RAES With	F-Value	p-Value	Degrees of Freedom
RGES	131.5	P<.0001	20,580
HD	251.8	P<.0001	20,580
CDS	80.1	P<.0001	20,580
AS	129.1	P<.0001	20,580

When the large problems are considered, all of the other heuristic algorithms produced job waiting times which were significantly larger than those yielded by the RAES algorithm. The probability that this occurred by chance is less than .0001. The same can be said for the small problems except in the case of the AS algorithm. The F-value calculated in the comparison with this latter algorithm was .759 with 15 and 435 degrees of freedom. When this was compared with the F-distribution value, it was concluded that the difference between the results of the AS and RAES algorithm was not significant since there was a probability of .7238 that this difference occurred by chance. This result is corroborated by the data in Table 5.3 where the ratios between the two algorithms are very close to 1.0. The plots of the means on the graphs in Figure 5.2 for the small problems also concur with this finding. The wavy nature of the remainder of the graphs in the same figure indicates that the remaining algorithms produced results that differed.

Non-statistical Tests

Hypothesis 8. This hypothesis states that the heuristic algorithms produce an equal number of optimal schedules. Hypothesis 8 is limited to the small problems since it was not feasible to determine optimal schedules for the large problems. The number of optimal schedules yielded by each algorithm for each problem size is recorded in Table 5.5. It can be concluded that the likelihood of producing an optimal schedule decreases with an increase in the problem size. This is expected because the larger the problem size, the larger the number of possible schedules, and thus, the smaller the probability of an optimal schedule being yielded by a method which does not consider all possible schedules.

The aggregate number of optimal schedules over all of the problem sizes are recorded in Table 5.14.

TABLE 5.14

THE AGGREGATE NUMBER OF OPTIMAL SCHEDULES

Heuristic Algorithm	Number
RGES	144
HD	20
RAES	203
CDS	126
AS	133

In keeping with all of the other comparisons, the RAES algorithm determined the largest number of optimal schedules while the HD algorithm produced the least. The numbers determined by RGES were greater than those determined by CDS and AS in that order, but there was not a very substantial difference between the quantities produced by these three algorithms.

Post Hoc Analysis

When a hypothesis is rejected using a multivariate analysis of variance technique, further investigation can be done to determine which dependent variables contributed to the rejection. In this research study, generalized discriminant analysis was used to perform this task. The correlation between the derived variable and the generalized discriminant functions, called discriminant or cannonical variates, was used to identify the dependent variables most responsible for the rejection of the hypothesis. Cannonical variates are those linear combinations of dependent variables which are most sensitive to deviations from the null hypothesis [28]. They maximize the variation in the hypothesis sums of squares Qh relative to the error sums of squares Qe [109]. They were obtained through eigenvectors from the determinantal equation which was defined in Equation 4.9. The coefficients of the dependent variables in the cannonical variates are called standardized discriminant coefficients when they are calculated under the condition that the within group variance of the discriminant scores is unity [109]. Thus, the coefficients of the variables appear in comparable units which permits statements to be made about their relative influence on dependent variables [66].

The number of cannonical variates were the same as the number of roots obtained from the determinantal equation. Bartlett's chi-squared statistic [28] was used to test each cannonical variate for its statistical significance with respect to its ability to discriminate.

The discriminant analysis was performed using derived variables X_{ij}'s, which are defined in Table 5.15, where each X_{ij} represents the difference between the RAES and another algorithm for one of the original dependent variables.

TABLE 5.15

DERIVED VARIABLES

	Original Dependent Variable		
RAES vs.	Makespan	Job Waiting Time	Machine Idle Time
RGES	X1	X5	X9
HD	X2	X6	X10
CDS	X3	X7	X11
AS	X4	X8	X12

The discriminant analysis technique used in this study is due to Bargmann and Porebski and is described in Timm [109]. In this approach, the correlation between the derived variable and the discriminant function is used as an indication of the ability of that variable to cause a rejection of the hypothesis. There is a direct relationship between the strength of the correlation and the ability to cause the rejection.

This technique is preferred over those which use the values of the standardized discriminant coefficients because any correlation among the dependent variables affects the interpretation of these standardized discriminant coefficients [66,109].

The Interaction Hypothesis

It was reported earlier in this chapter that the hypotheses postulating no interaction between the factors were rejected. The results from the discriminant analyses of the data were used to explore the cause of the rejections.

Small problems. The coefficients and correlations for the discriminant functions, which were significant according to Bartlett's chi-squared statistic, are recorded in Table 5.16. Functions with probabilities greater than .05 were considered not significant; therefore, only two functions are reported.

TABLE 5.16

DISCRIMINANT ANALYSIS, SMALL PROBLEMS

Derived Variable	Standardized Coefficients Function 1	Standardized Coefficients Function 2	Correlations Function 1	Correlations Function 2
X1	.001	.012	.174	.507
X2	-.002	.015	.437	.539
X3	-.015	-.014	.035	.348
X4	.003	-.007	.186	.337
X5	.002	.000	.316	.319
X6	-.000	.001	.263	.465
X7	.000	.002	.136	.485
X8	-.000	-.003	.068	.034
X9	-.004	-.000	.088	.369
X10	.004	-.003	.745	.137
X11	-.001	.002	.076	.283
X12	.001	.003	.276	.340

	Function 1	Function 2
Chi-squared	409.1	216.1
df	168	143
Probability	p<.0001	p<.0001
% Variance	51.6%	17.9%

The interpretation of the results is done using the Bargmann and the Porebski approaches reported by Timm [109].

From the first function, which accounted for 51.6% of the variance, the largest correlations were .745, .437, .316, .276 and .263, which were associated with the derived variables X10, X2, X5, X12 and X6. According to Table 5.15, these were formed from HD, HD, RGES, AS and HD algorithms, respectively. Thus, the variation between RAES and HD for all three original dependent variables contributed to the rejection of the interaction hypothesis. The variation between RAES and RGES for job waiting time and RAES and AS for machine idle time also contributed to the rejection. However, the HD algorithm was the most prominent contributor to the rejection since the two largest correlations are associated with it.

The second discriminant function is associated with the roots of the determinantal equation remaining after the largest root is eliminated. The correlations between each variable and the discriminant function indicate that the variables X2 and X1 were the best discriminators. The X2 variable was derived from the differences between the HD and RAES algorithms, while X1 was derived from RAES' difference with RGES. In both cases, the original dependent variable was makespan.

Large Problems. The standardized discriminant coefficients and the correlations between the discriminant function and each derived variable are reported in Table 5.17 for the large problems. Although Bartlett's chi-squared statistic indicated that the first four variates are significant at a 0.05 significance level it was decided to report only the first two because the third and fourth accounted for only 5.4% and 1.2% of the variation.

TABLE 5.17

DISCRIMINANT ANALYSIS, LARGE PROBLEMS

Derived Variable	Standardized Coefficients Function 1	Standardized Coefficients Function 2	Correlations Function 1	Correlations Function 2
X1	.001	.003	-.378	.109
X2	-.003	.002	-.785	.052
X3	.004	.001	-.207	.067
X4	-.002	-.000	-.618	-.086
X5	-.000	-.000	-.534	-.053
X6	-.000	.000	-.809	.098
X7	-.000	.000	-.422	.061
X8	-.000	-.000	-.598	-.100
X9	.000	.000	-.249	-.207
X10	.000	-.000	-.475	-.660
X11	-.000	-.000	-.160	-.173
X12	-.000	-.000	-.423	-.522

	Function 1	Function 2
Chi-squared	2192	1157
df	228	198
Probability	p<.0001	p<.0001
% Variance	59.2%	31.1%

According to the results from the first discriminant function, variables X6, X2, X4, X8 and X5 were mostly responsible for the rejection of the null hypothesis. Variables X6 and X2 were obtained from the HD algorithm, while variables X4 and X8 were derived from the AS algorithm. Variable X5 is associated with the RGES algorithm. These variables were derived from either makespan or job waiting time. Many of the standardized coefficients were zero, but it is believed that this is due to the program being written to provide only 3 places of decimal thus eliminating the significant digits in the remaining portion of the number.

The correlations associated with the second discriminant function indicate that derived variables X10 and X12 are mostly responsible for the rejection. Variable X10 is associated with the HD algorithm while variable X12 is derived from the AS algorithm. They are both related to the machine idle time. This discriminant function explained 31% of the variance.

The Difference in Heuristic Algorithms

This section investigates the cause of the rejection of the hypothesis on equal performances by the heuristic algorithms. The discriminant coefficients and the correlations used in this analysis were obtained as a by-product of the multivariate analysis of variance done by the FRMLM program. They are displayed in Table 5.18 for the small problems and Table 5.19 for the large problems.

Small problems. Although 12 cannonical variates were obtained in the output from the computer program, only the results of the first two are reported because they account for more than 87% of the variance. The third variate is statistically significant given a significance level of .05, but it accounts for only 4% of the variance. The remaining variates were not significant.

TABLE 5.18

DISCRIMINANT COEFFICIENTS, SMALL PROBLEMS

Derived Variable	Standardized Coefficients Function 1	Standardized Coefficients Function 2	Correlations Function 1	Correlations Function 2
X1	.004	-.014	-.231	.099
X2	-.019	.021	-.772	.196
X3	.008	.009	-.232	.241
X4	.003	-.007	-.246	.064
X5	-.001	-.001	-.218	-.143
X6	-.001	-.000	-.298	.015
X7	.001	.000	-.203	.102
X8	-.000	-.001	-.002	-.073
X9	.001	.003	-.133	.086
X10	.002	-.005	-.693	-.350
X11	-.000	.002	-.161	.119
X12	-.002	.000	-.251	-.059

	Function 1	Function 2
Chi-squared	869	310
df	180	154
Probability	p<.0001	p<.0001
% Variance	77%	10.6%

Variables X10, X2 and X6 have the strongest correlation with the discriminant function. These variables were constructed from the differences between the RAES and HD algorithms for each original dependent variable. Thus the results for the RAES algorithm on all three dependent variables were significantly different from those for the HD algorithm. The variable associated with HD and makespan, X2, had the greatest contribution towards the rejection of the hypothesis. The plots of the means in Figure 5.1 verified these findings. The values for the HD algorithm are undoubtedly larger than those for the other algorithms. The second discriminant function which is orthogonal to the first is most strongly correlated to variables X10 and X2. These variables are also strongly correlated to the first discriminant function. Thus, further evidence is obtained that the HD algorithm yielded results which were different from those of the other algorithms.

Large problems. The results for these problems are found in Table 5.19.

TABLE 5.19

DISCRIMINANT COEFFICIENTS, LARGE PROBLEMS

Derived Variable	Standardized Coefficient	Correlations
X1	.001	-.365
X2	-.007	-.901
X3	.004	-.212
X4	-.002	-.527
X5	-.000	-.325
X6	.000	-.440
X7	-.000	-.251
X8	.000	-.307
X9	.000	-.177
X10	.000	-.469
X11	-.000	-.130
X12	-.000	-.336

Chi-squared	= 3725
df	= 240
Probability	= p<.0001
% Variance	= 89

The first four cannonical variates were significant, given a .05 significance level. Only one was reported because it accounted for 89% of the variance. From the information provided, there is a very strong correlation between the discriminant function and the variable X2. This indicates that the difference between the data for the RAES and the HD algorithms on the makespan variable was significant, thus making it inappropriate to state that the algorithms produce the same results. The variables X4, X6 and X10 were also among those which contributed to the rejection of the hypothesis. The variable X4 was formed from the difference between the RAES and AS algorithms with makespan as the dependent variable, while variables X6 and X10 were formed from the RAES and HD algorithms with job waiting time and machine idle time as the dependent variables. An examination of the plots of the means for the large problems in Figures 5.4 to 5.6 corroborates these results.

Regression Analysis

There is considerable similarity between the analysis of variance and the regression analysis techniques. Since the former technique was the major tool in this research study, it was decided to use regression analysis to investigate the predictive relationship between the dependent and the independent variables of this study. In the initial phase, the dependent variables were taken collectively and regressed on the independent variables using a multivariate regression technique. In the second phase, the CPU time variable was regressed upon the independent variables, using multiple linear regression analysis where possible.

Multivariate Regression Analysis

The computations for the multivariate regression analyses were done by the BMDP6R computer program [38]. Makespan, job waiting time and machine idle time were the three dependent variables which were regressed on the independent variables, number of jobs and number of machines. The output from the computer program included the regression coefficients, t-statistics and significance tests.

Small problems. The regression coefficients for the small problems appear in Table 5.20. The t-statistics for the regression coefficients are displayed in Table 5.21. The significance levels (two-tails) for the t-ststistics are also included in this table.

According to the figures in Table 5.21, the constant terms and the coefficients in the regression equations are significantly different from zero because the absolute values of the t-statistics are all larger than two. The significance levels (two-tails) for the t-statistics also verify this conclusion. The F-statistics and the accompanying information also indicate that the regression equations are significant. The computer output stated that there is no correlation between the independent variables, thus indicating an absence of multi-colinearity.

TABLE 5.20

REGRESSION COEFFICIENTS FOR SMALL PROBLEMS

Independent Varibles	Dependent Variables		
	Makespan	Job Waiting Time	Machine Idle Time
Intercept	-134.2	-1972.2	-2082.4
Jobs	58.6	433.1	59.1
Machines	69.1	86.2	497.1
F-statistic	9616.3	8736.7	11558.3
df	2 and 2247	2 and 2247	2 and 2247
Significance	p<.000	p<.000	p<.000

TABLE 5.21

t-STATISTICS FOR REGRESSION COEFFICIENTS, SMALL PROBLEMS

Independent Variables	Dependent Variables		
	Makespan	Job Waiting Time	Machine Idle Time
Intercept	-19.25(.000)*	-59.51(.000)	-56.47(.000)
Jobs	82.84(.000)	128.93(.000)	15.81(.000)
Machines	111.22(.000)	29.18(.000)	151.22(.000)

*Numbers in parentheses denote the significance levels (two-tails) for the t-statistics

The values of the regression coefficients suggest that the number of machines have a greater influence on the amount of makespan and machine idle times. In the case of the job waiting times, the number of jobs has the greater influence. These findings are also available in the plots of the mean values for the dependent variables in Figures 5.1, 5.2 and 5.3. For example, each plot of the makespan means for the 10 jobs x 4 machines problems (plots on the line labeled M) needed less time than the plots for the 4 jobs x 10 machines problems (plots on the line labeled C). The same observations can be made on the plots of the machine idle time means. This statement is, however, not true for the plots of the job waiting means.

Large problems. The large problems yielded multivariate regression equations which were similar to those obtained from the small problems. The regression coefficients and F-statistics are reported in Table 5.22.

TABLE 5.22

REGRESSION COEFFICIENTS FOR LARGE PROBLEMS

Independent	Dependent Variables		
Variables	Makespan	Job Waiting Time	Machine Idle Time
Intercept	-590.2	-150,782.0	-79,258.0
Jobs	67.8	4,736.5	483.9
Machines	79.7	1,274.9	3,706.9
F-statistics	53,190.3	33,804.5	30,845.2
df	2 and 2,997	2 and 2,997	2 and 2,997
Significance	p<.000	p<.000	p<.000

The t-statistics for the regression coefficients are tabulated in Table 5.23. The significance levels (two-tails) for the t-statistics are also included in this table.

TABLE 5.23

t-STATISTICS FOR REGRESSION COEFFICIENTS, LARGE PROBLEMS

Independent Variables	Dependent Variables		
	Makespan	Job Waiting Time	Machine Idle Time
Intercept	-25.4(.000)*	-89.2(.000)	-95.5(.000)
Jobs	267.7(.000)	256.8(.000)	53.4(.000)
Machines	186.4(.000)	41.0(.000)	242.6(.000)

*Numbers in parentheses denote the significance levels (two-tail) for the t-statistics

The t-statistics indicate that the regression coefficients are all significantly different from zero. The regression equations are also significant because they yielded F-statistics that are greater than the critical F-values for the reported degrees of freedom. In addition, the output of the computer program reported that there was no correlation between the independent variables.

These results therefore lead to conclusions similar to those for the small problems. They can also be verified by examining the plots of the means of the dependent variables in Figures 5.3, 5.4 and 5.6, especially with respect to the relative values of the regression coefficients.

General considerations. The multivariate regression equations reported in the preceding sections can be used to predict values of the dependent variables. These equations were derived from data for all of the heuristic algorithms.

The accompanying statistical tests suggest that these equations are sound. Nevertheless, more exact equations were possible if the data used had been limited to a particular heuristic algorithm.

Regression Analysis With the CPU Times

The small problem. A crude plot of a small portion of the data for the small problems suggested that there was a linear relationship between the CPU time and the number of jobs. This was followed by a multiple regression analysis of the CPU times on the number of jobs and the number of machines. This analysis was done for each heuristic algorithm. In general, it can be concluded that a multiple linear regression model fits the data for the small problems. The r-squared value of the models for the various heuristic algorithms varied from .66 for the AS algorithm to .82 for the HD algorithm. This analysis was done with the GLM procedure of the SAS statistical package [18]. The regression equations for the various heuristic algorithms are reported in Table 5.24. From these equations, it appears that the number of jobs has a greater effect on the CPU time than the number of machines for all heuristic algorithms except the CDS algorithm.

The various test statistics associated with the regression equations for the CPU times demonstrate that the equations are statistically sound and would be good predictors of the CPU times required by the various heuristic algorithms for various problem sizes within the range of the small problems.

TABLE 5.24

CPU REGRESSION EQUATIONS, SMALL PROBLEMS

Heuristic Algorithm	Regression Equation*	R	F-value	PR>F
RGES	T=-.140+.025J+.012M	.79	816	.0001
HD	T=-.096+.018J+.004M	.82	1052	.0001
RAES	T=-.095+.014J+.009M	.67	456	.0001
CDS	T=-.011+.002J+.003M	.80	880	.0001
AS	T=-.026+.005J+.003M	.66	436	.0001

*T=CPU Time J=Number of Jobs M=Number of Machines

The t-statistics for the equations in Table 5.24 are reported in Table 5.25. Their values indicate that the intercepts and independent variables are all statistically significant at the .0001 level.

TABLE 5.25

t-STATISTICS FOR THE SMALL CPU REGRESSIONS

Heuristic	Independent Variables*		
Algorithm	Intercept	J	M
RGES	-20.3(.0001)	35.00(.0001)	20.2(.0001)
HD	-24.2(.0001)	44.63(.0001)	10.7(.0001)
RAES	-16.9(.0001)	24.70(.0001)	17.4(.0001)
CDS	-13.9(.0001)	20.00(.0001)	36.9(.0001)
AS	-13.1(.0001)	25.10(.0001)	15.5(.0001)

*J=Number of Jobs M=Number of Machines

The large problems. Multiple linear regression of the CPU time on the two independent variables, the number of jobs and the number of machines, did not yield good results. The r-squared values were extremely low, mostly near .20. A crude plot of a portion of the data revealed that the relationship between the CPU times and the number of jobs was not linear. The shape of the curve suggested that second degree terms might contribute toward a more accurate description of the relationship. Terms consisting of the second degree of the original independent variables were then added to the original equation and the regression analysis was repeated using the Stepwise Procedure of the SAS statistical package [18] with the quadratic terms redefined as linear variables. These definitions appear as a footnote at the end of Table 5.26.

TABLE 5.26

CPU REGRESSION EQUATIONS, LARGE PROBLEMS

Heuristic Algorithm	Regression Equation*	R^2	F-value	PR>F
RGES	T=-8.0J+7.2M-.03JM+.11JJ-.04MM	.79	441.8	.0001
HD	T=-3.7J+3.5M-.02JM+.05JJ-.02MM	.86	719.4	.0001
RAES	T=-8.7J+7.5M-.05JM+.12JJ-.03MM	.79	458.0	.0001
CDS	T=-.21J+.16M-.0007JM+.0029JJ	.88	1071.5	.0001
AS	T=-.75J+.69M-.003JM+.01JJ-.004MM	.85	698.8	.0001

*T=CPU Time J=Number of Jobs M=Number of Machines JM=JxM JJ=JxJ MM=MxM

Considerable improvement in the r-squared values was obtained as a result of this action. These new values varied from .79 to .88. They are displayed in Table 5.26 along with the regression equations and other statistics. The t-statistics for all of the independent variables appearing in the equations are reported in Table 5.27.

The t-statistics confirm that the variables included in the equations are all significantly different from zero. The F-ratios and the r-squared values indicate that the equations are statistically sound and that they would be good predictors of CPU time within the range of problem sizes considered.

TABLE 5.27

t-STATISTICS FOR THE LARGE CPU REGRESSIONS

Heuristic	Independent Variables*				
Algorithm	J	M	JM	JJ	MM
RGES	22.4**	16.5	6.6	27.1	5.0
HD	30.0	23.0	9.4	36.1	8.0
RAES	22.1	15.8	8.7	27.6	3.7
CDS	27.6	46.7	6.9	28.5	0.0
AS	26.7	20.4	7.8	32.7	6.4

*J=Number of Jobs M=Number of Machines JM=JxM
JJ=JxJ MM=MxM
**Non-zero t-values were significant at the .0001 level

Implications for Managers

The analysis of the data generated for this research proved that the RAES algorithm produced the best results for all three dependent variables. These findings were confirmed by multivariate analysis of variance, graphical analysis and ratio analysis. The statistical techniques were very sensitive and therefore detected minute differences. Thus, although the RAES produced the best results, the ratios reported in Table 5.3 and 5.4 indicated that at certain problem sizes, some of the other algorithms produced comparable results. On the other hand, it was clear that the HD algorithm produced by far the worst results and should be abandoned from further consideration over the range of problem sizes studied. Similarly, the conclusions about RAES' superior performance are also limited to the range of problems studied because the levels of problem size were fixed by the researcher.

When the CPU time required by the heuristic algorithms is considered, the RAES algorithm becomes less attractive since it is a large consumer of time, especially at the large problem sizes. The CDS algorithm needed considerably less CPU time to arrive at solutions which were slightly inferior to those of the RAES algorithm. It should be noted that the CDS algorithm did not show very prominently in the discriminant analysis when causes for the lack of equality of the results from the heuristic algorithms were being explored.

Thus, in comparing heuristic algorithms, managers should not focus solely on the values of the dependent variable, but should also consider the computer time needed by the heuristic algorithm to arrive at solutions. Perhaps a manager who is interested in minimizing the use of CPU time could select the CDS algorithm over the RAES algorithm and obtain results which are almost as good for a much smaller expenditure of time.

Much of the statistical analysis proceeded with the knowledge that there was interaction between the two factors. Plots of the data indicated that the interaction was very slight although it was recognized by the sensitive statistical technique used in this research. The multivariate procedure for investigating differences between the levels of the two factors was reported by Timm [109] who stated that parallelism, or a lack of interaction, was not a necessary condition for its use. It, however, requires caution in interpreting the results.

The sensitivity of the analytical technique for comparing heuristic algorithms also needs careful consideration. It may discriminate between the heuristic algorithms on the basis of minute differences which may be of much less importance than factors not included in the analysis.

All of the statistical tests were confirmed in a crude manner by graphical analysis. Graphical analysis, however, lacks the precision of statistical analysis and would not be a good substitute for statistical analysis.

There are certain decisions which cannot be made by this technique. In particular, when there are choices to be made based on management's utilities for particular variables, this technique cannot substitute for the manager. It can assist him in providing facts but cannot make the decision. An example of this occurred in a preceding paragraph when the issue was the choice between the RAES algorithm which produced the best values for the performance measures and the CDS algorithm which produced slightly worse values but used much less CPU time. The choice between these two algorithms depends upon the manager's utility for both the CPU time and the best results. These utilities are not incorporated into this analytical technique.

CHAPTER 6

SUMMARY AND CONCLUSIONS

The initial portion of this chapter consists of a summary of the research study described in the first five chapters. This is followed by the conclusions which were drawn from the results of the investigation. A number of recommendations about topics and directions for future studies are presented in the third and final section of the chapter.

Summary

The general _nxm_ flowshop scheduling problem was studied in this research. Five heuristic algorithms were used to solve a number of flowshop problems and the results were compared using three measures of performance. The following are the names of heuristic algorithms which were employed:

Random Generation with Extensive Search (RGES)
Heuristic Decomposition (HD)
Rapid Access with Extensive Search (RAES)
Campbell, Dudek and Smith (CDS)
Aggarwal and Stafford (AS).

The three measures of performance were:

Makespan
Job waiting time
Machine idle time.

The research design consisted of a two-factor experiment, with the heuristic algorithms as one factor and the problem sizes as the other factor. The experiments were designed to test for the equality of the results for the levels of each factor using both the univariate and multivariate analysis of variance techniques. In addition, graphical methods and ratio analysis were used to supplement the analysis of variance techniques. Post hoc analysis, using the multiple discriminant analysis technique, was used to isolate the causes of the differences in the effect of the levels of the factors.

The research was carried out entirely in a laboratory setting. In conformance with previous studies, one source of variance was eliminated by submitting the same problems to each heuristic algorithm. This, however, resulted in a lack of independence between the levels of the factor and, as a result, it became necessary to adopt a profile analysis approach to the multivariate analysis of variance technique.

Because of the laboratory nature of the research, the processing times for each job on each machine were randomly generated and were gathered into a matrix. This matrix was submitted to a computerized version of each heuristic algorithm which used it to arrive at a schedule. The schedule was then simulated through a computerized model of a flowshop and the values of the performance measures were obtained as output. Due to the simulation approach and the random nature of the processing times, the simulation was repeated 29 times, each time using a new matrix of processing times. This provided a sample size of 30, which, according to the Central Limit Theorem [31], satisfies the requirement under which statistics based on the normal distribution can be used.

The sizes of the problems solved varied considerably and, as a result, were divided into two categories: small problems and large problems. The small problems consisted of 10 or less jobs and 10 or less machines. The largest problem solved had 100 jobs and 60 machines. There were 15 problem sizes for the small problems and 20 for the large problems.

The values of the performance measures as well as the CPU times required by each algorithm to solve the flowshop problems were collected and the mean values were determined for each problem size. These mean values were then plotted on graphs for each performance measure and for the CPU times. Ratios of the values of each performance measure for each algorithm to the corresponding value for the RAES algorithm were also determined, averaged and plotted on graphs. The data was then subjected to the analysis of variance procedure. It was also subjected to regression analysis in order to develop predictive equations for the dependent variables, the performance measures and the CPU times, given the number of jobs and the number of machines.

Conclusions

Many of the conclusions are drawn from the statistical tests of the hypotheses. In addition, there are other conclusions which pertain to the heuristic algorithms and the analytical techniques which were employed. It was concluded that:

1. There was a significant difference in the values of the performance measures yielded by the heuristic algorithms. It was observed from the data that the RAES algorithm produced the best overall results for the range of problems studied. On the other hand, the HD algorithm yielded the worst results. The post hoc multiple discriminant analysis clearly identified the HD algorithm as the major cause for the differences in the measures of performance for the heuristic algorithms. The plots of the means of the performance measures as well as the ratios of these measures for the heuristic algorithms also confirm these conclusions.

2. When univariate analysis of variance was used, each algorithm produced results which were significantly different from the results of the RAES algorithm, which

produced the best overall figures. There was one exception--the AS algorithm matched RAES' performance over the small jobs when job waiting time was the measure of performance.

3. There was a significant difference in the results produced at the various problem sizes. It was noted that as the problem sizes increased, the efficiency of the heuristic algorithms slowly decreased.

4. The heuristic algorithms required significantly different amounts of CPU time to produce a schedule. The CDS algorithm always required the least amount of time to produce a schedule. The RGES algorithm generally consumed the greatest amount of time, except in the case of the 100 jobs problems where the RAES algorithm became the greatest consumer of time. There is a considerable difference in the time requirements between the CDS algorithm and the RGES or RAES algorithms. The time requirements of the AS algorithm were smaller than those for all of the other algorithms except CDS. They were, however, distinctly different from CDS' time consumption.

5. Although the values of the dependent variables for the CDS algorithm were inferior to the corresponding values for the RAES algorithm, the differences were very small and sometimes hardly distinguishable on the graphs. On the other hand, the CDS algorithm required considerably less CPU time than the RAES algorithm to develop schedules.

Therefore, the RAES algorithm produces slightly better values at the cost of a considerable increase in CPU time. Thus, the choice between these two algorithms would depend upon the goals of the manager, the availability of computer processing time, and the manager's preference for either the best solution or obtaining solutions with very little consumption of computer time.

6. The RAES algorithm produced more optimal solutions than the other algorithms. The HD algorithm yielded the least while the RGES, AS, and CDS algorithms determined a similar number of optimal solutions. The optimal solutions occurred mostly at the smaller problem sizes. The RGES algroithm produced optimal solutions for all 30 replications at the smallest problem size.

7. The number of machines had a considerably greater effect on the machine idle times than the number of jobs. The reverse was also true for the job waiting times. When makespan was the measure of performance, the number of machines had a slightly greater effect than the number of jobs.

8. The multivariate analysis of variance approach provided comparisons with statistical precision. When combined with multiple discriminant analysis, it enabled one to identify the causes of the differences between the results for the levels of the factors. This technique indicated differences that were sometimes difficult to

recognize from the graphs because they were obscurred by the scale. Due to its precision, this approach proved to be an improvement over the previous methods of comparison.

9. The regression equations, relating the CPU time required by each heuristic algorithm to the number of jobs and the number of machines, were statistically significant for both the small and the large problems. The t-statistics, r-squared values and F-ratios for the regression lines were all significant. Their values suggested that these equations had good predictive qualities.

Recommendations for Future Studies

1. Since the HD heuristic algorithm produced inferior results over the range of problems studied, it should be eliminated from future studies. Both the CDS and AS algorithms yielded better results with a much smaller consumption of CPU time. The RGES algorithm also produced better results than the HD algorithm but should also be eliminated over the range of problem studied, except for the 100 job problems. This is justified because the algorithm was the highest consumer of CPU time for problems with 80 or less jobs while its schedules were generally not the best except at the very smallest problem sizes where they were comparable with RAES'.

2. This study should be repeated with the above eliminations in effect. The remaining algorithms should be

further investigated because each one was outstanding at some particular sub-range of problem sizes. Thus, narrowing the range of problem sizes might lead to more useful information.

3. Other heuristic algorithms should be investigated and compared with the RAES, CDS and AS algorithms. For example, algorithms by Stinson [102], and Krone and Steiglitz [72], among others, produced interesting results in the articles in which they were reported. These and others which appeared in the literature too late for inclusion in the study are worthy of consideration.

4. Additional performance measures should be considered. For example, mean flowtime was used in a study reported by Gupta [54], and the results produced were different from the results for makespan. In that study, the CDS algorithm was not very efficient according to the mean flowtime criterion. If possible, actual costs should also be used which would permit production cost to be used as a criterion. This is the most ideal criterion since it is a true measure of business performance.

5. Since the improvement routine helped the RAES algorithm to produce better results than its predecessors, RA and RACS, it is believed that the CDS could be made to produce even better results by appending an improvement

routine to it. This may even enable it to surpass the performance of the RAES algorithm.

6. Although a very broad range of problem sizes was covered in this research, it is recommended that even larger problem sizes be investigated. It was noticed that at the largest sizes investigated, there were changes in the efficiencies of the heuristic algorithms. For example, the RAES algorithm needed more CPU time than the RGES algorithm to arrive at solutions. At all other problem sizes, the reverse was true. Thus, as noted earlier, algorithms behave differently at different problem sizes.

7. In this study, a number of regression equations were developed to relate the dependent variables and CPU times to the number of jobs and the number of machines. The equations for predicting the dependent variables were developed from the combined data for all of the heuristic algorithms. If this analysis is repeated with the data for each algorithm processed separately, then the equations developed would provide more accurate predictions because each equation pertains to a particular heuristic algorithm.

8. In future research, fewer levels for the problem size factor should be used. This would reduce the high sensitivity exhibited by the multivariate analysis of variance technique in this study because of a corresponding reduction in the degrees of freedom. This approach would

certainly raise the numerical value of the significance level at which interaction first occurs.

9. In this study, no test was done to check for the equality of the variance-covariance matrices. Instead, the equality was assumed because of the robustness of the multivariate technique towards violation of the assumption. This assumption, however, could be relaxed by actually testing for the equality of the variance-covariance matrices.

10. Similarly, a test of multivariate normality was not done but, instead, the guidelines of the Central Limit Theorem were followed in selecting a sample size of 30 in order that the sums and means of the observations may be assumed to follow a normal distribution. This assumption could also be relaxed by actually testing for multivariate normality.

BIBLIOGRAPHY

1. Aggarwal, S. C. and Stafford, E. "A Heuristic Algorithm for the Flowshop Problem With a Common Job Sequence on All Machines." Decision Sciences 6 (Apr. 1975):237-251.

2. Anderson, T. W. An Introduction to Multivariate Statistical Analysis. N. Y.: Wiley, 1958.

3. Anderson, T. W.; Das Gupta, S.; and Styan, G. P. H. A Bibliography of Multivariate Statistical Analysis. N. Y.: Wiley, 1972.

4. Appelbaum, M. I. The MANOVA Manual: Complete Factorial Design. Research Memorandum No. 44, University of N.C., Chapel Hill, N. C., 1974.

5. Arthanari, T. S. and Ramamurthy, K. G. "A Branch and Bound Algorithm for Sequencing n Jobs on m Parallel Processors." Opsearch 7, No. 3 (Sep. 1970):147-156.

6. Ashour, S. "A Decomposition Approach for the Machine Scheduling Problem." The International Journal of Production Research 6, No. 2 (1967):109-122.

7. Ashour, S. "A Modified Decomposition Algorithm for Scheduling Problems." The International Journal of Production Research 8, No. 3 (1970):281-284.

8. Ashour, S. "An Experimental Investigation and Comparative Evaluation of Flow-Shop Scheduling Techniques." Operations Research 18, No. 3 (May-June 1970):541-549.

9. Ashour, S. "A Branch-and-Bound Algorithm for Flow Shop Scheduling Problems." AIIE Transactions 2, No. 2 (June 1970):172-176.

10. Ashour, S. and Char, A. R. "Computational Experience on Zero-One Programming Approach to Various Combinational Problems." Journal of Operations Research Society of Japan 13, No. 2 (Oct. 1970):78-108.

11. Ashour, S. and Quraishi, M. N. "Investigation of Various Bounding Procedures for Production Scheduling Problems." The International Journal of Production Research 7, No. 3 (1969):249-252.

12. Bagga, P. C. and Chakravarti, N. K. "Optimal *m*-Stage Production Schedule." Canadian Operational Research Society Journal 6, No. 2 (July 1968):71-78.

13. Baker, K. R. Introduction t6 Sequencing and Scheduling. N. Y.: Wiley, 1974.

14. Baker, K. R. "A Comparative Study of Flow-Shop Algorithms." Operations Research 23, No. 1 (Jan.-Feb. 1975):62-73.

15. Baker, K. R. "An Elimination Method for the Flow-Shop Problem." Operations Research 23, No. 1 (Jan.-Feb. 1975):159-162.

16. Bakshi, M. S. and Arora, S. R. "The Sequencing Problem." Management Science 16, No. 4 (Dec. 1969):B247-263.

17. Bansal, S.P. "Global Lower Bound in the Flow-Shop Problem." Opsearch 13, No. 2 (June 1976):115-118.

18. Barr, A. J.; Goodnight, J. H.; Sell, J. P.; and Helwig, J. T. A User's Guide to SAS 76. Raleigh, N. C.: SAS Institute, 1976.

19. Bock, R. D. "Multivariate Analysis of Variance of Repeated Measurements." In Problems of Measuring Change, pp. 85-103. Edited by C. W. Harris. Madison: University of Wisconsin Press, 1963.

20. Bock, R. D. "Contributions of Multivariate Experimental Designs to Educational Research," In Handbook of Multivariate Experimental Psychology, pp. 820-840. Edited by R. B. Cattell. Chicago: Rand McNally, 1966.

21. Bock, R. D. Multivariate Statistical Methods in Behavioral Research. N. Y.: McGraw-Hill, 1975.

22. Bolch, B. W. and Huang, C. J. Multivariate Statistical Methods for Business and Economics. Englewood Cliffs, N. J.: Prentice-Hall, 1974.

23. Borovits, I. and Ein-Dor, P. "Two Issues in the Evaluation of Job-Shop Schedules." OMEGA 4, No. 3 (1976):313-320.

24. Bowman, E. H. "The Schedule-Sequencing Problem." Operations Research 7 (1959):621-624.

25. Brown, A. P. G. and Lomnicki, Z. A. "Some Applications of the 'Branch and Bound' Algorithm to the Machine Scheduling Problem." Operational Research Quarterly 17, No. 2 (June 1966):173-186.

26. Burns, F. and Rooker, J. "A Special Case of the 3X*n* Flow Shop Problem." Naval Research Logistics Quarterly 22, No. 4 (Dec. 1975):811-817.

27. Campbell, H. G.; Dudek, R. A.; and Smith, M. L. "A Heuristic Algorithm for the *n* Job, *m* Machine Sequencing Problem." Management Science: Application 16, No. 10 (June 1970):B630-B637.

28. Carlson, J. E. and Timm, N. H. "Program Manuals: FRULM and FRMLM Computer Programs." University of Pittsburgh, 1974.

29. Charlton, J. M. and Death, C. C. "A Method of Solution for General Machine-Scheduling Problems." Operations Research 18, No. 4 (July-Aug. 1970):689-707.

30. Chase, R. B. and Aquilano, N. J. Production and Operations Management. Homewood, Ill.: Irwin, Revised Edition, 1977.

31. Chou, Y. Statistical Analysis. New York: Holt, Rinehart and Winston, 1969.

32. Cole, J. W. L. and Grizzle, J. E. "Applications of Multivariate Analysis of Variance to Repeated Measurements Experiments." Biometrics 22 (Dec. 1966): 810:828.

33. Conway, R. W.; Maxwell, W. L.; and Miller, L. W. Theory of Scheduling. Reading, Mass.: Addison Wesley, 1967.

34. Cooley, W. W. and Lohnes, P. R. Multivariate Data Analysis. New York: Wiley, 1971.

35. Danford, M. B.; Hughes, H. M.; and McNee, R. C. "On the Analysis of Repeated-measurements Experiments." Biometrics 16 (Dec. 1960):547-65.

36. Dannenbring, D. G. "The Evaluation of Heuristic Solution Procedures for Large Combinational Problems." Ph.D. Dissertation, Columbia University, N. Y., 1973.

37. Dannenbring, D. G. "An Evaluation of Flow Shop Sequencing Heuristics." Management Science 23, No. 11 (July 1977):1174-1182.

38. Dixon, W. J. BMDP Biomedical Computer Programs P-Series. L.A.: UC Press, 1977.

39. Dudek, R. A.; Smith, M. L.; and Panwalkar, S. S. "Use of a Case Study in Sequencing/Scheduling Research." OMEGA 2, No. 2 (Apr. 1974):253-261.

40. Dudek, R. A. and Teuton, O. F., Jr. "Development of M-Stage Decision Rule For Scheduling n Jobs Through M Machines." Operations Research 12, No. 3 (May-June 1964):471-497.

41. Elmaghraby, S. E. "The Machine Sequencing Problem--Review and Extensions." Naval Research Logistics Quarterly 15, No. 2 (June 1968):205-232.

42. Giglio, R. J. and Wagner, H. M. "Approximate Solutions to the Three-Machine Scheduling Problem." Operations Research 12, No. 2 (Mar.-Apr. 1964):305-324.

43. Greenhouse, S. W. and Geisser, S. "On Methods In The Analysis of Profile Data." Psychometrika 24, No. 2 (June 1959):95-112.

44. Gupta, J. N. D. "Heuristic Rules for nXM Flowshop Scheduling Problem." Opsearch 5, No. 3 (Sep. 1968): 165-170.

45. Gupta, J. N. D. "Economic Aspects of Scheduling Theory." Ph.D. Dissertation, Texas Tech University, Lubbock, Texas, 1969.

46. Gupta, J. N. D. "A General Algorithm for the nXM Flowshop Scheduling Problem." The International Journal of Production Research 7, No. 3 (1969):241-247.

47. Gupta, J. N. D. "M-stage Flowshop Scheduling by Branch and Bound." Opsearch 7, No. 1 (Mar. 1970):37-43.

48. Gupta, J. N. D. "M-stage Scheduling Problem--A Critical Appraisal." The International Journal of Production Research 9, No. 2 (1971):267-281.

49. Gupta, J. N. D. "Economic Aspects of Production Scheduling Systems." Journal of the Operations Research Society of Japan 13, No. 4 (Mar. 1971):169-193.

50. Gupta, J. N. D. "A Functional Heuristic Algorithm for the Flowshop Scheduling Problem." Operational Research Quarterly 22, No. 1 (Mar. 1971):39-47.

51. Gupta, J. N. D. "Optimal Flowshop Scheduling with Due Dates and Penalty Costs." Journal of the Operations Research Society of Japan 14, No. 1 (June 1971):35-46.

52. Gupta, J. N. D. "The Generalized n-Job, M-Machine Scheduling Problem." Opsearch 8, No. 3 (Sep. 1971): 173-185.

53. Gupta, J. N. D. "An Improved Combinatorial Algorithm For the Flowshop-Scheduling Problem." Operations Research 19, No. 7 (Nov.-Dec. 1971):1753-1758.

54. Gupta, J. N. D. "Heuristic Algorithms for Multistage Flowshop Scheduling Problem." AIIE Transactions 4, No. 1 (Mar. 1972):11-18.

55. Gupta, J. N. D. "Analysis of a Combinatorial Approach to Flowshop Scheduling Problems." Operational Research Quarterly 26, No. 2, ii (July 1975):431-440.

56. Gupta, J. N. D. "A Search Algorithm for the Generalized Flowshop Scheduling Problem." Computers and Operations Research 2, No. 2 (Sep. 1975):83-90.

57. Gupta, J. N. D. and Dudek, R. A. "Optimality Criteria for Flowshop Schedules." AIIE Transactions 3, No. 3 (Sep. 1971):199-205.

58. Gupta, J. N. D. and Maykut, A. R. "Flow-shop Scheduling by Heuristic Decomposition." The International Journal of Production Research 11, No. 2 (Apr. 1973):105-111.

59. Gupta, J. N. D. and Maykut, A. R. "Heuristic Algorithms for Scheduling n Jobs in a Flowshop." Journal of the Operations Research Society of Japan 16, No. 3 (Sep. 1973):131-150.

60. Hair, J. F., Jr.; Anderson, R. E.; Tatham, R. L.; and Grablowsky, B. J. Multivariate Data Analysis. Tulsa, Ok.: PPC Books, 1979.

61. Harris, R. J. A Primer of Multivariate Statistics. N. Y.: Academic Press, 1975.

62. Heller, J. "Some Numerical Experiments for an MXJ Flow Shop and Its Decision-Theoretical Aspects." Operations Research 8, No. 2 (Mar.-Apr. 1960):178-184.

63. Huynh, H. and Feldt, L. S. "Conditions Under Which Mean Square Ratios in Repeated Measurement Designs Have Exact F-distributions." Journal of the American Statistical Association 65 (1970):1582-1589.

64. Ignall, E. and Schrage, L. "Application of the Branch and Bound Technique to Some Flow-Shop Scheduling Problems." Operations Research 13, No. 3 (May-June 1965):400-412.

65. Johnson, S. M. "Optimal Two- and Three-Stage Production Schedules with Set-up Times Included." Naval Research Logistics Quarterly 1, No. 1 (Mar. 1954):61-68.

66. Jones, L. V. "Analysis of Variance in Its Multivariate Developments." In Handbook of Multivariate Experimental Psychology, pp. 244-266. Edited by R. B. Cattell. Chicago: Rand McNally, 1966.

67. Kirk, R. E. Experimental Design: Procedures For the Behavioral Sciences. Monterey, Calif.: Brooks/Cole, 1968.

68. Kleijnen, J. P. C. Statistical Techniques in Simulation--Part I. N. Y.: Marcel Dekker, 1974.

69. Kleijnen, J. P. C. Statistical Techniques in Simulation--Part II. N. Y.: Marcel Dekker, 1975.

70. Krishnaiah, P. R. "Multiple Comparison Procedures in Multi-response Experiments." Sankhya, Series A 27 (1965):31-36.

71. Krishnaiah, P. R. "Some Recent Developments on Real Multivariate Distributions." In Developments in Statistics 1, pp. 135-169. Edited by P. R. Krishnaiah. N. Y.: Academic Press, 1978.

72. Krone, M. J. and Steiglitz, K. "Heuristic-Programming Solution of a Flowshop-Scheduling Problem." Operations Research 22, No. 3 (May-June 1974):629-638.

73. Lomnicki, Z. A. "A 'Branch-and-Bound' Algorithm for the Exact Solution of the Three-machine Scheduling Problem." Operational Research Quarterly 16, No. 1 (Mar. 1965):89-100.

74. McMahon, G. B. "Optimal Production Schedules for Flow Shops." Canadian Operational Research Society Journal 7, No. 2 (July 1969):141-151.

75. McMahon, G. B. and Burton, P. G. "Flow-Shop Scheduling with the Branch-and-Bound Method." Operations Research 15, No. 3 (May-June 1967):473-481.

76. Manne, A. S. "On the Job-Shop Scheduling Problem." Operations Research 8, No. 2 (Mar.-Apr. 1960):219-223.

77. Mellor, P. "A Review of Job Shop Scheduling." Operational Research Quarterly 17, No. 2 (June 1966):161-171.

78. Morrison, D. F. Multivariate Statistical Methods. 2nd Edition. N. Y.: McGraw-Hill, 1976.

79. Nabeshima, I. "The Order of n Items Processed on m Machines(III)." Journal of the Operations Research Society of Japan 16, No. 3 (Sep. 1973):163-185.

80. Namboodiri, N. K.; Carter, L. F.; and Blalock, H. M., Jr. Applied Multivariate Analysis and Experimental Designs. N. Y.: McGraw-Hill, 1975.

81. Naylor, T. H. Computer Simulation Experiments With Models of Economic Systems. N. Y.: Wiley, 1971.

82. Nicewander, W. A. "Multivariate Analysis of Univariate and Multivariate Repeated Measures Designs Using MANOVA." Paper presented at Sou'hwestern Psychological Association Meetings, New Orleans, Apr. 1978.

83. Overall, J. E. and Klett, C. J. Applied Multivariate Analysis. N. Y.: McGraw-Hill, 1972.

84. Page, E. S. "An Approach to the Scheduling of Jobs on Machines." Journal of Royal Statistical Society, ser. B, 23, No. 2 (1961):484-492.

85. Palmer, D. S. "Sequencing Jobs Through a Multi-Stage Process in the Minimum Total Time--A Quick Method of Obtaining a Near Optimum." Operational Research Quarterly 16, No. 1 (Mar. 1965):101-107.

86. Pandit, S. N. N. and Subrahmanyam, Y. V. "Enumeration of all Optimal Job Sequences." Opsearch 12, No. 1-2 (Mar.-June 1975):35-39.

87. Panwalkar, S. S. and Khan, A. W. "An Improved Branch and Bound Procedure for nXm Flow Shop Problems." Naval Research Logistics Quarterly 22, No. 4 (Dec. 1975): 787-790.

88. Pillai, K. C. S. Statistical Tables For Tests of Multivariate Hypotheses. Manila: Statistical Center, University of the Philippines, 1960.

89. Potthoff, R. F. and Roy, S. N. "A Generalized Multivariate Analysis of Variance Model Useful Especially for Growth Curve Problems." Biometrika 51 (1964):313-326.

90. Pounds, W. F. "The Scheduling Environment." In Industrial Scheduling, Chapter 1. Edited by J. Muth and G. L. Thompson. Englewood Cliffs, N. J.: Prentice-Hall, 1963.

91. Pruzek, R. M. "Methods and Problems in the Analysis of Multivariate Data." Review of Educational Research 41, No. 3 (1971):163-190.

92. Rao, C. R. Advanced Statistical Methods in Biometric Research. N. Y.: Wiley, 1952.

93. Rao, C. R. "Recent Trends of Research Work in Multivariate Analysis." Biometrics 28 (1972):3-22.

94. Roy, J. "Step-down Procedure in Multivariate Analysis." Annals of Mathematical Statistics 29 (1958):1177-1187.

95. Roy, S. N. Some Aspects of Multivariate Analysis. N. Y.: Wiley, 1957.

96. Sisson, R. L. "Sequencing Theory." In Progress in Operations Research, Volume 1, pp. 295-325. Edited by R. L. Ackoff. N. Y.: Wiley, 1961.

97. Smith, H.; Gnanadesikan, R.; and Hughes, J. B. "Multivariate Analysis of Variance (MANOVA)." Biometrics 18 (March 1962):22-41.

98. Smith, M. L. "A Critical Analysis of Flowshop Scheduling." Ph.D. Dissertation, Industrial Engineering Department, Texas Tech University, Lubbock, Texas, 1968.

99. Smith, M. L.; Panwalkar, S. S.; and Dudek, R. A. "Flowshop Sequencing Problem With Ordered Processing Time Matrices." Management Science 21, No. 5 (Jan. 1975):544-549.

100. Smith, R. D. and Dudek, R. A. "A General Algorithm for Solution of the n-Job, M-Machine Sequencing Problem of the Flow Shop." Operations Research 15, No. 1 (Jan.-Feb. 1967):71-82.

101. Stevens, J. P. "Four Methods of Analyzing Between Variation for the k-Group MANOVA Problem." Multivariate Behavioral Research (Oct. 1972):499-522.

102. Stinson, J. P. "A Heuristic Programming Algorithm For Sequencing the Static Flowshop." Working Paper, WP-77 19, Syracuse University, 1977.

103. Story, A. E. and Wagner, H. M. "Computational Experience With Job-Shop Scheduling." In Industrial Scheduling, pp. 207-219. Edited by J. Muth and G. L. Thompson. Englewood Cliffs, N. J.: Prentice Hall, 1963.

104. Subrahmaniam, K. and Subrahmaniam, K. Multivariate Analysis: A Selected and Abstracted Bibliography, 1957-1972. N. Y.: Marcel Dekker, 1973.

105. Szwarc, W. "Elimination Methods in the mXn Sequencing Problem." Naval Research Logistics Quarterly 18, No. 3 (Sep. 1971):295-305.

106. Szwarc, W. "Optimal Elimination Methods in the mXn Flow-Shop Scheduling Problem." Operations Research 21, No. 6 (Nov.-Dec. 1973): 1250-1259.

107. Tatsuoka, M. M. Multivariate Analysis. N. Y.: Wiley, 1971.

108. Timm, N. H. "Multivariate Profile Analysis of Split-Split Plot Designs and Growth Curve Analysis of Multivariate Repeated Measures Designs." A paper presented at the annual meeting of the American Educational Research Association, Chicago, Apr. 1974.

109. Timm, N. H. Multivariate Analysis With Applications in Education and Psychology. Monterey, Calif.: Brooks/Cole, 1975.

110. Timm, N. H. and Carlson, J. E. "Multivariate Analysis of Nonorthogonal Experimental Designs Using a Multivariate Full Rank Model." Paper presented at the annual meeting of the American Statistical Association, N. Y., Dec. 1973.

111. Timm, N. H. and Carlson, J. E. "Analysis of Variance Through Full Rank Models." Multivariate Behavioral Research Monographs, MBR Monograph No. 75-1, 1975.

112. Wagner, H. M. "An Integer Linear Programming Model for Machine Scheduling." Naval Research Logistics Quarterly 6, No. 2 (June 1959):131-140.

113. Wilk, M. B. and Gnanadesikan, R. "Graphical Methods for Internal Comparisons in Multiresponse Experiments." Annals of Mathematical Statistics 35 (1964):613-631.

114. Wind, Y. and Denny, J. "Multivariate Analysis of Variance in Research on the Effectiveness of TV Commercials." Journal of Marketing Research 11, No. 2 (May 1974):136-142.

115. Winer, B. J. Statistical Principles in Experimental Design. N. Y.: McGraw-Hill, 1971.

APPENDIX A
DESCRIPTION OF THE RGES HEURISTIC ALGORITHM

The RGES heuristic algorithm arrives at a solution in two phases. In the first phase, 25 different schedules are randomly generated and the schedule with the least makespan value is selected. In order to find a schedule with a lower makespan value, this selected schedule is then subjected to a neighbor exchange improvement routine very similar to that employed by the RAES heuristic algorithm [37]. Hence, this algorithm is explicitly directed towards finding the schedule with the lowest makespan value.

The first phase is performed with the aid of a Lehmer-type random number generator. Jobs are randomly selected for each position in the schedule by using a randomly selected number to generate a job number. A routine is incorporated to check whether or not the job selected for the i^{th} position has been assigned to a preceding position. If it has, the selection procedure is repeated so that another job can be selected for the i^{th} position. After all of the jobs are sequenced by filling the available positions, the makespan value for the schedule is calculated. An additional 24 schedules are similarly generated and the schedule with the least makespan value is selected to proceed to the next phase.

The second phase consists of attempts to improve the schedule by interchanging adjacent jobs. If there are n jobs, then (n-1) new schedules, which are called neighbors, are formed by interchanging the (n-1) pairs of adjacent jobs. The neighbors are then evaluated and the neighbor with the smallest makespan value is selected for improvement if its value is smaller than the value for its parent schedule. The neighbor selected for improvement is then subjected to the improvement routine and the process is repeated until the point is reached when the makespan values for the neighbors cease to be lower than that for the parent schedule. At this point, the best heuristic solution for technique has been obtained and the search process is terminated.

APPENDIX B
MANUAL SOLUTIONS BY THE HEURISTIC ALGORITHMS

The solution technique of each heuristic algorithm is demonstrated by developing a schedule from the following matrix of processing times. This matrix was randomly generated for a 6 job by 4 machine problem.

Machine	Job Number					
Number	1	2	3	4	5	6
1	47	75	28	94	19	64
2	20	42	76	85	23	78
3	63	78	99	98	99	15
4	1	74	42	83	28	18

Random Generation with Extensive Search Algorithm (RGES)

Initially, 25 schedules are randomly generated and their makespan values are evaluated. The typical schedules are:

Schedule	Makespan
2-6-1-5-3-4	679
4-6-1-3-5-2	705

The schedule which yields the least makespan value is selected. This schedule is 5-3-4-2-6-1 and its makespan value is 514 time units. This schedule is then subjected to an improvement routine which forms neighbors by temporarily interchanging a pair of adjacent jobs. The new schedule is called a neighbor and if there are n jobs, then (n-1) neighbors can be formed. Thus, the neighbors derived from the above initial schedule are:

Schedule	Makespan
3-5-4-2-6-1	576
5-4-3-2-6-1	566
5-3-2-4-6-1	583
5-3-4-6-2-1	514
5-3-4-2-1-6	563

Typically, the neighbor which has the least makespan value is selected and then subjected to the improvement routine to form new neighbors. The process is repeated until the new neighbors fail to improve the solution. This has occurred in this solution because the best neighbor yielded a makespan value of 514 which is the same as the value for its predecessor. Thus, the search is terminated.

Heuristic Decomposition Algorithm (HD)

This technique partitions the jobs into 2 groups. One group is scheduled by an optimizing algorithm while the other group is sequenced by a job-pairing algorithm. The sizes of the groups depend upon the number of jobs which one can afford to solve by the optimizing algorithm. In this case, the two groups will each have three jobs.

Since there are less than 12 jobs in the group, the job-pairing algorithm determines the initial pair of jobs by selecting that pair which minimizes the sum of the first jobs' processing time and the machine idle time on the last machine for the pair. Since there are 6 jobs and any job can be in the first position, there are 30 different pairs to be considered. The pair which yielded the smallest sum was:

5-6

This sequence is augmented by that job which forms the partial sequence that minimizes the machine idle time on the last machine. Thus, each of the remaining jobs is tried in position 3 in order to identify that job. In this particular problem, job 1 was selected and the partial schedule became:

5-6-1

The idle times for the partial schedules are:

Partial Schedule	Idle Time
5-6-1	198
5-6-2	238
5-6-3	293
5-6-4	317

This partial schedule forms a synthetic job which is placed in the first position. The remaining jobs are then scheduled by an optimizing algorithm, the implicit enumeration technique [1]. The minimum makespan value was obtained from the following sequence:

3-4-2

Thus, combining both sequences, the overall schedule is:

5-6-1-3-4-2

which has a makespan value of 611 time units.

Rapid Access with Extensive Search (RAES) Algorithm

This technique develops a pseudo two-machine problem from the m-machine problem and solves the former using Johnson's algorithm [65]. New processing times are derived for the two-machine problem by using equations 3.5 and 3.6. They are displayed in the following matrix:

Machine	Job Number					
Number	1	2	3	4	5	6
1	375	656	580	910	371	538
2	280	689	645	890	474	337

Ex. When m = 4

j	1	2	3	4
m-j+1	4	3	2	1
T_{ij}	47	20	63	1
$(m-j+1)xT_{ij}$	188	60	126	1
Sum		375		

Johnson's algorithm forms a schedule from the above matrix by successively selecting the least processing time under consideration and assigning the associated job to the earliest available position if the time occurred on the first machine, or the latest available position if it occurred on the second machine. For example, the smallest processing time is 280 for job 1 on machine 2, therefore job 1 is assigned to the latest available position. The processing times for job 1 is then removed from consideration and the procedure is repeated. The schedule derived from this procedure is:

5-3-2-4-6-1

This has a makespan value of 583 time units.

In the second phase, the improvement routine which was used in the RGES algorithm to generate neighbors is employed. The following neighbors were generated:

Schedule	Makespan
3-5-2-4-6-1	580
5-2-3-4-6-1	518
5-3-4-2-6-1	514
5-3-2-6-4-1	547
5-3-2-4-1-6	518

The neighbor 5-3-4-2-6-1 generated the least makespan value. This is the same solution as the one yielded by the RGES algorithm. This is therefore a terminal solution because it was found, as in the case of the RGES algorithm, that new neighbors do not improve the solution.

Campbell, Dudek and Smith Algorithm (CDS)

This algorithm develops (m-1) pseudo two-machine problems from the m-machine problem and solves them using Johnson's algorithm. The schedule which yields the least makespan value is reported as the solution.

The new processing times for the pseudo two-machine problems are obtained from equations 3.2 and 3.3. The new processing times matrix for the (m-1) problems is as follows:

Problem 1

Machine	Job Number					
Number	1	2	3	4	5	6
1	47	75	28	94	19	64
2	1	74	42	83	28	18

Problem 2

Machine Number	Job Number 1	2	3	4	5	6
1	67	117	104	179	42	142
2	64	152	141	181	127	33

Problem 3

Machine Number	Job Number 1	2	3	4	5	6
1	110	195	193	277	141	157
2	84	194	217	266	150	111

The solutions to these problems are:

Problem	Sequence	Makespan
1	5-3-4-2-6-1	514
2	5-3-2-4-1-6	518
3	5-3-4-2-6-1	514

The best schedule is yielded by pseudo problems 1 and 3. The schedule is:

5-3-2-4-6-1

Its makespan value is 514.

Aggarwal and Stafford Algorithm (AS)

This algorithm consists of a 2 phase procedure. In the first phase, jobs are assigned priority positions on the machines in the first half of the process based on the increasing order of their processing times. For machines in the latter half of the process, assignments are based on the decreasing order of the processing times. The assignments are therefore as follows:

Machine Number	Position in Preferred Job Sequence 1	2	3	4	5	6
1	5	3	1	6	2	4
2	1	5	2	3	6	4
3	3	5	4	2	1	6
4	4	2	3	5	6	1

The matrix of job numbers is then partitioned into 4 sectors. The jobs bounded by the broken line in the NW and SE sectors have top priority for assignment to the positions under which they fall. Thus, the jobs are allocated to the positions as follows:

Position	Jobs Competing	Job Selected
1	5	5
2	3,5	3
3	1,2	1
4	2,5	2
5	1,6	6
6	1	4

When more than one job competes for a position, the job that formed the partial sequence which yields the lowest makespan value is selected. Thus, job 1 was selected over job 2 for position 3 because the partial sequence 5-3-1 has a smaller makespan value than partial sequence 5-3-2. Their respective values are 304 and 392. Positions 2, 4, 5 and 6 were filled by default.

This initial sequence is then subjected to an improvement routine by having the jobs in positions i and i+1 compete for position i. This is done by forming partial schedules and then determining the makespan values of these partial schedules. The process is repeated until no improvement is obtained in a pass.

The results are tabulated as follows:

Pass No. 1

Position	Competing Sequences	Makespan	Winner	Interchange
1	5-3	282	5	No
	3-5	303		
2	5-3-1	304	3	No
	5-1-3	345		
3	5-3-1-2	455	2	Yes
	5-3-2-1	393		
4	5-3-2-1-6	473	6	Yes
	5-3-2-6-1	411		
5	5-3-2-6-1-4	593	4	Yes
	5-3-2-6-4-1	547		
6	5-3-2-6-4-1	547	1 (by default)	

Since jobs were interchanged in the first pass, a second pass is made. The results are presented in a more condensed form as follows:

Pass No. 2

Position	Jobs Competing	Makespan Values	Winner	Interchange
1	5 vs 3	282 vs 303	5	No
2	3 vs 2	392 vs 360	2	Yes
3	3 vs 6	378 vs 452	3	No
4	6 vs 4	556 vs 517	4	Yes
5	6 vs 1	518 vs 518	None	No

Schedule yielded: 5-2-3-4-6-1 or 5-2-3-4-1-6

Pass No. 3

Position	Jobs Competing	Makespan Values	Winner	Interchange
1	5 vs 2	293 vs 322	5	No

The other positions will be the same as those derived from the 2nd pass; therefore no improvement is obtained from this pass and a terminal solution has been determined.

The schedule is either 5-2-3-4-6-1

or 5-2-3-4-1-6

both yielding a makespan value of 518.

APPENDIX C

RANDOM NUMBER SEEDS FOR PROCESSING TIMES

Problem Size	Random Number Seeds
4x4	243,246,735
4x7	6,665,554,441
4x10	4,781,121
6x4	9,547,312,571
6x7	45,387,111
6x10	4,500,002,241
8x4	7,755,883,333
8x7	5,274,125,937
8x10	4,521,785,337
9x4	9,477,711,111
9x7	4,445,556,235
9x10	321,456,789
10x4	6,666,666,663
10x7	4,466,882,221
10x10	453,777
20x15	1,311
20x30	5,741,221
20x45	4,000,521
20x60	5,234,777,799
40x15	10,055,333
40x30	11,115
40x45	4,422,000,099
40x60	77,785
60x15	8,607,341
60x30	12,365
60x45	3,472,358,715
60x60	24,577
80x15	19,221,941
80x30	7,700,000,111
80x45	888,173
80x60	18,571,741
100x15	1,002,009,753
100x30	3,004,001,357
100x45	2,468,775
100x60	3,697,531

APPENDIX C --Continued

RANDOM NUMBER SEEDS FOR RGES

Problem Size	Random Number Seeds
4x4	7,546,210,001
4x7	2,531,475,441
4x10	7,821,452,223
6x4	7,621,563,889
6x7	4,112,257,773
6x10	3,322,541,781
8x4	6,655,366,621
8x7	5,544,321,851
8x10	7,568,423,819
9x4	7,531,467,951
9x7	5,283,738,121
9x10	9,173,284,511
10x4	6,734,257,111
10x7	2,145,632,177
10x10	6,655,897,981
20x15	77
20x30	500,535
20x45	2,040,722,999
20x60	5,662,244,111
40x15	445,555
40x30	223,333
40x45	6,633,397,777
40x60	963,377
60x15	9,850,000,237
60x30	632,669
60x45	52,251,595
60x60	1,000,251,597
80x15	19,421,949
80x30	666,777
80x45	90,232,199
80x60	39,201,971
100x15	4,812,163,357
100x30	5,101,520,411
100x45	6,121,873
100x60	7,142,128,355

APPENDIX D

MEANS OF VARIABLES, SMALL PROBLEMS

NUMBER OF JOBS	NUMBER OF MACHINES	HEURISTIC ALGORITHM[a]	MAKESPAN	JOB WAITING TIME	MACHINE IDLE TIME	CPU TIME
4	4	1	389.	370.	376.	0.0414
4	4	2	419.	395.	441.	0.0112
4	4	3	390.	370.	373.	0.0179
4	4	4	395.	369.	386.	0.0093
4	4	5	391.	370.	378.	0.0128
4	7	1	572.	485.	1459.	0.0518
4	7	2	608.	511.	1530.	0.0167
4	7	3	572.	489.	1460.	0.0245
4	7	4	573.	462.	1445.	0.0146
4	7	5	576.	473.	1471.	0.0154
4	10	1	756.	548.	2902.	0.0605
4	10	2	819.	608.	3207.	0.0190
4	10	3	757.	540.	2879.	0.0303
4	10	4	761.	554.	2890.	0.0198
4	10	5	760.	542.	2928.	0.0173
6	4	1	493.	936.	453.	0.0665
6	4	2	556.	1014.	577.	0.0258
6	4	3	491.	944.	445.	0.0325
6	4	4	507.	981.	456.	0.0110
6	4	5	501.	927.	470.	0.0181
6	7	1	677.	1166.	1531.	0.0832
6	7	2	743.	1178.	1757.	0.0256
6	7	3	671.	1119.	1545.	0.0456
6	7	4	684.	1119.	1582.	0.0162
6	7	5	677.	1087.	1579.	0.0226
6	10	1	872.	1236.	3152.	0.1135
6	10	2	941.	1279.	3533.	0.0360
6	10	3	868.	1221.	3189.	0.0625
6	10	4	878.	1218.	3212.	0.0234
6	10	5	683.	1239.	3188.	0.0271
8	4	1	615.	1751.	425.	0.0987
8	4	2	669.	1802.	516.	0.0402
8	4	3	601.	1680.	381.	0.0509
8	4	4	614.	1773.	413.	0.0124
8	4	5	605.	1664.	401.	0.0240
8	7	1	817.	2044.	1612.	0.1371
8	7	2	893.	2088.	1837.	0.0472
8	7	3	795.	1943.	1560.	0.0710
8	7	4	809.	2016.	1611.	0.0195
8	7	5	814.	1988.	1642.	0.0323
8	10	1	1034.	2372.	3466.	0.1855
8	10	2	1129.	2366.	3972.	0.0536
8	10	3	1022.	2208.	3470.	0.1190
8	10	4	1034.	2295.	3518.	0.0275

APPENDIX D--Continued

MEANS OF VARIABLES, SMALL PROBLEMS

NUMBER OF JOBS	NUMBER OF MACHINES	HEURISTIC ALGORITHM[a]	MAKESPAN	JOB WAITING TIME	MACHINE IDLE TIME	CPU TIME
8	10	5	1043.	2161.	3535.	0.0382
9	4	1	648.	2131.	435.	0.1128
9	4	2	710.	2300.	556.	0.0826
9	4	3	634.	2096.	404.	0.0570
9	4	4	649.	2175.	436.	0.0132
9	4	5	649.	2084.	445.	0.0273
9	7	1	855.	2546.	1675.	0.1595
9	7	2	950.	2802.	1982.	0.1109
9	7	3	837.	2466.	1595.	0.0913
9	7	4	862.	2521.	1681.	0.0210
9	7	5	867.	2423.	1696.	0.0374
9	10	1	1083.	2885.	3585.	0.2097
9	10	2	1170.	3070.	3948.	0.1205
9	10	3	1058.	2788.	3471.	0.1285
9	10	4	1075.	2844.	3541.	0.0300
9	10	5	1088.	2796.	3640.	0.0469
10	4	1	696.	2593.	430.	0.1379
10	4	2	750.	2709.	544.	0.0994
10	4	3	682.	2565.	393.	0.0664
10	4	4	701.	2699.	438.	0.0138
10	4	5	693.	2501.	430.	0.0318
10	7	1	906.	3109.	1669.	0.2061
10	7	2	1001.	3342.	2029.	0.1205
10	7	3	886.	2969.	1646.	0.1088
10	7	4	900.	3093.	1720.	0.0227
10	7	5	918.	2977.	1729.	0.0425
10	10	1	1147.	3563.	3567.	0.2618
10	10	2	1247.	3707.	4117.	0.1437
10	10	3	1119.	3297.	3489.	0.1463
10	10	4	1134.	3445.	3513.	0.0387
10	10	5	1140.	3349.	3658.	0.0672

[a] 1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS

APPENDIX E

MEANS OF VARIABLES, LARGE PROBLEMS

NUMBER OF JOBS	NUMBER OF MACHINES	HEURISTIC ALGORITHM[a]	MAKESPAN	JOB WAITING TIME	MACHINE IDLE TIME	CPU TIME
20	15	1	2133.	15637.	9782.	1.7815
20	15	2	2395.	15142.	11498.	0.6031
20	15	3	2046.	14012.	9290.	1.3525
20	15	4	2101.	14812.	9639.	0.1649
20	15	5	2158.	14821.	10204.	0.3351
20	30	1	3159.	19625.	37258.	3.3694
20	30	2	3492.	18639.	41065.	1.0946
20	30	3	3078.	17855.	36696.	2.9770
20	30	4	3114.	18610.	37177.	0.4444
20	30	5	3229.	19280.	38916.	0.5542
20	45	1	4146.	22679.	80362.	5.1756
20	45	2	4521.	21888.	86234.	1.5633
20	45	3	4062.	20486.	79631.	4.3797
20	45	4	4082.	21063.	79771.	0.8607
20	45	5	4229.	21141.	82150.	0.8144
20	60	1	5129.	24956.	138839.	5.9420
20	60	2	5599.	24370.	147685.	2.0728
20	60	3	4994.	22837.	135714.	6.9151
20	60	4	5037.	23330.	138520.	1.4061
20	60	5	5202.	23816.	141923.	1.0943
40	15	1	3364.	58393.	12007.	9.5782
40	15	2	3838.	59660.	14920.	3.4815
40	15	3	3188.	54046.	10971.	9.1843
40	15	4	3299.	57252.	12022.	0.4410
40	15	5	3439.	56620.	12971.	1.4332
40	30	1	4558.	70799.	44759.	17.9269
40	30	2	5137.	71996.	51215.	6.4955
40	30	3	4384.	64790.	43088.	18.1837
40	30	4	4483.	68656.	44470.	1.1459
40	30	5	4670.	68120.	47445.	2.4884
40	45	1	5646.	79956.	96109.	27.0170
40	45	2	6232.	79441.	105151.	9.4581
40	45	3	5470.	73510.	92632.	26.7491
40	45	4	5565.	77158.	95230.	2.0647
40	45	5	5830.	77159.	100497.	3.3762
40	60	1	6666.	86766.	162433.	36.1783
40	60	2	7284.	87300.	174055.	12.2574
40	60	3	6494.	80976.	157802.	37.3267
40	60	4	6566.	83928.	160299.	3.3709
40	60	5	6883.	84828.	167789.	4.2627
60	15	1	4559.	126956.	14391.	24.3108
60	15	2	5184.	129482.	17882.	10.7876
60	15	3	4285.	118732.	12418.	24.8647
60	15	4	4420.	125073.	13683.	0.8767

APPENDIX E--Continued

MEANS OF VARIABLES, LARGE PROBLEMS

NUMBER OF JOBS	NUMBER OF MACHINES	HEURISTIC ALGORITHM[a]	MAKESPAN	JOB WAITING TIME	MACHINE IDLE TIME	CPU TIME
60	15	5	4642.	123721.	15201.	3.2443
60	30	1	5876.	148930.	51399.	52.6908
60	30	2	6576.	153474.	58690.	20.0235
60	30	3	5620.	139426.	48060.	55.5813
60	30	4	5730.	145212.	50711.	2.1602
60	30	5	6108.	148551.	55173.	6.2971
60	45	1	7039.	166312.	107025.	76.0022
60	45	2	7752.	170515.	117418.	29.6856
60	45	3	6785.	154664.	102440.	86.6194
60	45	4	6911.	162037.	105532.	3.9134
60	45	5	7278.	165624.	112224.	9.1589
60	60	1	8159.	180081.	180535.	104.2851
60	60	2	8921.	185196.	194957.	38.8078
60	60	3	7941.	169689.	176263.	114.1369
60	60	4	8034.	175259.	179417.	5.9762
60	60	5	8473.	180503.	191676.	10.4755
80	15	1	5761.	219074.	16072.	47.3982
80	15	2	6541.	229048.	20572.	24.9037
80	15	3	5456.	208881.	13861.	51.5034
80	15	4	5592.	216346.	15147.	1.3940
80	15	5	5919.	217502.	17681.	5.8063
80	30	1	7151.	253199.	57079.	103.2424
80	30	2	7960.	264610.	64705.	46.8903
80	30	3	6830.	240377.	53067.	121.9248
80	30	4	6973.	248879.	55736.	3.3992
80	30	5	7379.	253283.	60898.	11.1021
80	45	1	8424.	281558.	119412.	160.3882
80	45	2	9206.	291213.	129461.	68.2706
80	45	3	8095.	265556.	113308.	176.0837
80	45	4	8245.	274009.	117715.	5.9653
80	45	5	8708.	283313.	124211.	16.2585
80	60	1	9580.	307176.	198489.	219.5613
80	60	2	10413.	311539.	213266.	90.0250
80	60	3	9279.	288061.	191834.	219.2510
80	60	4	9398.	297564.	195675.	9.0730
80	60	5	9879.	303372.	205579.	22.2463
100	15	1	6971.	338556.	17811.	81.0668
100	15	2	7767.	353914.	22234.	46.5136
100	15	3	6616.	321309.	15138.	82.1661
100	15	4	6769.	335204.	16660.	2.1123
100	15	5	7174.	336229.	19285.	8.8689
100	30	1	8379.	385197.	62453.	180.1714
100	30	2	9326.	399291.	69468.	87.5800
100	30	3	8000.	364973.	57274.	191.5453

APPENDIX E--Continued

MEANS OF VARIABLES, LARGE PROBLEMS

NUMBER OF JOBS	NUMBER OF MACHINES	HEURISTIC ALGORITHM[a]	MAKESPAN	JOB WAITING TIME	MACHINE IDLE TIME	CPU TIME
100	30	4	8166.	377615.	59848.	5.0319
100	30	5	8732.	386514.	66615.	17.8245
100	45	1	9698.	425371.	128632.	292.4789
100	45	2	10715.	438575.	140816.	127.9308
100	45	3	9327.	401561.	121226.	319.0515
100	45	4	9468.	414549.	124491.	8.5838
100	45	5	10092.	425777.	134792.	26.5248
100	60	1	10916.	454859.	211856.	384.4664
100	60	2	11911.	470045.	229849.	169.2427
100	60	3	10525.	427329.	201243.	461.3044
100	60	4	10733.	444810.	208589.	12.8760
100	60	5	11369.	459740.	222895.	37.0008

[a] 1=RGES; 2=HD; 3=RAES; 4=CDS; 5=AS