EQUIPMENT SELECTION AND TASK ASSIGNMENT FOR MULTIPRODUCT ASSEMBLY SYSTEM DESIGN

by

Carol Anne Holmes
B.S., Duke University
(1983)

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN OPERATIONS RESEARCH

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

January 1987

Signature of Author__

Operations Research Center
Alfred P. Sloan School of Management
January 16, 1987

Certified by___

Stephen C. Graves
Thesis Supervisor

Accepted by____

Thomas L. Magnanti
Cochairman Operations Research Center

EQUIPMENT SELECTION AND TASK ASSIGNMENT FOR MULTIPRODUCT ASSEMBLY SYSTEM DESIGN

by

CAROL ANNE HOLMES

Submitted to the Alfred P. Sloan School of Management on January 16, 1987 in partial fulfillment of the requirements for the Degree of Master of Science in Operations Research.

ABSTRACT

Given a set of products to be assembled, a fixed sequence of assembly operations for each product, and a list of available resource types to accomplish those operations, a decision is required as to which resource types to select and which operations to assign to each resource type so as to meet production requirements at the minimum total system cost.

An optimization method for this equipment selection problem has been developed. The method seeks to implicitly enumerate candidate work stations (operation groupings) using an adaptation of the binary counting method for partially ordered task sets. The cost of each candidate work station is evaluated and the least cost resource type for the work station is selected. A shortest path problem is solved to find the set of work stations which accomplishes the given sequence of assembly operations for the least total system cost.

This research is part of an ongoing study of assembly system design at the Charles Stark Draper Laboratory. The equipment selection problem is the second phase of a two phase approach to assembly system design developed by CSDL which emphasizes the integration of product design and assembly line design. In addition to describing our solution method to the equipment selection problem in detail, this thesis is intended to justify the need for this integrated approach to assembly system design and to compare our work to related work discussed in the literature.

Thesis Supervisor: Professor Stephen C. Graves

Title: Associate Professor, Management

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to Professor Stephen Graves of MIT and Dr. Daniel Whitney of the Charles Stark Draper Laboratory for their continuous support and guidance as they supervised the progress of this thesis.

I would also like to thank Richard Gustavson of the Charles Stark Draper Laboratory for his help with the implementation and testing of the computer algorithm.

Finally, I would like to extend a special thanks to the students, faculty, and administration of the Operations Research Center who helped to make my years at MIT both enriching and enjoyable.

This research was supported by the National Science Foundation under Grant DMC-84-17949 made to the Charles Stark Draper Laboratory, Inc.

TABLE OF CONTENTS

CHAPTER 1 - INTRODUCTION

The traditional approach to assembly system design has been assembly line balancing. This approach was appropriate for a labor-intensive assembly environment where the primary concerns were distributing work evenly among assembly workers and keeping labor costs to a minimum. Today that labor intensive assembly environment is disappearing. Manufacturing has entered an age of automation and flexible assembly which better meets the demands of fluctuating markets and continuous innovation. New equipment choices provide better quality, manageability and reliability than the more traditional manual assembly methods. In this new environment, robots offer the extreme in flexibility and adaptability. Acquisition of this automated equipment, however, requires a substantial capital investment. The decision to automate will be dictated by total system costs. Line balancing is not of primary concern in such a system. This suggests the need for a means of evaluating new equipment choices and a method for designing a minimum cost assembly system. This equipment selection problem is addressed in this thesis.

Specifically, given a set of products to be assembled, given the exact order of assembly operations for each product, and given a list of candidate resources available to complete the operations, a decision is required as to which resource types to select and which operations to assign to each resource so as to meet production requirements for the set of products at the minimum total system cost.

The products to be assembled may be a single product, a family of related products, or different models of the same product. The variation between products may be one or many task differences, or it may be a complete subassembly that one product requires which the others do not.

The assembly of each product entails a prespecified sequence of assembly operations where an assembly operation(task) is any well-defined unit of work involved in the assembly of a product. Possible tasks are inserting a bolt, inspecting a product, or attaching a subassembly.

Resources to perform the operations include humans, fixed automation such as a transfer line station, and programmable machines such as robots. A resource type may have the capability of performing all the assembly tasks, or it may be a special-purpose machine capable of doing only a small subset of the operations. The annualized fixed cost of each resource and any necessary tooling are explicitly accounted for in the computation of total cost. Associated with each resource is the time needed to complete each operation and the time needed to change tools.

Given this information, the problem is to design a minimum cost assembly system with sufficient capacity to meet production demands. An assembly system is a grouping of tasks into work stations where an available resource is assigned to each station. Production demands are given as the expected annual volume of product to be assembled.

Production requirements are met by imposing a cycle time constraint on the system. The cycle time, defined as the frequency(seconds/unit) with which units of product must come off the line to meet production demands, is given by:

$$\text{Cycle Time(seconds/unit)} = \frac{\text{Total Available Time (seconds)}}{\text{Production Demand (units)}}.$$

Total production time per unit for a station is the sum of the operation times and the tool change times for the tasks assigned to the station. A system has sufficient production capacity to complete the assembly of the products if each work station satisfies the condition that total production time per unit for the station is less than or equal to the cycle time per unit for the system.

The work flow of the line is restricted to a linear floor layout where work moves from station to station as, for example, in a conveyor belt transport system. The transport system is not explicitly modeled, but station-to-station move time (transfer time) is accounted for as a factor in determining the available cycle time for the system.

The following facts are emphasized about the problem to be solved. The goal is equipment selection for minimum cost assembly system design rather than assembly line balance. In automated assembly, fixed costs often dominate labor costs in the design of an assembly system, in which case the least cost system may be highly unbalanced.

Specifically addressed in the current formulation of the equipment selection problem is multiproduct assembly system design, a special case of which is the single product case. Adding the multiproduct dimension allows for the exploration of flexible assembly systems with more than one product type being assembled on a single assembly line. When assembly system design is restricted to the single product case, the full potential of the flexibility of the automated equipment may not be realized.

The solution formulated for the equipment selection problem is part of a two phase approach to assembly system design developed by the Charles Stark Draper Laboratory (CSDL) which stresses a global approach to assembly system design. The desired goal of CSDL's method is complete integration of product and assembly line design. A fixed sequence of assembly tasks for each product is taken as given for the equipment selection problem. In Phase One of CSDL's approach, candidate assembly sequences for a product are enumerated and represented graphically by the liaison sequence analysis method developed by DeFazio and Whitney [2]. Phase Two solves the equipment selection problem to find the corresponding least cost assembly system for the assembly sequences generated by Phase One.

The solution procedure for the equipment selection problem is an optimization algorithm. All possible candidate work stations for the assembly system are implicitly enumerated and evaluated. The solution found is the minimum cost assembly system. CSDL has developed an efficient heuristic approach to the same problem [6]. Their method can not guarantee that the least cost solution will be found, but as with most heuristic methods, it requires less computational time and storage than the optimization method.

The new optimization method is programmed in BASIC on the IBM PC XT for maximum portability. Compiled run times to date have all been under 3 minute for problems with a maximum of 28 tasks, 3 products and 5 resources. Because run times are short, the program can effectively be used as a tool to test out numerous assembly design possibilities using different input parameters or different assembly sequences for the products.

This thesis is organized into five additional chapters. Chapter 2 is a summary of relavant research. Also presented in this chapter is a review of the related work done at CSDL.

Chapter 3 presents the solution to the multiproduct equipment selection problem in detail using a sample problem to illustrate the method.

Chapter 4 describes the software that has been developed including implementation details, a discussion of the flexibility in the use of the program, and bounds on the maximum problem size that can be expected.

Chapter 5 presents in detail the results of a realistic design problem from the automobile industry.

Chapter 6 presents possible extensions of this research including a discussion of unresolved issues concerning the definitions of cycle time and system reliability and enhancements to the system that might be considered.

CHAPTER 2 - RESEARCH RELATED TO THE MULTIPRODUCT EQUIPMENT SELECTION PROBLEM

Surprisingly little research has been done on equipment selection for multiproduct assembly system design. Historically, because manual assembly has dominated manufacturing, the assembly line balancing (ALB) problem has been the accepted approach to designing effective work station configurations. This chapter begins by exploring the relation between the ALB problem and the equipment selection problem. Three versions of the ALB problem discussed in the literature are presented. Also discussed is the Generative Process Planning method developed by Halevi [8] which is an equipment selection problem with special application to the metal cutting industry. For each problem presented a discussion of the problem description, the problem setting, the solution method, and a comparison to the multiproduct equipment selection problem (hereafter referred to as MESP) is given. Except for the work currently being done at the Charles Stark Draper Laboratory (CSDL), to the best of our knowledge, no one has addressed precisely the same equipment selection problem that is presented in this thesis. To provide background and motivation for the current formulation of the equipment selection problem a summary of the related work done at CSDL over the past decade is presented.

2.1 EXACT AND HEURISTIC METHODS FOR SIMPLE ASSEMBLY LINE BALANCING

Problem Description: The simple assembly line balancing problem seeks to assign assembly tasks to work stations so as to minimize the idle time of a system for a fixed cycle time and a fixed set of precedence relations among the tasks. By definition, the most balanced system is the system for which the idle time has been minimized or correspondingly the number of stations has been minimized.

Problem Setting: In his "Survey of Exact Algorithms for the Simple Assembly Line Balancing Problem," Baybars [1] specifies the following assumptions of the simple ALB problem.

1) All input parameters are known with certainty.
2) A task can not be split among two or more stations.
3) Tasks cannot be processed in arbitrary sequences due to technological precedence relationships.
4) All tasks must be processed.
5) All stations are equipped and manned to process any one of the tasks (i.e. It is assumed that the fixed and variable costs associated with all the stations are the same and, therefore, they need not be considered in the model.)
6) Task process times are independent of the station at which they are performed.
7) Any task can be processed at any station.
8) The total line is considered to be serial with no feeder or parallel subassembly lines.
9) The assembly system is assumed to be designed for a unique model of a single product.
10) The cycle time is given and fixed and the goal is to minimize idle time.

or

11) The number of stations is given and fixed and the goal is to minimize the cycle time.

Any problem meeting assumptions 1 thru 9 and either 10 or 11 can be classified as a simple ALB problem.

Solution Method: The reader is referred to Baybars' survey for a discussion of exact (optimization) methods for the simple ALB problem. Baybars discusses a variety of integer programming and dynamic programming formulations for the problem and summarizes computational results for the methods. Talbot, Patterson, and Gehrlein's "Comparative Evaluation of Heuristic Line Balancing Techniques" [14] presents a summary of heuristic methods for the ALB problem. Specifically, Talbot et al. address the problem of determining a good approximation to the minimum number of work stations required for a given cycle time. Twenty-six list processing and optimum-seeking techniques (limited by computational time restrictions) are compared. The computational results of four test data sets are presented and discussed.

Comparison to the MESP: The MESP retains assumptions 1 thru 4 and 8 of Baybars' list for the simple ALB problem and relaxes Assumptions 5, 6, 7, and 9. Assumption 5 states that all stations are alike. The MESP specifically considers the fixed and variable costs of different resource types in determining the configuration of the system. Whereas assumption 6 states that each station takes the same time to complete a task and that all stations can do all the tasks, task times are resource specific for the MESP, and not every resource can do all the tasks. Assumption 9 limits the simple ALB to one product, but the MESP specifically addresses multiproduct assembly. Cycle time is given as in assumption 10, but the goal of MESP is total cost minimization rather than the minimization of idle time. The simple ALB problem is appropriate for manual assembly where labor constraints and costs are the critical factors in assembly design. The MESP approach is more appropriate for automated assembly where equipment selection and total cost minimization are the factors.

In the next two sections problems are presented that are relaxations of the simple ALB problem but that retain the goal of line balancing. The first problem considers ALB with processing alternatives, and the second considers ALB for mixed model assembly.

2.2 ASSEMBLY LINE BALANCING WITH PROCESSING ALTERNATIVES

Problem Description: ALB with processing alternatives relaxes Assumptions 5, 6, and 7 that state that manufacturing methods have been predetermined. The problem takes as given the set of facilities with least fixed cost that are sufficient to operate a line. Processing alternatives (alternative facilities) are explored to determine if, for an incremental fixed cost, the balance of the line can be improved and the total cost reduced. A basic assumption of the model is that even though a processing alternative reduces the work content of a set of tasks, a cost saving is not achieved unless the resulting line balancing assignment reduces either the cycle time or the number of stations required.

Problem Setting: Pinto, Dannenbring and Khumawala [12] propose ALB with processing alternatives as a means to explore the tradeoff between labor and capital intensive processing alternatives. They emphasize that their method provides a way to consider the selection of manufacturing alternatives and task assignments jointly, rather than as separate decisions.

Solution Method: The problem is formulated as an integer program and is solved using a branch and bound procedure. The objective is to minimize the annualized cost of the line. Given an initial set of least fixed cost facilities, all combinations of a given list of processing alternatives are considered. At each step of the branch and bound procedure, a feasible set of processing alternatives is selected and task assignments are made using a line balancing heuristic. The branch and bound procedure continues until the optimal (least cost) combination of processing alternatives has been found.

Comparison to the MESP: ALB with processing alternatives considers only single product assembly system design. As in the simple ALB problem, this formulation assumes that the least cost system is the most balanced system and is, therefore, limited by the applicability of line balancing. The MESP considers the broader criterion of minimizing total system cost, irrespective of line balance. Nevertheless, ALB with processing alternatives has been successfully applied to the redesign of a production facility in the automotive industry. It is an appropriate means of doing a simulation type analysis of the tradeoff between fixed facility costs and labor costs in a manual setting where line balancing is a valid approach.

2.3 LINE BALANCING - SEQUENCING FOR MIXED-MODEL ASSEMBLY

Problem Description: Mixed-model assembly relaxes Baybars' assumption 9 of simple ALB and considers the assembly of more than one model (product) on a single assembly line. The problem addressed by Thomopolous in "Line Balancing - Sequencing for Mixed Model Assembly" [15] is a standard ALB problem with the added dimension of multiproduct assembly. The objective is to assign operations to workers so as to balance the workload and minimize manpower requirements when there is more than one model of a product to be assembled on a line.

Problem Setting: Mixed-model ALB differs from simple ALB in that the work content of each model may be different. If simple ALB techniques are used, there may be flow imbalances as the different models move down the line. Developed in 1967, Thomopolous' method assumes a manual assembly system. Cost penalties are assigned to "inefficiencies" such as idleness, work deficiencies, utility work (assigning an extra person to a task) and work congestion when measuring the total cost of the solution.

Solution Method: The solution presented is an adaptation of the heuristic ALB method developed by Kilbridge and Webster [10] for single model ALB. Kilbridge and Webster balance the work load of an assembly line on the basis of the individual times needed to complete one unit of an operation. Operations are assigned to workers so as to give an equal amount of work to each worker while meeting a cycle time constraint to ensure that production demands are met. Thomopolous adapts this method to mixed-model assembly by using total operation time (individual operation time multiplied by the sum of the units of each model to be assembled) rather than individual operation times to make the assignment of operations to workers. Doing so accounts for the differences in work content of the models in the balancing process. In Thomopolous' method, work is assigned on the basis of the time available per worker per shift rather than on a cycle time basis to accomodate the use of total operation time rather than individual operation time. The work content for a person-shift is total work content divided by the minimum number of workers needed, where total work content is based on production demands and individual operation times.

Comparison to the MESP: In a manual setting it is possible to know before configuring the assembly line what the total work content of the mixed model assembly will be because only one resource type is considered. In the MESP the choice of resources and, therefore, the total work content is not known until after the assembly system has been designed. In general, line balancing for mixed-model assembly is limited, as the other ALB techniques, to a manual assembly setting, and does not consider equipment selection.

2.4 GENERATIVE PROCESS PLANNING (GPP)

Problem Description: Developed by Halevi and discussed in his book, <u>The Role of Computers in Manufacturing Processes</u> [8], GPP is the closest problem formulation to MESP that we have found. Given a list of operations to complete and a list of available machines, the problem is to decide which machines to use, which operations to perform on each machine, what the sequence of operations should be, and what cutting conditions should be used. The optimization criterion is either cost minimization or production maximization.

Problem Setting: Halevi has developed a unified approach to manufacturing planning which he calls "Hal Technology" (Hal being the Hebrew word for whole). His approach "supplies computer services to each phase of the manufacturing cycle independently, while maintaining a data base that serves as a single source for all company activities." GPP is one phase of the Hal Technology. It is a two part problem of which machine selection and task assignment is the second part. Part one is an engineering design problem which determines what process should be used based on engineering constraints. Part two adjusts the process to the available machines and finds an optimal configuration of the assembly line. This two part approach to process design is similar to the two part approach to assembly system design used at CSDL. Because GPP was developed for the metal cutting field, there is heavy emphasis on cutting conditions, which are of no concern in assembly system design.

Solution Method: Halevi's machine selection and task assignment problem is formulated as an allocation(assignment) problem and solved by a variation of dynamic programming. An allocation problem takes as given a number of jobs to be completed and a finite number of resources available to complete the jobs and assigns resources to jobs to minimize costs or maximize returns. An operation/resource matrix is created where each cell represents either the cost or time associated with assigning a job to a particular resource. Operations in the matrix are arranged according to the order in which they must be completed. Dynamic programming is used to make the operation/resource assignment by finding the least cost path through the matrix.

Halevi varies the allocation problem to accomodate precedence relations rather than a fixed ordering of the operations and to allow for the addition of transfer times between the machines. Each operation is assigned a priority number based on the partial ordering specified by the precedence relations. Operations are then arranged in the matrix by priority number and ties are broken arbitrarily. Halevi's version of the dynamic programming routine checks at each step to see if a new machine is being used and adds a transfer time between machines when necessary. The routine allows for the reordering of tasks during optimization if 1) the transfer time between machines could be eliminated by doing so and 2) the priority numbers of the operations are not violated. Because the solution is heavily dependent on the ordering of operations in the matrix, the problem is solved using both forward and backward recursion and the best solution is selected.

Comparison to the MESP: GPP takes as input precedence relations between the tasks rather than fixed sequence operations as in MESP. However, Halevi's assignment of priority numbers and ordering of tasks in the operations/resource matrix significantly limits the task sequence possibilities. GPP is intended for process planning with applications such as sheet metal, welding, molding textiles or chemical processes. As Halevi states, an appropriate criterion for process planning is cost minimization or production maximization. In assembly system design the objective is to minimize cost for a given production level. Halevi's method provides no means for assuring that a certain production level is met if the optimization criterion is chosen to be cost minimization. Halevi accounts for cost variations due to differences in batch sizes by adjusting the transfer time between machines. In particular, transfer times are lower for larger batches due to economies of scale. The formulation of the equipment selection problem that Halevi presents has many similarities to the single product case of the MESP. Halevi makes no attempt to deal with the multiproduct case.

2.5 RELATED RESEARCH AT THE CHARLES STARK DRAPER LABORATORY (CSDL)

For almost a decade now the Manufacturing Automation and Computation Department at CSDL has been doing fundamental research in the design and implementation of decision support systems for assembly system design. This section reviews the progression of research on the equipment selection problem that has led to the current formulation of the MESP. The discussion includes both the optimization and heuristic methods that have been developed at CSDL.

Optimization methods for the equipment selection problem have been limited to the single product case prior to the MESP. Graves and Whitney [5] originally presented the single product case in 1979. Similar to the MESP, the stated goal of the problem was to select equipment and make task assignments so as to minimize the sum of fixed and variable costs. They assume a fixed sequence of tasks as in the MESP, but they permit non-serial line layouts in which an assembly unit may return more than once to a given station. Their model did not account explicitly for tool change times and tool costs, which was a serious drawback to the model. The problem was formulated as a mixed integer program and was solved using branch and bound and a subgradient optimization procedure.

In 1981, Graves and Lamar [4] extended the formulation of Graves and Whitney to explicitly include tool change times in the formulation. The problem was formulated as an integer program and solved by finding upper and lower bounds for a linear relaxation of the problem. The integer programming problem as formulated was a very large problem whose computational requirements grew exponentially with the number of candidate resources. As a result of allowing unresticted floor layouts for the problem, the solutions found by both of these early formulations were not necessarily physically realizable.

The present MESP formulation was intended to address the limitations of the two earlier methods. In particular the goals were:

- Guarantee the feasiblity of the layout by restricting the system to a linear floor layout.
- Explicitly model tool costs.
- Develop a computationally efficient model that can be implemented on a PC.
- Add the multiproduct dimension to the problem.

In the next chapter the details of the current optimization approach to the MESP are discussed.

Paralleling the evolution of optimization methods for the equipment selection problem at CSDL has been the design and implementation of heuristic methods by Gustavson [6]. Gustavson has developed programs which find very good (optimal or near optimal) solutions to both the single and multiple product equipment selection problem. As in any heuristic method, the programs can not guarantee that an optimal solution will be found. One reason for developing an optimization approach to the MESP was to calibrate the effectiveness of Gustavson's heuristic methods. In most single product test cases, the two methods find the same solution. Gustavson's multiple product heuristic method is still being tested.

CHAPTER 3 - DETAILS OF THE OPTIMIZATION METHOD

In this section an optimization method for the equipment selection problem is described. The optimization criterion is to find that assembly system which is least cost among all design possibilities. The first step of the solution enumerates all candidate work stations for the system and selects the least cost resource type for each station. Then, the least cost assembly system is identified as that set of non-overlapping work stations which has the least total cost. This, as it turns out, is a Shortest Path problem which we solve using Dijkstra's algorithm [3]. This chapter describes the solution procedure in detail using a sample problem as a means of explanation.

Consider the following assembly problem. A product to be assembled has two different model types. Model A requires tasks (1,2,3,5,6,8,9,10,12) for assembly, while Model B requires tasks (1,2,4,5,6,7,10,11,12) for assembly. For each model, the tasks must be completed in the exact sequence given. However, between models there is flexibility in the order in which tasks may be assigned to work stations. For example, since Model A does not require task 4 and Model B does not require task 3, there is no fixed order between these tasks. This, as a result, is a partially ordered set of tasks which must be accomplished to complete the assembly of both products. This partially ordered task set is represented by the precedence constraints shown in the network diagram in Figure 3.1.

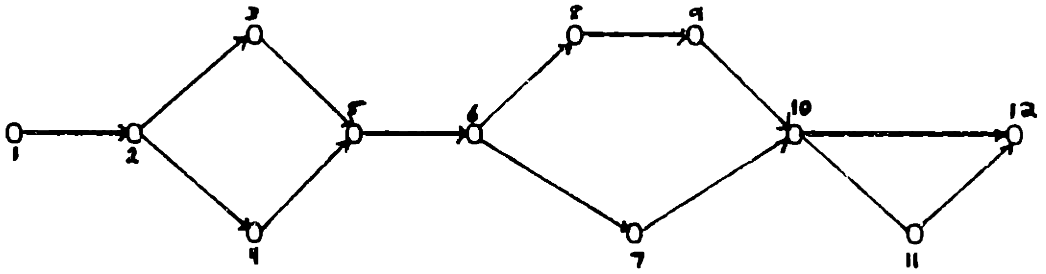


Figure 3.1

In addition to this task data, production demands for each product and information concerning the candidate resource types to complete the necessary tasks are given. For the example problem the production demand is 216,000 units per year for each model. Figure 3.2 summarizes the resource data. As shown, for each resource the operational times, tool numbers and annualized tool costs needed to complete each task are given. It is important to note that a resource may not be able to perform all tasks. A particular resource may be able to do only a small subset of the given task set. Also listed in Figure 3.2 are the annualized fixed costs and the variable cost per hour associated with each resource. This variable cost includes both the labor cost and the variable operating cost for the resource. The last item given in the resource data is the time needed by each resource to switch tools. This tool change time is assumed to be resource specific, but does not depend on the task or tool. However, it is possible to extend this solution approach to permit this.

	RESOURCE TYPE 1			RESOURCE TYPE 2		
Task	Operation Time (seconds)	Tool #	Tool Cost ($)	Operation Time (seconds)	Tool #	Tool Cost ($)
1	5.6	100	11000	---	---	----
2	3.6	120	8000	2.4	220	8000
3	3.6	121	3000	2.4	221	3000
4	1.8	121	3000	1.8	221	3000
5	1.8	121	3000	1.8	221	3000
6	3.6	131	8000	2.4	231	3000
7	3.6	141	7000	3.0	241	7000
8	2.0	142	2000	2.0	242	2000
9	4.0	150	7000	3.6	250	7000
10	7.2	160	4000	7.2	260	4000
11	5.4	170	10000	---	---	----
12	5.4	170	10000	---	---	----
Annualized fixed cost	$40,000			$50,000		
Variable cost/hour	$4.8			$5.3		
Tool change time(seconds)	2.0			2.0		

Figure 3.2

For this problem total time available for production is computed as:

(240 days/year)(1 shift/day)(8 hours/shift) =

1920 hours/year

Because there is more than one product type, the total available time must be divided between the products. One half is selected as the fraction of time for each product in this example, since the same quantity of each product is being produced and since the work content of each product is about the same. The selection of this fraction is actually a complex issue for which we defer discussion until the next chapter.

Once the annual volume for each product has been selected, the cycle time for each product can be computed by the equations:

$$\text{Cycle Time (product 1)} = \frac{(1920 \text{ hours/year})(3600 \text{ seconds/hour})(1/2)}{(216{,}000 \text{ units/year})}$$

$$= 16 \text{ seconds/unit}$$

$$\text{Cycle Time (product 2)} = \frac{(1920 \text{ hours/year})(3600 \text{ seconds/hour})(1/2)}{(216{,}000 \text{ units/year})}$$

$$= 16 \text{ seconds/unit}$$

The cycle time is the rate at which products must come off the assembly line to ensure that production demands are met. The station-to-station move time (transport time) for a system is the time required for a unit of product to move between work stations. Because this is time that is not available for assembly, the available cycle time is effectively the cycle time, as computed above, minus the transport time. For this example the transport time is 2 seconds so that the available cycle time is reduced to 14 seconds per unit.

Given the above data, the first step of the solution is to enumerate all candidate work stations for the system. A candidate work station is a group of tasks which satisfies the precedence relationships among the tasks and satisfies the cycle time constraint. To begin, consider again the graphical representation of the task set.

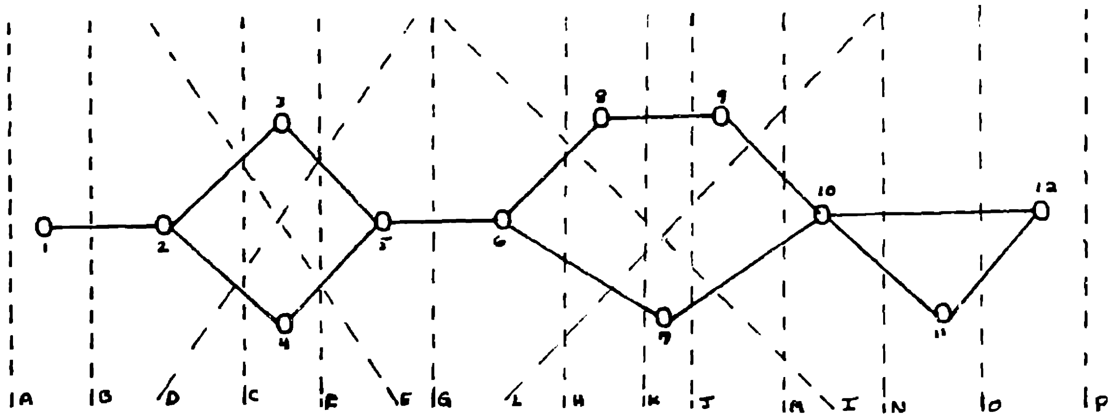


Figure 3.3

Define a cut set as a group of tasks such that for any task in the set, all the predecessors of that task are also in the set. Each dashed line in Figure 3.3 is a "cut" that determines a cut set. For example, cut E determines the cut set (1,2,4) and cut K determines the cut set (1,2,3,4,5,6,8). A candidate work station can now be represented as the difference between two cut sets. For instance, the work station {2,3,4,5} is given by G - B. A work station may be identified by more than one pair of cuts. For instance, the work station {4} is given by E - D, by E - C, and by F - D. In general, to generate all candidate work stations, all pairs of cut sets in which one cut set is a subset of the other are enumerated. For example, the pair (A,G) is the work station (1,2,3,4,5) and the pair (E,G) is the work station (3,5). Pairs such as (D,E), where cut set D is not a subset of cut set E, need not be considered since another pair will generate the same work station.

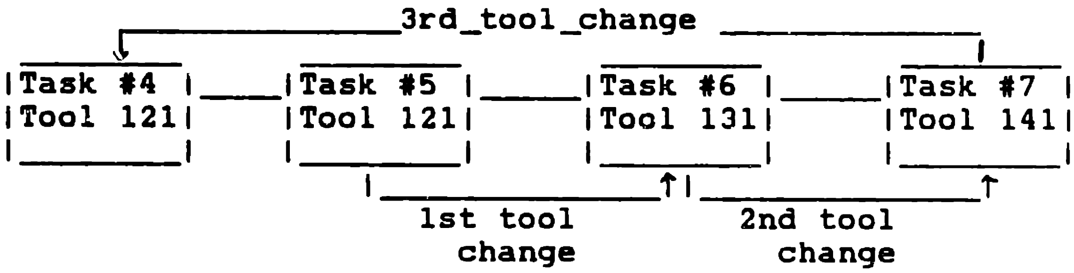
To enumerate candidate work stations, an efficient algorithm for enumerating the cut sets in a network is needed. The binary counting method for enumerating all subsets of a partially ordered set is such an algorithm. Define the vector m such that m(j) = 1 if task j is in the cut set and m(j) = 0 otherwise. The algorithm for enumerating all cut sets m is:

```
Let m(j) = 0 for all j
Loop:Find the smallest positive integer j
       for which m(j) = 0 and call it i.
       (If m(j) = 1 for all j then all
        subsets have been enumerated.)
     Set m(i) = 1.
     For j = i-1 to 1 step -1
        If  m(j) = 1 and j is such that no
        successor of j is a member of the
        current set m, then let m(j) = 0.
     Next j
     The vector m now represents a
       feasible subset.
     Go To Loop:
```

Following the implementation of the algorithm, the array of vectors m repesents the cut sets. The candidate work stations are then enumerated by considering pairs of these cut sets.

Not every work station that satisfies the precedence constraints is a feasible work station. A candidate work station is feasible if it satisfies the condition that at least one resource type is able to accomplish all of the tasks within the available cycle time.

The time to complete a set of tasks, which are assigned to one work station, is the sum of the task times and any necessary tool change times. If the tool change time occurs simultaneously with the station-to-station move time, then only the amount by which the tool change time exceeds the station-to-station move time is added to the total station time. As an example consider the pair of cuts (D,I) which is the candidate work station (4,5,6,7). Note that this work station does tasks (4,5,6,7) for product 2, but only tasks (5,6) for product 1. Figure 3.4 illustrates how the tool change time for resource 1, product 2 is calculated for this work station.



Tool change time for resource 1 is 2 seconds.
Transport time for resource 1 is 2 seconds.
Since the 3rd tool change occurs during the station to station move time, this tool change is not added to the total tool change time.
Total tool change time is 4 seconds for this example.

Figure 3.4

Below, the feasibility of workstation (4,5,6,7) for resource type 1 is considered. (See Figure 3.2 for task times.)

Resource 1	Product	Operation Time(secs)	Tool Change Time(secs)	Total Station Time(secs)
	1	5.4	2.0	7.4
	2	10.8	4.0	14.8

Resource 1 is feasible for product 1 since the total station time, 7.4 seconds, is less than the available 14 seconds. The total station time for product 2, 14.8 seconds, is greater than 14 seconds, so resource type 1 is infeasible for product 2. Resource type 1 is labeled as infeasible for this work station since it is not feasible for both products. The computation of total station time for resource type 2 is:

Resource 2	Product	Operation Time(secs)	Tool Change Time(secs)	Total Station Time(secs)
	1	4.2	2.0	6.2
	2	10.0	4.0	14.0

The feasibility of both products in this case implies that resource 2 is feasible for the work station. Clearly then, resource 2 is the least cost resource type for the work station since it is the only feasible resource type.

As a second example consider the work station (3,4,5) which is feasible for both resources.

	Product	Operation Time(secs)	Tool Change Time(secs)	Total Station Time(secs)
Resource 1	1	5.4	0	5.4
	2	3.6	0	3.6
Resource 2	1	4.2	0	4.2
	2	3.6	0	3.6

The work station cost for each resource type must be computed to select the least cost resource for this work station.

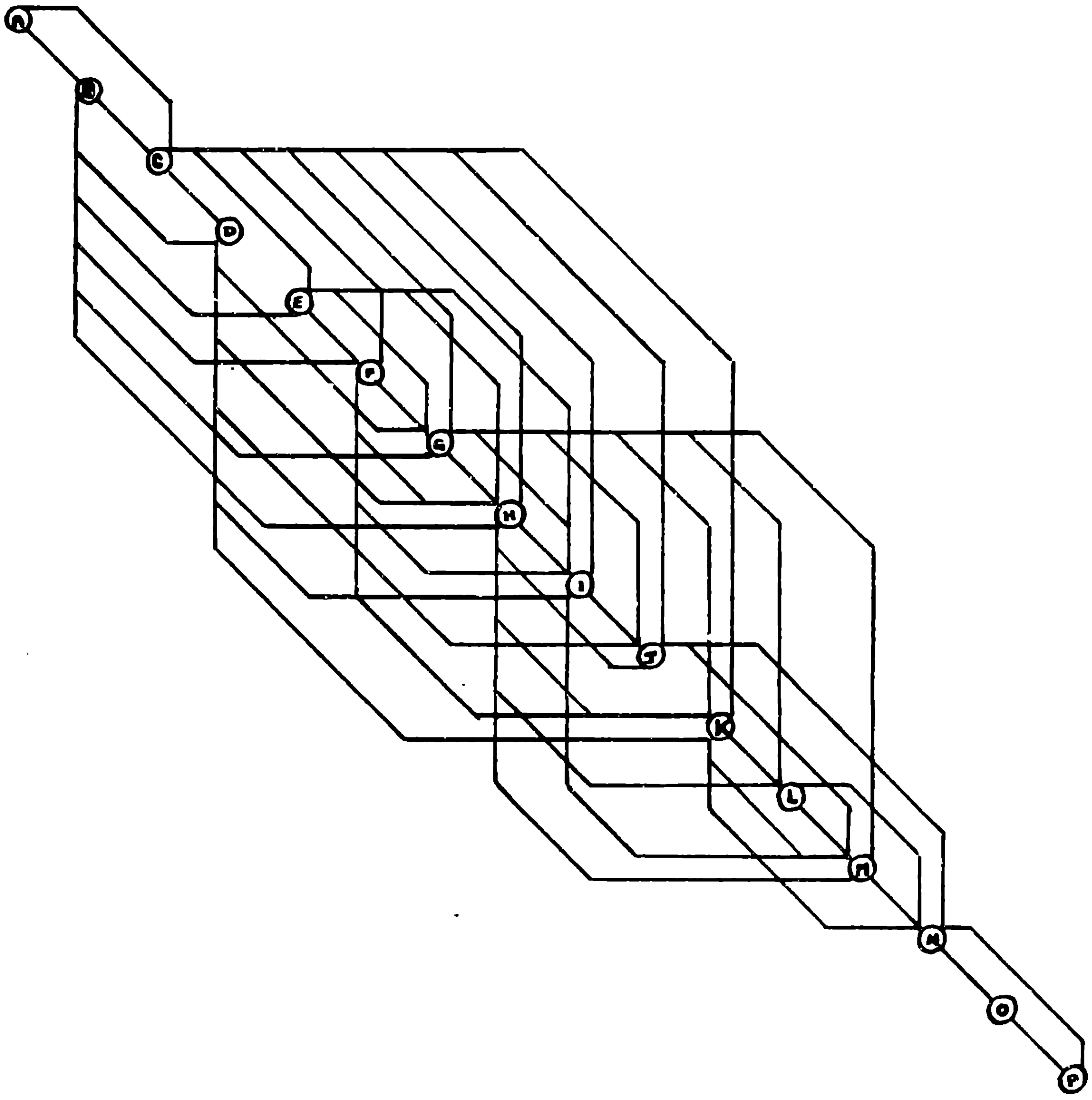
Work station cost has both a fixed and a variable part. The fixed part is the sum of the annualized fixed cost for the resource type and the annualized cost of the necessary tools. The variable cost is the product of the variable cost per hour for the given resource and the total hours the system will be in operation. For the current example, total system time is 1920 hours from before. By adding the fixed and variable costs, the total station cost is obtained. This quantity is computed for each feasible resource type and the least cost resource type for the candidate work station is selected. For work station (3,4,5) the following chart shows the total station cost computation.

	Resource 1		Resource 2	
Annualized Fixed Cost		$40000		$50000
Annualized Tool Cost		$3000		$3000
Variable Cost/hour	$4.8		$5.3	
Hours in production	1920		1920	
Total Variable Cost		$9216		$10216
Total Station Cost		$52216		$62176

For this work station resource 1 is identified as least cost.

Once the least cost resource type has been determined for each candidate work station, the least cost system of non-overlapping work stations must be found. That is, the desired system is the least cost set of work stations such that each task is assigned to exactly one work station. One such system for our example problem which is not necessarily least cost is: {(1,2), (3,4,5), (6,7), (8,9), (10), (11,12)}. To find the least cost system we consider the network diagram in Figure 3.5.

NETWORK REPRESENTATION OF FEASIBLE WORK STATIONS



Each node in the diagram is a cut from the original task set diagram.

Each arc connects a pair of cuts and represents a candidate work station.

Figure 3.5

The nodes in this diagram represent the cuts from the original task set diagram. The arcs in this diagram are candidate work stations which are pairs of cuts. For example, Arc (E,G) is the candidate work station with tasks (3,5). Associated with each arc (work station) is a cost equal to the station cost for that resource which was selected as least cost in the previous step. Finding the least cost system is then a matter of finding the least cost path from node A to node O in the diagram. This is a Shortest Path Problem where the length of each arc is defined to be its associated cost.

Dijkstra's Shortest Path Algorithm [3] is a well known efficient way to solve this problem. The algorithm seeks to find the shortest (least cost) path from origin to destination in a connected network diagram. The algorithm proceeds by finding the shortest path from the origin to the next closest node in the network adding one node at each step until all nodes have been labeled and the shortest path from origin to destination has been found.

For the example problem there were 16 cut sets from which 64 feasible work stations were generated. The complete shortest path solution is given in Figure 3.6.

The least cost system for the example has three stations of resource type 1 and one station of resource type 2. Because resource type 1 has a lower fixed and variable cost than resource 2, it is in general the least cost choice for a work station.

LEAST COST SYSTEM

Work Station	Resource Type	Task #'s	Tool #'s	Station Cost
1	1	1, 2	100, 120	$68216
2	2	3, 4, 5 6, 7, 8	221, 231 241, 242	$75216
3	1	9, 10	150, 160	$60216
4	1	11, 12	170	$59216

TOTAL SYSTEM COST $262,824

WORK STATION SUMMARY BY PRODUCT

	Work Station	Tasks	Station Time(seconds)
<u>Product 1</u>	1	1,2	11.2
	2	3,5,6,8	12.6
	3	9,10	13.2
	4	12	5.4
<u>Product 2</u>	1	1,2	11.2
	2	4,5,6,7	13.0
	3	10	7.2
	4	11,12	10.8

Maximum Station Time for Product 1 = 13.2 seconds

Maximum Station TIme for Product 2 = 13.0 seconds

Figure 3.6

Resource 2 is the optimal resource for work station 2 because resource 1 requires more time to do the assembly operations than resource 2 and cannot complete the tasks in work station 2 in the allotted cycle time. Six tasks have been assigned to work station 2, but examining the system by product shows that a maximum of four tasks will be done at station 2 at one time, four by product 1 and four by product 2. The available cycle time for the system is 14 seconds. The maximum station time of the least cost system is 13.2 seconds for product 1 and 13.0 seconds for product 2. These maximum station times are the effective cycle times of the system. That is, they determine the actual frequency with which products will be assembled if this system is employed.

This solution is the least cost assembly system design for the input data. In the next chapter the sensitivity of the solution to changes in the input parameters is explored.

CHAPTER 4 - IMPLEMENTATION OF THE PROGRAM

Designing an effective assembly system is contingent upon careful parameter selection for the model to be solved. In this chapter we present a discussion of parameter choices and the sensitivity of the model to variations in those parameters. Successfully implemented, the model can be a useful tool for testing alternative system designs based on parameter choices. In the implementation of any optimization method, one must be concerned about the computational and storage requirements especially as they grow in relation to the number of tasks, products and resources. In this chapter the theoretical bounds on problem size and the actual results which can be expected when implementing the model are discussed.

4.1 THE PARAMETERS OF THE MODEL

The principal parameters of the model can be grouped into two categories: 1) those parameters which affect available cycle time, 2) and the cost parameters. The cycle time for each product governs the feasibility of candidate work stations. Recall that cycle time, the frequency with which products come off the line, is computed:

$$\begin{array}{c}\text{Cycle Time for}\\ \text{product P}\\ \text{(secs/unit)}\end{array} = \frac{\text{Total Available Time for product P(secs)}}{\text{Annual Volume for product P(units)}}$$

If the accumulated station time for a particular resource is less than the cycle time for all products, then the candidate is feasible. In general, a larger cycle time allows for more flexibility in the system design because fewer candidate work stations are eliminated as infeasible. A small cycle time may restrict the number of design possiblities so that a higher system cost could result than would have if a larger cycle time was available.

There are three factors which affect the cycle time for a product: 1) the total available time for the system, 2) the fraction of production time spent on the product, and 3) the annual volume for the product.

The Total Available Time for product P is computed as:

(Days/Year)x(Shifts/Day)x(Hours/Shift)x(3600 Secs/Hour)
x (Fraction of total time spent on product P).

System-wide adjustments to cycle time can be made by altering the total production time for the system. For example, increasing the number of shifts per day increases the total available time and, therefore, increases the cycle time proportionally for all products.

The fraction of time spent on product P is a measurement of how production time is divided among the products. It can be the desired proportion of time given to each product, or an approximation to the actual fraction of time allotted to the assembly of each product, or the fraction of production volume which the product represents. Sometimes the desired proportion of time to be spent on each product is a known quantity, perhaps set by management. In this case, the fraction for each product can be input as a fixed quantity. Typically, these quantities are not known with certainty before the system is designed. In this case the fractions are computed by making an approximation, F(P), to the actual fraction of time that will be spent in the assembly of the product as follows:

$$\text{(Time to assemble a unit of product p)} = \sum_{\substack{\text{\{tasks for}\\ \text{product p\}}}} \text{(avr time to complete complete a task)}$$

$$F(P) = \frac{\text{(Time to assemble a unit of p)(annual volume of p)}}{\sum_{P} \text{(Time to assemble a unit of p)(annual volume of p)}}$$

Varying the fraction values is a means of testing the sensitivity of the solution to fluctuations in the distribution of the total cycle time among the products.

Calculating the fraction of time spent on product P simply as the fraction of total annual volume that product P represents forces the cycle time to be equal for all products. Although this is sometimes a desired result, it can have the effect of constraining the system to less than full capacity. Redistributing the cycle time so that products which actually require more production time are allotted a larger relative cycle time effectively increases the available capacity of the system and can result in a lower system cost.

The final factor in the computation of cycle time is the annual volume for each product. The greater the production demand which must be satisfied, the lower the cycle times will be and, hence, the more constrained the system will be.

At some production level the system becomes infeasible because the largest task time exceeds the available cycle time of the system. To accomodate this production level the available time can be increased, parallel stations can be added, or duplicate systems can be built for each product. By varying the annual volume for each product the sensitivity of the optimal solution to different production levels can be explored.

The second category of parameters of our model contains the cost parameters. Unlike the cycle time parameters, the cost parameters do not affect the feasibility of the work stations in a system. Once feasibility has been determined, the cost parameters govern the selection of the least cost resource for each work station and the overall design of the minimum cost system.

Costs are separated into two types: fixed and variable. The fixed costs are the resource and tool prices which are annualized by multiplying by the annualized cost factor to reflect that portion of the total cost which will be charged in a given year. The annualized cost factor is a system wide parameter which is set according to management cost accounting methods. The fixed costs are multiplied by an installed cost factor for each resource which reflects the expected total installed cost of a resource or tool as a multiple of the fixed hardware cost. That is,

Installed Cost Factor = Total Cost/ Hardware Cost.

The implication is that only a fraction of the total cost of a given resource or tool is reflected in its price. The annualized fixed cost of a resource is then given by:

(Resource Price)x(Annualized Cost Factor)
x (Installed Cost Factor)

and similarly the annualized tool cost is given as:

(Tool Price)x(Annualized Cost Factor)
x (Installed Cost Factor)

There are two types of variable costs: Labor costs and operational costs. The adjusted labor cost per hour for a resource is computed as:

$$\frac{\text{Loaded Labor Rate (\$/hour)}}{\text{Maximum \# Stations per Worker}}$$

The loaded labor rate is the total cost including benefits for each hour of labor input. The maximum number of stations per worker is the maximum number of stations to which one worker can be assigned. Only that fraction of a worker which will actually be assigned to a resource is billed as a variable cost.

In addition to the labor costs there are operational variable costs associated with using a resource. Total variable cost is the sum of the operation rate and the adjusted labor rate which is billed for each hour the system is in operation. The time the system is in operation is the total available time for production which is computed as:

Total Available Time = (Days/Year) (Shifts/Day) (Hours/Shift)

Then for each work station:

Total Variable Cost = (Total Available Time in Hours) x (Variable Cost/Hour)

The actual production time for the system is based on the maximum station time for each product and may be less than the total available production time. Actual production time is computed as:

$$\sum_{P} (\text{Maximum station time for P}) \times (\text{Annual Volume P})$$

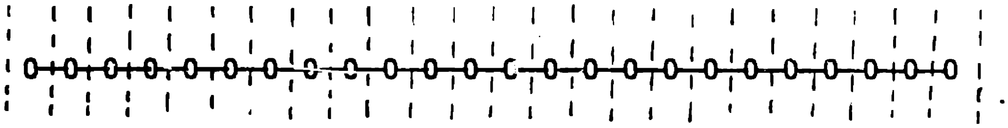
For each solution both the system cost based on the total available time and an updated system cost based on the actual production time that is needed by the system are computed.

4.2 THE PROBLEM SIZE

The preceding discussion suggests the use of the model iteratively to test the sensitivity to variations to the input parameters. As a practical issue, it is necessary to know the bounds on problem size if this type of iterative testing is to be done. Below, both the theoretical maximum on problem size and the typical results the user can expect are examined.

The theoretical maximum on problem size for the solution is very large. Any solution that requires the enumeration of feasible subsets of a given task set has as its worst case complete enumeration of all subsets assuming no structure to the problem. Let t be the number of tasks, p be the number of products and r be the number of resources for a given problem. If we assume no ordering to the tasks, there are, in the worst case, $(2^t - 1)$ feasible subsets of the t tasks. The feasibility of each subset (work station) must be checked for all resource and product combinations. Therefore, the absolute maximum number of feasibility checks is $(2^t - 1)(p)(r)$. Fortunately there is plenty of structure to a realistic task set, and, typically, the actual problem size is only a small fraction of this maximum.

For purposes of example, consider a 24 task problem. The most structured problem for any number of tasks is the single product case, as depicted below:

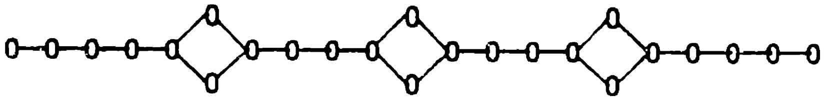


For this example there are 25 possible cut sets corresponding to the cuts represented as dashed lines in the above diagram. If k is the number of cut sets in a task diagram, the maximum number of candidate work stations for that task set is:

$$(k - 1) + (k - 2) + . . . + 2 + 1 = \frac{(k)(k - 1)}{2}$$

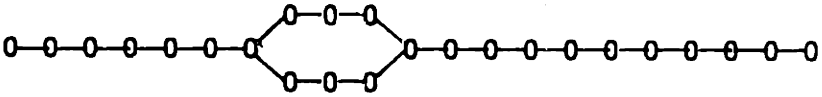
Then, for the 24 task example we have a maximum of (25 x 24)/2 = 300 candidate work stations for the single product case.

The size and complexity of the problem increases as the number of products is increased. For two products, consider the 24 task examples below.

Example 1 

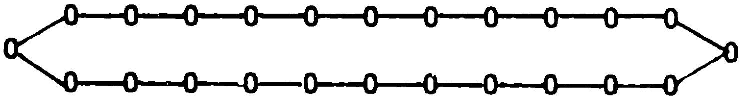
Number of cuts = 32   Number of work stations <= 496

In this example, the two products diverge at three separate points. Each point of divergence adds 1 cut to the total number of cuts.

Example 2 

Number of cuts = 34   Number of work stations <= 561

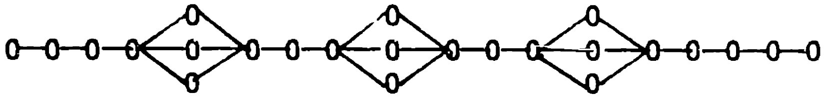
In this example, the two products diverge at one point for three tasks. This three task divergence adds 9 cuts to the total number of cuts.

Example 3 

Number of cuts = 146   Number of work stations <= 10,585

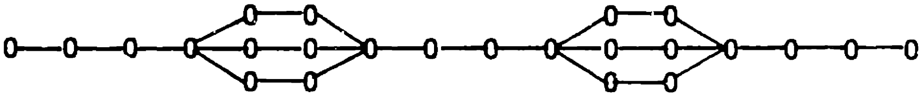
This is a worst case example for a two product design problem. In reality, this worst case would never occur because there would be no incentive for trying to assemble two such diverse products on the same assembly line.

Similarly, consider the following three product cases for the 24 task example.

Example 4 

Number of cuts = 37 Number of work stations <= 666

In this example, the three products diverge at three points. Each point of divergence adds 4 cuts to the total number of cuts.

Example 5 

Number of cuts = 65 Number of work stations <= 2080

In this example, the three products diverge at two separate points for two tasks at each point of divergence. Each point of divergence adds 20 cuts to the total number of cuts.

As the number of tasks at each point of divergence increases, the size of the problem grows rapidly (that is, the number of design possibilities for the configuration of the assembly system multiplies). A good approximation to the number of cuts that can be expected for a problem with p products is:

$$\#\text{ CUTS} = [\ \#\text{TIMES}\ ]\ [\ \#\text{DIFF} + 1\ ]^{p} + [\ \#\text{SAME}\ ]$$

where #TIMES = the number of points of divergence
#DIFF = the number of task differences at each point of divergence
#SAME = the number of tasks that are the same for all products
p = the number of products

assuming that each point of divergence is the same.

Even the worst cases for the two and three product cases are no where near the theoretical maximum of $2^{24} - 1$. The problem, for the most part, is somewhat self-limiting. It is not likely that problems with a large number of task differences would be selected for assembly on the same assembly line. For a small number of products (one, two, or three), assuming that the products are reasonably similar (over half the tasks are the same), the problem will not become prohibitively large.

As the approximation formula above shows, when the assembly of more than three products is considered, the problem size can grow very quickly. It would be inaccurate to imply that the problem size is not computationally large in spite of all of its self limiting properties. Even so, the computational results so far have been encouraging. An application of the system with computational results is presented in the next chapter.

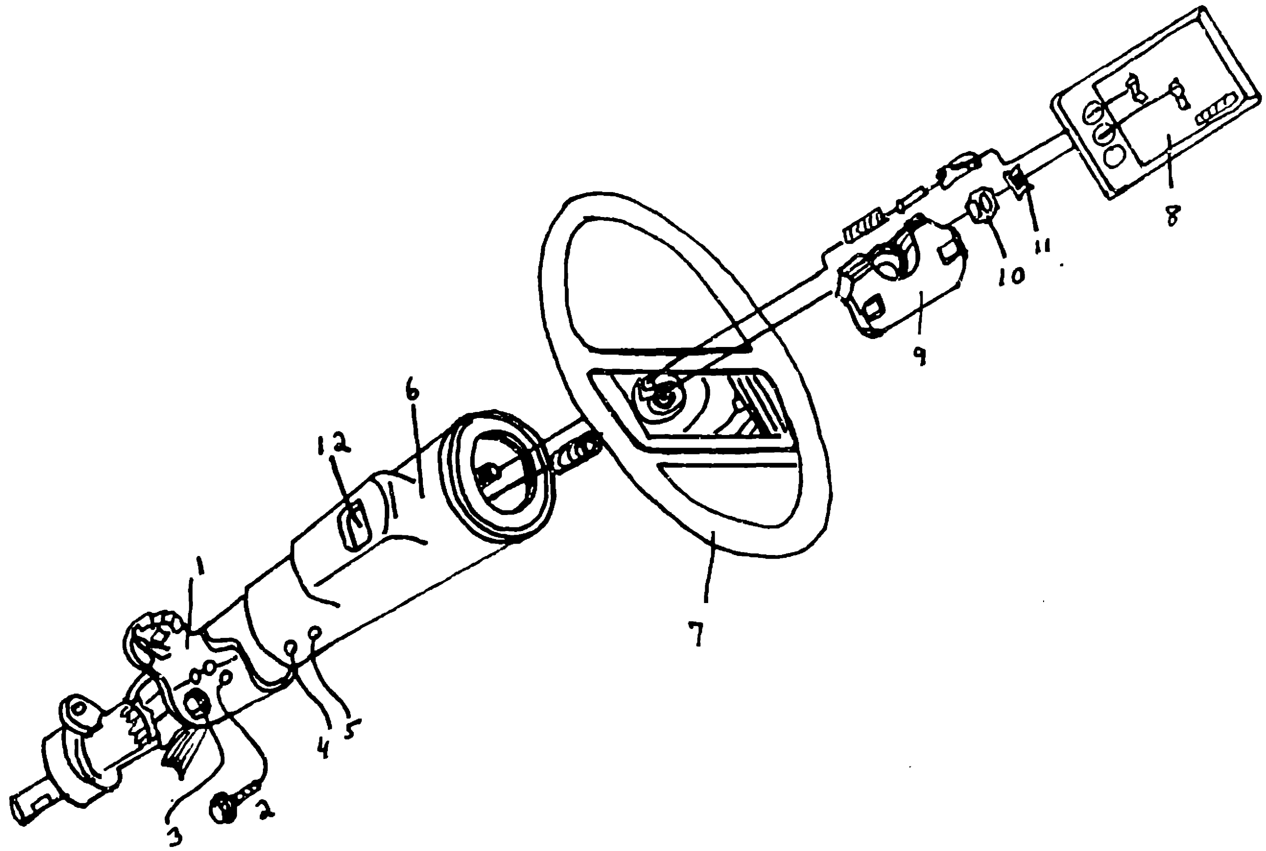
CHAPTER 5 - APPLICATION OF THE SYSTEM TO THE AUTOMOTIVE INDUSTRY

As an application of the multiproduct equipment selection system, consider a design problem from the automotive industry. A manufacturer wishes to assemble three models of a steering column as a subassembly for an automobile. The three models have most of their parts in common, but options include a turn/cruise lever rather than the basic turn lever, a hazard switch, and a tilt option on the steering wheel. The targeted annual volume of product is 250,000 units distributed in a .63/.29/.08 ratio between the models. This is a hypothetical example based on the assembly specifications of a real product. The multi-model dimension of the problem was fabricated to demonstrate the MESP system. In this chapter, the basic solution to the design problem is presented in detail with a discussion of the sensitivity of the solution to variations in annual production volume.

5.1 STEERING COLUMN SUBASSEMBLY DATA

Figures 5.1, 5.2, and 5.3 present the steering column subassembly data. Figure 5.1 is a drawing of the steering column to be assembled. The parts to be assembled are identified by number. Figure 5.2 describes the resources available to complete the tasks and gives the operation times and tool data for each of the 28 assembly tasks. In Figure 5.2 a blank resource/task box implies that the selected resource cannot perform the task. For this application the resource options are manual, fixed machines (fixed

AUTOMOBILE STEERING COLUMN



1 BRACKET
2 BOLT #1
3 BOLT #2
4 BOLT #3
5 BOLT #4
6 STEERING COLUMN
7 STEERING WHEEL
8 HORN PAD
9 DAMPNER
10 STEERING WHEEL NUT
11 RETAINER
12 TURN/CRUISE LEVER

FIGURE 5.1

STEERING COLUMN SUBASSEMBLY DATA

WORKING DAYS PER YEAR: 235
SHIFTS AVAILABLE: 2
ANNUALIZED COST FACTOR: .35
LABOR RATE($/HOUR): 21

STATION-TO-STATION MOVE TIME(SECS): 4
ANNUAL VOLUME(UNITS): 250,000

KEY
C =FIXED RESOURCE COST($)
I =INSTALLED COST FACTOR
U =UPTIME EXPECTED(%)
V =VARIABLE OPERATING COST($/HR)
T =TOOL CHANGE TIME(SECS)
M =MAXIMUM STATIONS/WORKER

FOR EACH TASK

OPERATION TIME(SEC)	TOOL NUMBER
TOOL COST($)	

RESOURCE / TASK	MA1		MA2		FXD		PT1		PT2	
	C:	500	C:	1000	C:	15000	C:	35000	C:	70000
	I:	1.5	I:	1.5	I:	1.5	I:	1.5	I:	1.5
	U:	99.2	U:	99.2	U:	98	U:	99	U:	99
	V:	.5	V:	1.0	V:	1.5	V:	1.5	V:	3.0
	T:	2.5	T:	1.25	T:	2.0	T:	2.0	T:	1.0
	M:	.90	M:	.45	M:	5	M:	.90	M:	.45
1 ALIGN STEERING COLUMN	23	100	11	100						
	—		—							
2 BRACKET & BOLT TO COLUMN	8	100	4	100						
	—		—							
3 FINGER START BOLT #2	3	100	1.5	100						
	—		—							
4 FINGER START BOLT #3	3	100	1.5	100						
	—		—							
5 FINGER START BOLT #4	3	100	1.5	100						
	—		—							
6 PAINT STEERING COLUMN							43	401	21	401
							15000		30000	
7 ALIGN STEERING WHEEL	17	100	8.5	100						
	—		—							
8 ALIGN HORN PAD	9	100	4.5	100						
	—		—							
9 STEERING WHEEL TO COLUMN	7	100	3.5	100	6	201				
	—		—		17000					
10 PLACE DAMPNER TO COLUMN	7	102	3.5	102	6	202				
	19000		38000		24000					

FIGURE 5.2

SHEET 1/2

STEERING COLUMN SUBASSEMBLY DATA

TASK RESOURCE	MA1		MA2		FXD		PT1		PT2	
11 DRIVE STEERING WHEEL NUT	6	103	3	103	3	203				
	19000		18000		24000					
12 INSPECT NUT TORQUE	2	103	1	103	2	203				
	19000		18000		24000					
13 INSTALL NUT RETAINER	2	104	1	104	1	204				
	2500		5000		11000					
14 VISUAL INSPECT RETAINER	2	100	1	100						
	—		—							
15 INSTALL TRN/CRUISE LEVER	14	105	7	105						
	1000		2000							
16 INSTALL TURN LEVER	8	100	4	100	4	205				
	—		—		14000					
17 HORN PAD TO WHEEL	8	100	4	100	5	206				
	—		—		17000					
18 INSTALL TILT LEVER	7	100	3.5	100	3	207				
	—		—		13000					
19 SECURE BRACKET BOLT #1	3	108	1.5	108	1	208				
	13000		26000		28000					
20 SECURE BRACKET BOLT #2	3	108	1.5	108	1	208				
	13000		26000		28000					
21 SECURE BRACKET BOLT #3	3	108	1.5	108	1	208				
	13000		26000		28000					
22 SECURE BRACKET BOLT #4	3	108	1.5	108	1	208				
	13000		26000		28000					
23 TEST BRACKET SECURENESS	2	100	1	100						
	—		—							
24 TEST HORN PAD SECURENESS	2	100	1	100						
	—		—							
25 INSTALL HAZARD SWITCH	9	109	4.5	109						
	2500		5000							
26 TEST TURN/CRUISE LEVER	21	110	11	110						
	30000		60000							
27 TEST HAZARD SWITCH	7	110	3.5	110						
	30000		60000							
28 ELECTRICAL TEST HORN PAD	12	110	6	110						
	30000		60000							

FIGURE 5.2 SHEET 2/2

automation), and automated paint machines. A programmable machine was considered in the original problem formulation, but as it was never selected in the optimal solution, it was omitted from this presentation. Resources MA1 and MA2 are manual work stations. MA1 is a single manual station (one worker), and MA2 is a double manual station with two workers working in parallel on the same set of tasks. MA2 can do the same amount of work as MA1 in half the time for twice the fixed and variable cost. Resources PT1 and PT2 are paint stations used only to complete task #6 (paint steering column). PT1 is a single paint machine which requires one operator. PT2 is two (parallel) paint machines each with its own operator. Similar to MA2, resource PA2 can do the same work as resource PA1 in half the time for twice the cost. Resource FXD is a fixed automation machine.

Figure 5.3 shows the model differences by giving the specific task requirements of each model and by graphically depicting the network of tasks to be completed for all models. As shown, model 1 has all the options including a dampner, a turn/cruise lever, a hazard switch, and a tilt lever, and accounts for 63% of the annual demand. Model 2 is the basic model with no options, and accounts for 29% of demand. Model 3 has a dampner and a turn/cruise lever, but no tilt lever or hazard switch, and accounts for 8% of demand. The models are, for the most part quite similar, requiring close to the same amount of time to assemble a unit of any one of the three models.

MODEL DIFFERENTIATION

	TASKS FOR ASSEMBLY													
MODEL #1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	15		17	18	19	20	21	22	23	24	25	26	27	28
MODEL #2	1	2	3	4	5	6	7	8	9		11	12	13	14
		16	17		19	20	21	22	23	24				28
MODEL #3	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	15		17		19	20	21	22	23	24		26		28

NETWORK DIAGRAM OF TASK DIFFERENCES

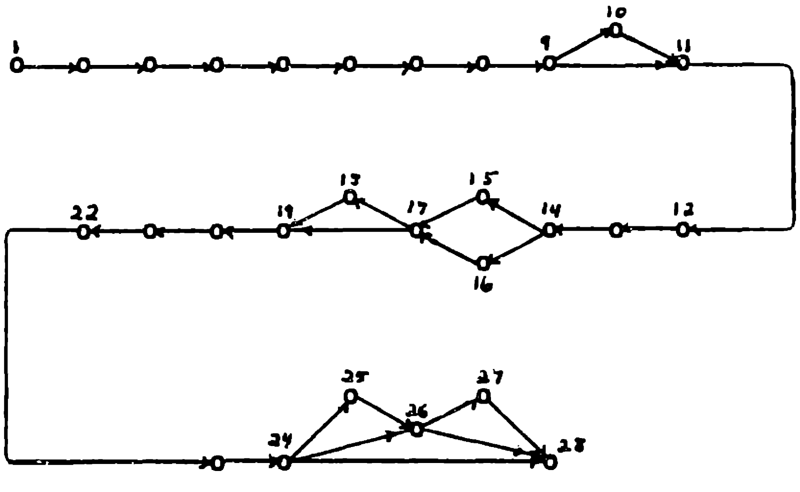


FIGURE 5.3

5.2 SOLUTION OF THE BASIC PROBLEM

The solution procedure described in chapter four was used to find an optimal system configuration for the basic 250,000 unit problem. Values for the fraction of total time to be spent on each model (needed to compute cycle times) were selected as the portion of total volume which the model represents. Selecting the fractions in this manner forces the cycle times to be equal for all models, which is the best allocation of the available time when there are not substantial differences between the models. For this application, the fractions of total time to be spent on models one, two, and three, respectively, are .63, .29, .08.

Figure 5.4 is the report generated by the equipment selection system giving the complete solution to the basic 250,000 unit problem. The cycle time available for each model (product) is about 54 seconds, which determines the maximum amount of time that can be used to assemble a unit of product if the goal of 250,000 units is to be produced in the available time. The optimal system configuration has 6 work stations, five manual stations and one paint station. The General Report gives the tasks and tool requirements for each of the stations and gives the cost of each station which, when summed together, give the system cost. The program takes as input a target cycle time that is sufficient to meet annual demand. Variable costs are computed assuming that the system will have this cycle time. However, the algorithm finds a system whose actual cycle time, equal to maximum

```
                         GENERAL REPORT

ANNUAL VOLUME:  250000            NUMBER OF SHIFTS/DAY:  2

          CYCLETIME PRODUCT( 1 ):  54.144  SECONDS

          CYCLETIME PRODUCT( 2 ):  54.144  SECONDS

          CYCLETIME PRODUCT( 3 ):  54.144  SECONDS

                     APPARENT   ADJUSTED
STATION   RESOURCE     COST       COST        TASKS          TOOLS

   1         MA1     $ 92388.5  $ 82096.39   1 2 3 4 5        100

   2         PT1     $ 122730   $ 112189.9   6                401

   3         MA1     $ 102591.5 $ 92299.39   7 8 9 10         100 102

   4         MA1     $ 104471   $ 94178.89   11 12 13         103 104
                                             14 15 16         100 105
                                             17

   5         MA1     $ 100712   $ 90419.89   18 19 20         100 108
                                             21 22 23         109
                                             24 25

   6         MA1     $ 108498.5 $ 98206.39   26 27 28         110

TOTAL APPARENT COST          $631392

ADJUSTED SYSTEM COST         $569391

TOTAL TIME FOR REPORT RUN ==> 00:02:55
```

Figure 5.4   Sheet 1/2

FOR PRODUCT 1

STATION	RESOURCE	ACCUMULATED STATION TIME	TASKS
1	MA1	40.00 seconds	1 2 3 4 5
2	PT1	43.00 seconds	6
3	MA1	42.50 seconds	7 8 9 10
4	MA1	44.00 seconds	11 12 13 14 15 17
5	MA1	39.50 seconds	18 19 20 21 22 23 24 25
6	MA1	40.00 seconds	26 27 28

FOR PRODUCT 2

STATION	RESOURCE	ACCUMULATED STATION TIME	TASKS
1	MA1	40.00 seconds	1 2 3 4 5
2	PT1	43.00 seconds	6
3	MA1	33.00 seconds	7 8 9
4	MA1	33.00 seconds	11 12 13 14 16 17
5	MA1	18.50 seconds	19 20 21 22 23 24
6	MA1	12.00 seconds	28

FOR PRODUCT 3

STATION	RESOURCE	ACCUMULATED STATION TIME	TASKS
1	MA1	40.00 seconds	1 2 3 4 5
2	PT1	43.00 seconds	6
3	MA1	42.50 seconds	7 8 9 10
4	MA1	44.00 seconds	11 12 13 14 15 17
5	MA1	18.50 seconds	19 20 21 22 23 24
6	MA1	33.00 seconds	26 28

Figure 5.4 Sheet 2/2

station time, is less than or equal to cycle time. As a consequence, the system may not have to be run for the entire available time to meet demand. Then, if the variable cost is truly variable, the actual variable cost will be less than predicted since it depends on actual rather than the target cycle time . The report gives both the apparent cost and an adjusted cost for each station. The apparent cost is computed using a variable cost based on total available time (initial cycle time). The adjusted cost recomputes variable cost using the maximum station time (actual cycle time) once the system has been configured. For this application the adjusted costs will be compared. The initial cycle time for each product is 54 seconds. The actual cycle times are 44, 43, and 44 seconds for models one, two and three respectively. The least cost system design to assemble 250,000 steering columns has a total adjusted cost of $569,391. Total run time for the computer program was just under 3 minutes on an IBM PC XT.

In the Individual Products Report, the work station configurations are given on a product by product basis. The tasks and the accumulated station times are given for each work station showing the distribution of the work among the stations.

5.3 SENSITIVITY ANALYSIS

The solution given in Figure 5.4 is optimal assuming a production level of 250,000 units and 2 complete shifts. Often, it is difficult or impossible to forecast the annual production volume with certainty. In this section the sensitivity of the solution to different production levels is tested. For all production levels, two full shifts of labor are used. Clearly, with large fluctuations in volume, the amount of available production time might be altered, but for this analysis the available time is assumed to remain constant. Figure 5.5 summarizes the system results for a range of volumes from 150,000 to 500,000 units. As might be expected, fixed machine stations become more attractive as the production level increases. The large fixed cost investment associated with fixed machines is justified by their speed of assembly which lowers variable costs. It is interesting, however, that the optimal solution at 150,000 units using two full shifts also selects a fixed machine for task #17.

In conclusion, at 250,000 units of production, the least cost solution is all manual except for the one paint station for task #6. If the volume of production is expected to grow in the future or there is a good chance that the annual volume will be higher than the predicted 250,000 units, then an investment in the fixed equipment is justified for tasks #10-12 and should be considered for tasks #17-22. In the next chapter, the possibility of designing a computer algorithm that will accept dynamic demand levels rather than a single fixed volume will be discussed. Such a system would have the advantage of performing a sensitivity analysis as a part of the system which finds the optimal solution.

STEERING COLUMN SUBASSEMBLY

SUMMARY OF RESULTS

FOR EACH STATION

RESOURCE

TASKS

ANNUAL VOLUME	SYSTEM COST($)	STATIONS									
		1	2	3	4	5	6	7	8	9	10
150000	441919	MA1 1-5	PT1 6	MA1 7-16	FXD 17	MA1 18-28					
200000	528309	MA1 1-5	PT1 6	MA1 7-14 16	MA1 15 17-24	MA1 25-28					
250000	569391	MA1 1-5	PT1 6	MA1 7-10	MA1 11-17	MA1 18-25	MA1 26-28				
300000	783390	MA1 1-5	PT2 6	MA1 7-9	FXD 10-12	MA1 13-18	MA1 19-25	MA1 26-28			
400000	932431	MA1 1	MA1 2-5	PT2 6	MA1 7-8	FXD 9-13	MA1 14-17	FXD 18-22	MA2 23-28		
500000	1111290	MA2 1-5	PT2 6	MA1 7	MA1 8-9	FXD 10-13	MA1 14-16	FXD 17-22	MA1 23-25	MA1 26	MA1 27-28

FIGURE 5.5

CHAPTER 6 - EXTENSIONS AND RESEARCH POSSIBILITIES

Extensive testing of the multiproduct equipment selection problem (MESP) system has uncovered several possibilities for extensions and enhancements to the MESP program. These topics are presented here as starting points for future research into multiproduct assembly system design.

6.1 ALLOCATION OF AVAILABLE TIME BETWEEN PRODUCTS

When using the MESP system, the maximum cycle time for each product must be determined before the optimal system can be found. In the single product case, the cycle time is computed directly as total available time for the system divided by the expected product volume. In the multiproduct case, total available time must be allocated between the products in order to compute a separate cycle time for each product. As discussed in chapter 4, an estimate for the fraction of total time spent on each product can be made based on an average across all resources of the time needed to assemble a unit of product. Alternatively, the fractions can be selected so that they force the cycle times for all products to be equal. At the present time, the only way to find an appropriate choice for the fractions for a given application is by trial and error with no guarantee that the best choice of fractions has been found. Because efficient allocation of the available time between the products is essential to finding the best configuration of the system, further research into the best way to allocate available time is needed.

6.2 SYSTEM RELIABILITY

System reliability is the portion of available time that a system is expected to be functioning properly. The current formulation of the MESP accounts for reliability on an individual resource basis by assigning a Percent Uptime Expected to each resource. This Percent is then used in the computation of accumulated station time. Considering the reliability of each resource separately assumes that a failure of one resource will not reduce the availability of any other resource. Although this is sometimes true, it is not always the case. If a system can continue to function when one station fails (due to low utilization and/or sufficient buffer stock), then accounting for reliability on an individual resource basis is appropriate. On the other hand, if the failure of one resource shuts down the entire system, then a system-wide reliability factor would be more appropriate. In this case the system-wide reliability is equal to the product of the reliabilities of the resources that make up the assembly system. The goal of the MESP, however, is to select the most economical resources. Before the system has been designed, and the resources selected, the system reliability cannot be determined. This leaves the unresolved issue of how best to account for system reliability in assembly system design.

6.3 CHANGEOVER TIME BETWEEN PRODUCTS

Each time an assembly system must be changed from the assembly of one product to another, time is needed to reconfigure the system for the assembly of the new product. Each resource may have a different changeover time. Since the MESP program assumes that changeover for different resources can occur simultaneously, the changeover time for the system is the largest changeover time among the resources. If a changeover time for the system is known, then it can be subtracted from the available time before the cycle time is computed. As is the case with system reliability, however, the actual changeover time cannot be determined until the resources have been selected and the system has been designed.

In the MESP program, there are two ways to account for changeover time, neither of which is considered a satisfactory solution to the problem of determining changeover time. The method that has been used most often in applying the MESP system simply requires the user to approximate the amount of time that will be spent on changeovers between products before the system is designed and to adjust total available time accordingly. The second alternative inputs for each resource a changeover time and then iterates the MESP system, once for each different value of the changeover time. At each iteration, the current value of the changeover time determines a feasible set of candidate resources for the system by selecting only those resources with a changeover time less than or equal to the system changeover time. After the optimal system has been found for each changeover time, the least cost solution can be selected. This method is time consuming computationally and, to date, has not been sufficiently tested. In general, further study of the changeover time between products is needed.

6.4 PARALLEL STATIONS

In the steering column application of chapter 5, MA2 and PT2 were explicitly included in the list of available resources. MA2 was a double manual station which had two people working in parallel on the same group of tasks. PT2 was a paint station with parallel paint machines. Double (or parallel) stations were included in the steering column application because at high demand levels the single stations were unable to complete all the assembly tasks in the available cycle time. Without the parallel stations, the problem would have been infeasible for volumes above 300,000 units.

For the steering column application, the parallel stations were manually added by observing that the system was infeasible, finding the point of infeasibility, and adding a double resource to the list of candidate resources. Ideally, the manual addition of parallel stations could be replaced by a routine that would be able to detect problem infeasibility and add parallel resources when necessary as a part of the solution to the MESP.

6.5 DYNAMIC DEMANDS

The sensitivity analysis performed on the steering column application demonstrated considerable variation in the optimal solution with changes in demand level. The current formulation of the MESP can accommodate only a single forecasted demand volume. A suggested enhancement to the MESP system would allow for the input of a trajectory of demands over a given time frame rather than a single volume. With this enhancement, the MESP system would specify a dynamic solution in which the assembly system "grows" over time with the demand trajectory.

6.6 FUTURE DIRECTIONS FOR RESEARCH RELATED TO THE MESP

As stated earlier, the MESP is part of ongoing research studying assembly system design at the Charles Stark Draper Laboratory (CSDL). The current direction of research related to the MESP at CSDL includes both enhancements to the current system and incorporation with other systems at CSDL. At the present time, an external routine is being added to the system which will automatically add a parallel resource if the problem is infeasible. Following that addition, the possibility of incorporating dynamic demands will be explored.

Simultaneously, efforts are being made to incorporate the MESP system into the system developed at CSDL which heuristically solves the multiproduct equipment selection problem. The heuristic system is much further developed than the MESP system in terms of data input, output reports, and cost evaluation. Once combined, the MESP system could be used as one subroutine of the heuristic system to give the user the option of exploring both heuristic and optimal solutions.

Finally, CSDL is developing a program which will automatically generate assembly sequences using their liaison sequence analysis method. When that program is complete, steps will be taken to adapt the MESP system to automatically evaluate assembly sequences. The two routines together will form a single system that can generate all possible assembly sequences for a product, determine the optimal assembly system for each sequence, and select the least cost sequence and assembly system design for the product.

APPENDIX

This appendix contains the BASIC code for the MESP program. The code was written for the IBM PC version of BASICA and was compiled using the IBM PC BASIC compiler 1.0.

```
30 ' ***            EQUIPMENT SELECTION AND TASK ASSIGNMENT      ***
40 ' ***            FOR MULTIPRODUCT ASSEMBLY SYSTEM DESIGN      ***

80 ' **************************************************************
90  SCREEN 1:COLOR 1,15:CIRCLE (40,30),10,4,,,1:PAINT (40,30),2,4
100 LINE (30,30)-(30,60):LINE (50,30)-(50,60),4:LINE (30,60)-(50,60),4
110 PAINT (40,50),2,4:LINE (35,60)-(35,70):LINE (45,60)-(45,70)
120 CIRCLE (45,70),2:CIRCLE (35,70),2:CIRCLE (44,26),1,4:CIRCLE (36,26),1,4
130 LINE (37,34)-(43,34):LINE (36,33)-(37,34):LINE (43,34)-(44,33)
140 LINE (45,50)-(83,30):LINE (45,53)-(80,33):LINE (83,30)-(93,50)
150 LINE (80,33)-(90,50):LOCATE 3,16:PRINT"SELECTEQUIP":LOCATE 4,16
160 LINE (120,30)-(208,30),2:LINE (120,28)-(208,28),2:LOCATE 8,10
170 PRINT " ";CHR$(219);CHR$(219);CHR$(219);CHR$(219):LOCATE 9,10
180 PRINT " *  *":LOCATE 12,12:PRINT"EQUIPMENT SELECTION"
190 LOCATE 14,6:PRINT"AND WORKSTATION IDENTIFICATION":LOCATE 16,12
200 PRINT"FOR THE MULTIPRODUCT":LOCATE 18,10:PRINT"ASSEMBLY LINE BALANCING"
210 LOCATE 20,18:PRINT"PROBLEM":LOCATE 22,18:LINE (134,165)-(190,165),2
220 LOCATE 24,8:PRINT"  HIT ANY KEY TO CONTINUE";
230 Y$=INKEY$:IF Y$="" THEN 230 ELSE SCREEN 0:WIDTH 80:CLS:COLOR 3
240 ' ** INITIALIZATION AND GENERAL DATA SET ENTRY
250 DEFINT H,I,J,N:OPTION BASE 1:IFLAG3=0:IFLAG4=0:IFLAG2=0:SET=0
260 INPUT "Enter General Data set name: ";GDSNAME$
270 OPEN GDSNAME$ AS #1 LEN=36
280 FIELD #1, 2 AS N1$, 2 AS N2$, 2 AS N3$, 4 AS M1$, 4 AS M2$,
    4 AS M3$, 4 AS R$,  6 AS F1$, 6 AS F2$, 2 AS F3$
290 CLS:FOR I = 1 TO 7:PRINT:NEXT
300 PRINT TAB(25) "Would you like to:"
310 PRINT TAB(25) "   1. Create or edit the general data"
320 PRINT TAB(25) "   2. View the general data"
330 PRINT TAB(25) "   3. Neither 1 or 2"
340 PRINT
350 PRINT TAB(25);:INPUT "==> Enter choice";ANS$
360 CLS
370 IF ANS$="1" THEN GOTO 640
380 GET #1
390 NTASK = CVI(N1$)
400 NRES = CVI(N2$)
410 NPROD = CVI(N3$)
420 DYEAR = VAL(M1$)
430 SDAY = VAL(M2$)
440 SHIFT = VAL(M3$)
450 REL = VAL(R$)
460 ANCOSTFAC = VAL(F1$)
470 COSTLABOR = VAL(F2$)
480 TRANSTIME = VAL(F3$)
490 IF ANS$="3" THEN CLOSE:GOTO 860
500 LPRINT "Number of Products:";NPROD
510 LPRINT "Number of Resources:";NRES
520 LPRINT "Number of Tasks:";NTASK
530 LPRINT "Number of Working Days per Year:";DYEAR
540 LPRINT "Number of Shifts per Day:";SHIFT
550 LPRINT "System Reliability Factor:";REL
560 LPRINT "Annualized Cost Factor:";ANCOSTFAC
```

```
570 LPRINT "Cost of Labor per Hour:";COSTLABOR
580 LPRINT "Station-to-Station Transport Time:";TRANSTIME
590 LPRINT:LPRINT
600 PRINT:PRINT"HIT ANY KEY TO CONTINUE"
610 Y$=INKEY$:IF Y$="" THEN 610
620 CLOSE
630 GOTO 270
640 INPUT "NUMBER OF PRODUCTS: ";NPROD
650 INPUT "NUMBER OF RESOURCE TYPES: ";NRES
660 INPUT "NUMBER OF TASKS: ";NTASK
670 INPUT "# DAYS PER YEAR:";DYEAR
680 INPUT "# SHIFTS PER DAY:";SDAY
690 SHIFT = 8
700 REL = 100
710 INPUT "ANNUALIZED COST FACTOR:";ANCOSTFAC
720 INPUT "COST OF LABOR ($/HOUR):";COSTLABOR
730 INPUT "TRANSPORT TIME:";TRANSTIME
740 LSET N1$ =MKI$(NTASK)
750 LSET N2$ =MKI$(NRES)
760 LSET N3$ =MKI$(NPROD)
770 LSET M1$ =STR$(DYEAR)
780 LSET M2$ =STR$(SDAY)
790 LSET M3$ =STR$(SHIFT)
800 LSET R$ =STR$(REL)
810 LSET F1$ = STR$(ANCOSTFAC)
820 LSET F2$ = STR$(COSTLABOR)
830 LSET F3$ = STR$(TRANSTIME)
840 PUT #1
850 CLOSE:GOTO 270
860 IF SET = 1 GOTO 1290
870 DIM FIXCOST(6)
880 DIM IFLAG1(6)
890 DIM ARC(30)
900 DIM INTASK(30,30)
910 DIM MINRESOURCE(300)
920 DIM VCOSTC(6)
930 DIM VCOSTS(6)
940 DIM UPTIME(6)
950 DIM XINCOSTFAC(6)
960 DIM RCHANGETIME(6)
970 DIM TCHANGETIME(6)
980 DIM STATPERWORK(6)
990 DIM OPTIME(30,6)
1000 DIM TOOLNUM$(30,6)
1010 DIM TOOLCOST(30,6)
1020 DIM SEQ$(6)
1030 DIM CYEAR(6)
1040 DIM ANVOL(6)
1050 DIM USED(6)
1060 DIM FRACTIME(6)
1070 DIM CYCTIME(6)
1080 DIM M(50,30)
1090 DIM IDIFF(30)
1100 DIM IBLOCKCUT(50)
1110 DIM TAIL(50)
```

```
1130 DIM MCOST(50)
1140 DIM PRED(50)
1150 DIM S(50)
1160 DIM MINARC(50)
1170 DIM HEAD(400)
1180 DIM ACCUMCOST(6)
1190 DIM ACCUMTIME(6)
1200 DIM ACOST(400)
1210 DIM ISUC(100)
1220 DIM IBLOCK(30)
1230 DIM IFIRST(30)
1240 DIM TASKTIME(30)
1250 DIM RESNAME$(6)
1260 DIM TIME(6)
1270 SET = 1
1280 ' ** ENTER DATA SETS **
1290 PRINT
1300 INPUT "TASK DATA SET NAME: ";TASKDAT$
1310 INPUT "PRODUCT DATA SET NAME: ";PRODDAT$
1320 INPUT "RESOURCE DATA SET NAME: ";RESDAT$
1330 ' ** MAIN MENU **
1340 CLS:FOR I=1 TO 7:PRINT:NEXT
1350 PRINT TAB(25) "==> MAIN MENU <=="
1360 PRINT TAB(25) "1. EDIT REMAINING DATA"
1370 PRINT TAB(25) "2. RUN MAIN PROGRAM"
1380 PRINT TAB(25) "3. QUIT"
1390 PRINT
1400 PRINT TAB(25);: INPUT "==> Enter choice";CHOICE
1410 IF CHOICE <1 OR CHOICE >3
     THEN PRINT "RETRY BETWEEN 1 AND 3": GOTO 1400
1420 ON CHOICE GOSUB 1430,3520:IF CHOICE = 3 THEN END
1430 ' ** DATA FILES MENU **
1440 CLS:FOR I = 1 TO 7:PRINT:NEXT
1450 PRINT TAB(25) "==> DATA SET OPTIONS <=="
1460 PRINT TAB(25) "1. TASK DATA"
1470 PRINT TAB(25) "2. PRODUCT DATA"
1480 PRINT TAB(25) "3. RESOURCE DATA"
1490 PRINT TAB(25) "4. RETURN TO MAIN MENU"
1500 PRINT
1510 PRINT TAB(25);:INPUT"==> Enter Choice";CHOICE1
1520 IF CHOICE1 <1 OR CHOICE1 >4
     THEN PRINT "RETRY BETWEEN 1 AND 4": GOTO 1510
1530 IF CHOICE1 = 4 THEN 1340
1540 IF CHOICE1 = 1 THEN K$ = "TASK"
1550 IF CHOICE1 = 2 THEN K$ = "PRODUCT"
1560 IF CHOICE1 = 3 THEN K$ = "RESOURCE"
1570 CLS:FOR I = 1 TO 7:PRINT:NEXT
1580 PRINT TAB(15) "==> ";K$;" DATA MANIPULATION AND DISPLAY OPTIONS <=="
1590 PRINT TAB(25) "1. CREATE DATA"
1600 PRINT TAB(25) "2. DISPLAY DATA"
1610 PRINT TAB(25) "3. UPDATE DATA"
1620 PRINT TAB(25) "4. RETURN TO DATA MENU"
1630 PRINT
1640 PRINT TAB(25);:INPUT "==> Enter choice"; CHOICE2:CLS
```

```
  1650 IF CHOICE2 <1 OR CHOICE2 >4
THEN PRINT "RETRY BETWEEN 1 AND 4": GOTO 1640
  1660 IF CHOICE2 = 4 THEN CLOSE : GOTO 1440
  1670 ON CHOICE1 GOSUB 1690,2380,2850
  1680 ' ** EDIT TASK DATA **
  1690 OPEN TASKDAT$ AS #1 LEN=16
  1700 FIELD #1, 4 AS A1$,6 AS A2$,6 AS A3$
  1710 ON CHOICE2 GOSUB 1750, 1940, 2200
  1720 CLOSE
  1730 GOTO 1570
  1740 '  ** SUBROUTINE TO CREATE TASK DATA **
  1750 PRINT "Enter an operation time of `100' to indicate that the"
  1760 PRINT "resource cannot perform the particular task"
  1770 IFLAG3 = 1
  1780 FOR R= 1 TO NRES
  1790 PRINT "FOR RESOURCE ";R
  1800 FOR N = 1 TO NTASK
  1810 PRINT "    FOR TASK NUMBER ";N
  1820 CODE% = N + NTASK*(R-1)
  1830 INPUT "OPERATING TIME (SECONDS): ";OPTIME(N,R)
  1840 IF OPTIME(N,R)=100 THEN TOOLNUM$(N,R)="0":TOOLCOST(N,R)=0:GOTO 1870
  1850 INPUT "TOOL NUMBER";TOOLNUM$(N,R)
  1860 INPUT "TOOL COST";TOOLCOST(N,R)
  1870 LSET A1$ = STR$(OPTIME(N,R))
  1880 LSET A2$ = TOOLNUM$(N,R)
  1890 LSET A3$ = STR$(TOOLCOST(N,R))
  1900 PUT #1,CODE%
  1910 NEXT N
  1920 NEXT R
  1930 RETURN
  1940 ' ** SUBROUTINE TO DISPLAY TASK DATA **
  1950 '.
  1960 CLS
  1970 LPRINT "RESOURCE    TASK      OPTIME       TOOL#       TOOL COST"
  1980 ZZ = 0
  1990 FOR R =1 TO NRES
  2000 FOR N = 1 TO NTASK
  2010 CODE% = N + NTASK*(R-1)
  2020 GET #1, CODE%
  2030 LPRINT TAB(4) R;
  2040 LPRINT TAB(14) N;
  2050 LPRINT TAB(23) A1$;
  2060 LPRINT TAB(37) A2$;
  2070 LPRINT TAB(48) "$";A3$
  2080 ZZ = ZZ + 1
  2090 IF ZZ = 20 THEN PRINT:PRINT"HIT ANY KEY TO CONTINUE ";
  2100 IF ZZ = 20 THEN PRINT"OR `N' TO RETURN TO THE TASK DATA MANIPULATION MENU"
  2110 IF ZZ <> 20 THEN 2150
  2120 Y$=INKEY$:IF Y$="" THEN 2120
  2130 IF Y$="n" OR Y$="N" THEN RETURN
  2140 ZZ = 0 : CLS
  2150 NEXT N
  2160 NEXT R
  2170 PRINT:PRINT"HIT ANY KEY TO RETURN TO THE TASK DATA MANIPULATION MENU"
  2180 Y$=INKEY$:IF Y$="" THEN GOTO 2180
  2190 RETURN
```

```
2200 ' ** SUBROUTINE TO UPDATE TASK DATA **
2210 PRINT "Enter an operation time of `100' to indicate that the"
2220 PRINT "resource cannot perform the particular task"
2230 IFLAG3 = 0
2240 INPUT "WHICH TASK";N
2250 INPUT "WHICH RESOURCE";R
2260 CODE% = N + NTASK*(R-1)
2270 INPUT "OPERATING TIME: ";OPTIME(N,R)
2280 IF OPTIME(N,R)=100 THEN TOOLNUM$(N,R)="0":TOOLCOST(N,R)=0:GOTO 2310
2290 INPUT "TOOL NUMBER";TOOLNUM$(N,R)
2300 INPUT "TOOL COST";TOOLCOST(N,R)
2310 LSET A1$ = STR$(OPTIME(N,R))
2320 LSET A2$ = TOOLNUM$(N,R)
2330 LSET A3$ = STR$(TOOLCOST(N,R))
2340 PUT #1,CODE%
2350 INPUT "UPDATE ANOTHER TASK (Y/N)";ANS$
2360 IF LEFT$(ANS$,1)="Y" OR LEFT$(ANS$,1)="y" THEN GOTO 2240
2370 RETURN
2380 ' ** SUBROUTINE TO EDIT PRODUCT DATA **
2390 OPEN PRODDAT$ AS #2 LEN=44
2400 FIELD #2, 30 AS B1$, 6 AS B2$, 4 AS B3$, 4 AS B4$
2410 ON CHOICE2 GOSUB 2450,2620,2730
2420 CLOSE
2430 GOTO 1570
2440 ' ** SUBROUTINE TO CREATE PRODUCT DATA **
2450 FOR P = 1 TO NPROD
2460 CODE% = P
2470 GOSUB 2500
2480 NEXT P
2490 RETURN
2500 FOR T = 1 TO NTASK
2510   PRINT "PRODUCT: ";P; " TASK: ";T
2520   INPUT "INPUT (0/1): ";S$
2530   IF T=1 THEN SEQ$ = S$ ELSE SEQ$ = SEQ$ + S$
2540 NEXT T
2550 INPUT "ANNUAL VOLUME:";V$
2560 LSET B1$ = SEQ$
2570 LSET B2$ = V$
2580 LSET B3$ = "0"
2590 LSET B4$ = "0"
2600 PUT #2,CODE%
2610 RETURN
2620 ' ** SUBROUTINE TO DISPLAY PRODUCT DATA **
2630 CLS
2640 FOR P = 1 TO NPROD
2650 CODE% = P
2660 GET #2, CODE%
2670 LPRINT:LPRINT "PRODUCT: ";P;"   TASK SEQUENCE: ";B1$
2680 LPRINT TAB(16) "ANNUAL VOLUME: ";B2$
2690 NEXT P
2700 PRINT: PRINT"HIT ANY KEY TO CONTINUE"
2710 Y$=INKEY$:IF Y$="" THEN GOTO 2710
2720 RETURN
```

```
2730 ' ** SUBROUTINE TO UPDATE PRODUCT DATA **
2740 INPUT "WHICH PRODUCT";P
2750 CODE% = P
2760 INPUT "UPDATE TASK SEQUENCE (Y/N)";ANS$
2770 IF LEFT$(ANS$,1)="Y" OR LEFT$(ANS$,1)="y" THEN GOTO 2810
2780 GET #2, CODE%
2790 SEQ$ = B1$
2800 GOSUB 2550
2810 INPUT "UPDATE ANOTHER PRODUCT (Y/N)";ANS$
2820 IF LEFT$(ANS$,1)="Y" OR LEFT$(ANS$,1)="y" THEN GOTO 2730
2830 RETURN
2840 RETURN
2850 ' ** SUBROUTINE TO EDIT RESOURCE DATA **
2860 OPEN RESDAT$ AS #1 LEN=39
2870 FIELD #1, 3 AS N$, 6 AS R1$, 6 AS R4$, 6 AS R9$, 6 AS R3$, 6 AS R6$,
     6 AS R11$
2880 ON CHOICE2 GOSUB 2910,2980,3090
2890 CLOSE
2900 GOTO 1570
2910 ' ** SUBROUTINE TO INPUT RESOURCE DATA **
2920 IFLAG2 = 1
2930 FOR R = 1 TO NRES
2940 PRINT:PRINT "RESOURCE:";R
2950 GOSUB 3190
2960 NEXT R
2970 RETURN
2980 ' ** SUBROUTINE TO DISPLAY RESOURCE DATA **
2990 CLS
3000 IF NRES = 1 THEN R = 1 : GOTO 3030
3010 INPUT "WHICH RESOURCE";R:IF R>NRES THEN PRINT"MAX =";NRES:GOTO 3010
3020 LPRINT
3030 GOSUB 3390
3040 LPRINT
3050 INPUT "ANOTHER RESOURCE (Y/N)";ANS$
3060 PRINT
3070 IF LEFT$(ANS$,1)="Y" OR LEFT$(ANS$,1)="y" THEN GOTO 3000
3080 RETURN
3090 ' ** SUBROUTINE TO UPDATE RESOURCE DATA **
3100 INPUT "WHICH RESOURCE";R
3110 PRINT "FOR RESOURCE:";R;"INPUT"
3120 GOSUB 3190
3130 PRINT
3140 INPUT "UPDATE ANOTHER RESOURCE (Y/N)";ANS$
3150 PRINT
3160 IF LEFT$(ANS$,1)="Y" OR LEFT$(ANS$,1)="y" THEN GOTO 3100
3170 RETURN
3180 RETURN
3190 ' ** RESOURCE DATA INPUT **
3200 CODE% = R
3210 INPUT "SYMBOLIC RESOURCE NAME (3 CHARACTERS):";NM$
3220 IF LEN(NM$) > 3 THEN NK$ = LEFT$(NM$,3)
3230 INPUT "FIXED RESOURCE COST:";FIXCOST$
3240 FIXCOST(R) = VAL(FIXCOST$)
3250 INPUT "VARIABLE COST DUE TO STATION TIME ($/HOUR):";VCOSTS(R)
```

```
3260 INPUT "% UPTIME EXPECTED:";UPTIME(R)
3270 INPUT "INSTALLED COST FACTOR:";XINCOSTFAC(R)
3280 INPUT "TOOL CHANGE TIME (SECONDS)";TCHANGETIME(R)
3290 INPUT "NUMBER OF STATIONS PER WORKER:";STATPERWORK(R)
3300 LSET N$ = NM$
3310 LSET R1$ = FIXCOST$
3320 LSET R4$ = STR$(UPTIME(R))
3330 LSET R9$ = STR$(TCHANGETIME(R))
3340 LSET R3$ = STR$(VCOSTS(R))
3350 LSET R6$ = STR$(XINCOSTFAC(R))
3360 LSET R11$ = STR$(STATPERWORK(R))
3370 PUT #1, CODE%
3380 RETURN
3390 ' ** RESOURCE DATA DISPLAY **
3400 CODE% = R
3410 GET #1, CODE%
3420 LPRINT "FOR RESOURCE # ";R
3430 LPRINT "SYMBOLIC NAME: ";N$
3440 LPRINT "FIXED RESOURCE COST: ";R1$
3450 LPRINT "VARIABLE COST DUE TO STATION TIME ($/HOUR):";R3$
3460 LPRINT "% UPTIME EXPECTED:";R4$
3470 LPRINT "INSTALLED COST FACTOR:";R6$
3480 LPRINT "TOOL CHANGE TIME (SECONDS): ";R9$
3490 LPRINT "NUMBER OF STATIONS PER WORKER:";R11$
3500 RETURN
3510 '
3520 'ROUTINE TO CREATE SUCCESSOR LIST
3530 COLOR 15,1,15
3540 TIME$ = "00:00:00"
3550 OPEN PRODDAT$ AS #2 LEN=44
3560 FIELD #2, 30 AS B1$, 6 AS B2$, 4 AS B3$, 4 AS B4$
3570 CLS
3580 TOTVOL = 0
3590 FOR P = 1 TO NPROD
3600 CODE% = P
3610 GET #2,CODE%
3620 SEQ$(P) = B1$
3630 CYEAR(P) =VAL(B3$)
3640 ANVOL(P) =VAL(B2$)
3650 TOTVOL = TOTVOL + ANVOL(P)
3660 NEXT P
3670 CLOSE
3680 '
3690 ' ESTABLISH BLOCK TASKS
3700 FOR T = 1 TO NTASK
3710  IBLOCK(T) = 1
3720  FOR P = 1 TO NPROD
3730   IF MID$(SEQ$(P),T,1) = "0" THEN IBLOCK(T) = 0
3740  NEXT P
3750 NEXT T
3760 '
3770 I = 1
3780  FOR NODE1 = 1 TO NTASK-1
3790 IFIRST(NODE1) = I
3800   FOR P = 1 TO NPROD
```

```
3810    IF MID$(SEQ$(P),NODE1,1) = "0" THEN 3920
3820    FOR NODE2 = NODE1+1 TO NTASK
3830    IF MID$(SEQ$(P),NODE2,1) = "1" THEN 3860
3840   NEXT NODE2
3850   GOTO 3930
3860   IF P=1 THEN 3900
3870   FOR J = IFIRST(NODE1) TO I
3880   IF ISUC(J) = NODE2 THEN 3920
3890   NEXT J
3900   ISUC(I) = NODE2
3910   I = I + 1
3920   NEXT P
3930  NEXT NODE1
3940  IFIRST(NTASK) = I
3950 '
3960 'READ RESOURCE DATA
3970 '
3980 IF FLAG2 = 1 THEN 4160
3990 OPEN RESDAT$ AS #1 LEN=39
4000 FIELD #1, 3 AS N$, 6 AS R1$, 6 AS R4$, 6 AS R9$, 6 AS R3$, 6 AS R6$,
6 AS R11$
4010 FOR R = 1 TO NRES
4020 IFLAG1(R) = 0
4030 CODE% = R
4040 GET #1, CODE%
4045 RESNAME$(R) = N$
4050 FIXCOST(R) =VAL(R1$)
4060 VCOSTS(R) =VAL(R3$)
4070 UPTIME(R) =VAL(R4$)
4080 XINCOSTFAC(R) =VAL(R6$)
4090 RCHANGETIME(R) = 0
4100 TCHANGETIME(R) =VAL(R9$)
4110 STATPERWORK(R) =VAL(R11$)
4120 RESNAME$(R) = N$
4130 NEXT R
4140 CLOSE
4150 '
4160 'READ TASK DATA
4170 IF IFLAG3 = 1 THEN 4300
4180 OPEN TASKDAT$ AS #1 LEN=16
4190 FIELD #1, 4 AS A1$,6 AS A2$,6 AS A3$
4200 FOR R = 1 TO NRES
4210 FOR T = 1 TO NTASK
4220 CODE% = T + NTASK*(R-1)
4230 GET #1, CODE%
4240 OPTIME(T,R) =VAL(A1$)
4250 TOOLNUM$(T,R) =A2$
4260 TOOLCOST(T,R) =VAL(A3$)
4270 NEXT T
4280 NEXT R
4290 CLOSE
4300 ' Fractions Routine: FRACTIME(P)
4310 INPUT "DO YOU WANT TO INPUT FRACTIONS (Y/N)";ANS$
4320 IF LEFT$(ANS$,1) = "y" OR LEFT$(ANS$,1) = "Y" THEN GOTO 4530
4330 FOR T = 1 TO NTASK
4340 NUMFEAS = 0 : TASKTOT = 0
```

```
4350 FOR R = 1 TO NRES
4360 IF OPTIME(T,R) = 100 THEN GOTO 4390
4370 TASKTOT = TASKTOT + OPTIME(T,R)
4380 NUMFEAS = NUMFEAS + 1
4390 NEXT R
4400 TASKTIME(T) = TASKTOT/NUMFEAS
4410 NEXT T : PRODSUM = 0
4420 FOR P = 1 TO NPROD : TOTTIME = 0
4430 FOR T = 1 TO NTASK
4440 IF MID$(SEQ$(P),T,1) = "0" THEN GOTO 4460
4450 TOTTIME = TOTTIME + TASKTIME(T)
4460 NEXT T
4470 TIME(P) = TOTTIME * ANVOL(P)
4480 PRODSUM = PRODSUM + TIME(P)
4490 NEXT P
4500 FOR P = 1 TO NPROD
4510 FRACTIME(P) = TIME(P)/PRODSUM
4520 NEXT P:GOTO 4560
4530 FOR P = 1 TO NPROD
4540 PRINT "WHAT IS FRACTION OF TIME FOR PRODUCT";P;:INPUT FRACTIME(P)
4550 NEXT P
4560 LPRINT:FOR P = 1 TO NPROD
4570 LPRINT"FRACTION OF TIME FOR PRODUCT";P;"IS";
4580 LPRINT USING "##.#####";FRACTIME(P)
4590 NEXT P:LPRINT
4600 'ENUMERATION OF CUTS
4610 ICUT = 2
4620 FOR N = 1 TO NTASK
4630 M(1,N) = 0
4640 NEXT N
4650 FOR N = 1 TO NTASK
4660 M(ICUT,N) = M(ICUT-1,N)
4670 NEXT N
4680 FOR N = 1 TO NTASK
4690  IF M(ICUT,N) = 0 THEN 4880
4700 NEXT N
4710 '
4720 NCUT = ICUT - 1
4730 INEXT = 1
4740 FOR T = 1 TO NTASK
4750 IF IBLOCK(T) = 0 THEN 4830
4760 FOR ICUT = INEXT TO NCUT
4770  FOR I = 1 TO T
4780   IF M(ICUT,I) = 0 THEN 4820
4790  NEXT I
4800  IBLOCKCUT(ICUT) = 1
4810  INEXT = ICUT:GOTO 4830
4820  NEXT ICUT
4830 NEXT T
4840 '
4850 'IF PROGRAM REACHES THIS POINT, ALL CUTS HAVE BEEN ENUMERATED.
4860 GOTO 4990
4870 '
4880 ITASK = N
4890 M(ICUT,ITASK) = 1
```

```
4900 FOR I = ITASK-1 TO 1 STEP -1
4910 IF M(ICUT,I) = 0 THEN 4960
4920   FOR K = IFIRST(I) TO IFIRST(I+1)-1
4930   IF M(ICUT,ISUC(K)) = 1 THEN 4960
4940   NEXT K
4950  M(ICUT,I) = 0
4960 NEXT I
4970 ICUT = ICUT + 1
4980 GOTO 4650
4990 '
5000 'COMPUTATION OF CYCLE TIME
5010 '
5020 'TAT = TOTAL AVAILABLE TIME (SECONDS)
5030 TAT = DYEAR * SDAY * SHIFT
5040 '
5050 ITER = 1
5060 'RETURN TO HERE FROM CORE PROGRAM
5070 NARCS = 0
5080 MAX = -100
5090 FOR R = 1 TO NRES
5100 IF IFLAG1(R) = 1 THEN 5120
5110 IF RCHANGETIME(R) > MAX THEN NMAX = R:MAX = RCHANGETIME(R)
5120 NEXT R
5130 IF MAX = -100 THEN CLOSE:END
5140 PRINT:FOR P = 1 TO NPROD
5150 CYCTIME(P) = TAT*FRACTIME(P)*3600 - (RCHANGETIME(NMAX) * CYEAR(P))
5160 CYCTIME(P) = (CYCTIME(P)/ ANVOL(P)) * REL/100
5170 LPRINT "CYCLE TIME FOR PRODUCT";P;"IS ";
5180 LPRINT USING "##.#####";CYCTIME(P);:LPRINT" SECONDS"
5190 NEXT P
5200 '
5210 'CALCULATION OF DIFFERENCES BETWEEN CUTS
5220 'M(ICUT,N); N = 1 TO NTASK IS TASK SET
5230 '
5240 TAIL(1) = 1
5250 FOR ICUT = 1 TO NCUT - 1
5260  NHEADS = 0
5270 FOR R = 1 TO NRES
5280  IRESBLOCK(R) = 0
5290 NEXT R
5300 FOR JCUT = ICUT+1 TO NCUT
5310  FOR N = 1 TO NTASK
5320   IDIFF(N) = M(JCUT,N) - M(ICUT,N)
5330  NEXT N
5340  GOSUB 5460
5350 NEXT JCUT
5360 TAIL(ICUT+1) = TAIL(ICUT) + NHEADS
5370 NEXT ICUT
5380 TAIL(NCUT) = 0
5390 '
5400 LPRINT:LPRINT:LPRINT "NCUT=";NCUT
5410 LPRINT "NARCS=";NARCS
5420 'ALL FEASIBLE ARCS HAVE BEEN ENUMERATED. SOLVE SHORTEST PATH PROBLEM.
5430 'CUT TO CUT ARCS ARE STORED IN ARC.DAT
5440 GOTO 6240
```

```
5450 '
5460 'SUBROUTINE TO CHECK FOR FEASIBILITY OF A GIVEN WORKSTATION
5470 '
5480 RMIN = 0
5490  MINACCCOST = 1E+08
5500 FOR R = 1 TO NRES
5510 IF IFLAG1(R) = 1 THEN 6090
5520 IF IRESBLOCK(R) = 1 THEN 6090
5530  FOR T = 1 TO NTASK
5540   IF IDIFF(T) = 1 AND OPTIME(T,R) = 100 THEN 6080
5550  NEXT T
5560  MAXACCTIME = 0
5570  FOR P = 1 TO NPROD
5580   ACCTIME = TRANSTIME
5590   FIRSTTOOL$ = "0"
5600   PRIORTOOL$ = "0"
5610   FOR T = 1 TO NTASK
5620    IF IDIFF(T) = 0 THEN 5720
5630    IF MID$(SEQ$(P),T,1) = "0" THEN 5720
5640    ACCTIME = ACCTIME + OPTIME(T,R)
5650    IF FIRSTTOOL$ = "0" THEN FIRSTTOOL$ = TOOLNUM$(T,R)
5660    THISTOOL$ = TOOLNUM$(T,R)
5670    IF PRIORTOOL$ = "0" THEN 5700
5680    IF THISTOOL$ = PRIORTOOL$ THEN 5700
5690   ACCTIME = ACCTIME + TCHANGETIME(R)
5700   IF ACCTIME > (CYCTIME(P)*UPTIME(R)/100) THEN 6080
5710   PRIORTOOL$ = THISTOOL$
5720  NEXT T
5730 IF TCHANGETIME(R)<= TRANSTIME THEN 5760
5740 IF THISTOOL$ = FIRSTTOOL$ THEN 5760
5750 ACCTIME = ACCTIME + TCHANGETIME(R) - TRANSTIME
5760   IF ACCTIME > (CYCTIME(P)*UPTIME(R)/100) THEN 6080
5770 IF ACCTIME >MAXACCTIME THEN MAXACCTIME = ACCTIME:PMAX = P
5780 NEXT P
5790 '
5800 'IF PROGRAM REACHES THIS POINT ALL PRODUCTS ARE FEASIBLE.
5810 'COMPUTE WORKSTATION COST
5820 '
5830 ACCUMTIME(R)=MAXACCTIME
5840 QQ = 0
5850 FOR T = 1 TO NTASK
5860  IF IDIFF(T) = 0 THEN 5910
5870   QQ = QQ + 1
5880   ACCTOOLCOST = TOOLCOST(T,R)
5890  FIRSTT = T
5900  GOTO 5920
5910 NEXT T
5920 FOR T = FIRSTT+1 TO NTASK
5930 IF IDIFF(T) = 0 THEN 5980
5940 FOR N = 1 TO T-1
5950   IF IDIFF(N) = 1 AND TOOLNUM$(T,R) = TOOLNUM$(N,R) THEN 5980
5960 NEXT N
5970 ACCTOOLCOST = ACCTOOLCOST + TOOLCOST(T,R)
5980 NEXT T
5990 FIXED = FIXCOST(R) *ANCOSTFAC*XINCOSTFAC(R)
```

```
6000 LABOR = TAT *COSTLABOR/STATPERWORK(R)
6010 VARSTATCOST = TAT * VCOSTS(R)
6020 IF RESNAME$(R) = "FXD" THEN VARSTATCOST = VARSTATCOST * QQ
6030 IF RESNAME$(R) = "FXD" THEN LABOR = LABOP * QQ
6040 ACCTOOLCOST = ACCTOOLCOST * XINCOSTFAC(R) * ANCOSTFAC
6050 ACCUMCOST(R) = FIXED + ACCTOOLCOST + LABOR + VARSTATCOST
6060 IF ACCUMCOST(R) < MINACCCOST THEN MINACCCOST = ACCUMCOST(R):RMIN = R
6070 GOTO 6090
6080  IF IBLOCKCUT(JCUT) = 1 THEN IRESBLOCK(R) = 1
6090 NEXT R
6100 '
6110 'LEAST COST RESOURCE IS RMIN.  IF RMIN = 0 THEN NO RESOURCE IS
6120 'FEASIBLE.  COST IS ACCUMCOST(RMIN).  TIME IS ACCUMTIME(RMIN).
6130 'NEXT STEP IS TO FILE THE CUT TO CUT ARC FOR SHORTEST PATH SOLUTION.
6140 '
6150 IF RMIN = 0 THEN RETURN
6160 NARCS = NARCS + 1
6170 NHEADS = NHEADS + 1
6180 HEAD(NARCS) = JCUT
6190 MINRESOURCE(NARCS) = RMIN
6200 ACOST(NARCS) = ACCUMCOST(RMIN)
6210 RETURN
6220 '
6230 '
6240 'ROUTINE TO SOLVE SHORTEST PATH PROBLEM.
6250 '
6260 'DIJKSTRA'S ALGORITHM FOR SHORTEST PATH
6270 '
6280 'INITIALIZE VALUES
6290 '
6300 INFEAS = 0
6310 LET M = ACOST(1)
6320 FOR I = 2 TO NARCS
6330  IF ACOST(I) >= M THEN M = ACOST(I)
6340 NEXT I
6350 M = 1000*M
6360 FOR I = 2 TO NCUT
6370  PRED(I) = 0
6380  S(I) = 0
6390  MCOST(I) = M
6400  MINARC(I) = 0
6410 NEXT I
6420 FOR J = TAIL(1) TO TAIL(2)-1
6430  MCOST(HEAD(J)) = ACOST(J)
6440 PRED(HEAD(J)) = 1
6450 MINARC(HEAD(J)) = J
6460 NEXT J
6470 S(1) = 1
6480 MCOST(1) = 0
6490 PRED(1) = 0
6500 MINARC(1) = 0
6510 '
6520 IPRED = 0
6530 'ASSIGN PERMANENT LABELS
6540 '
```

```
6550 MINNODE = 1
6560 MINCOST = M
6570 REMPTY = 1
6580 FOR I = 2 TO NCUT
6590  IF S(I) = 1 THEN 6640
6600  REMPTY = 0
6610  IF MINCOST <= MCOST(I) THEN 6640
6620  MINCOST = MCOST(I)
6630  MINNODE = I
6640 NEXT I
6650 IF REMPTY=1 THEN 6900
6660 S(MINNODE) = 1
6670 IF IPRED = 1 THEN 6740
6680 IPRED = 1
6690 PRED(MINNODE) = 1
6700 FOR J = TAIL(1) TO TAIL(2)-1
6710 IF HEAD(J) = MINNODE THEN MINARC(MINNODE) = J
6720 NEXT J
6730 '
6740 'UPDATE ALL LABELS
6750 '
6760 IF MINNODE = NCUT THEN 6900
6770 INFEAS = INFEAS + 1
6780 IF INFEAS >= NCUT^2 + 50 THEN LPRINT "PROBLEM IS INFEASIBLE":END
6790 A = TAIL(MINNODE)
6800 B = TAIL(MINNODE+1)-1
6810 IF B = -1 THEN B = NARCS
6820 FOR I = A TO B
6830  IF MCOST(HEAD(I))< ACOST(I) + MCOST(MINNODE) THEN 6870
6840  MCOST(HEAD(I)) = ACOST(I) + MCOST(MINNODE)
6850  PRED(HEAD(I)) = MINNODE
6860  MINARC(HEAD(I)) = I
6870 NEXT I
6880 GOTO 6530
6890 '
6900 'OUTPUT REPORT
6910 I = 1
6920 N = NCUT
6930 ARC(I) = MINARC(N)
6940 FOR T = 1 TO NTASK
6950 INTASK(I,T) = M(N,T) - M(PRED(N),T)
6960 NEXT T
6970 N = PRED(N)
6980 IF PRED(N) = 0 THEN 7010
6990 I = I + 1
7000 GOTO 6930
7010 NUMARCS = I
7020 APPTOTAL = 0
7030 ACTTOTAL = 0
7040 TOTALTIME = 0
7050 '
7055 LPRINT CHR$(12)
7060 LPRINT:LPRINT TAB(35) "INDIVIDUAL PRODUCTS REPORT"
7070 '
7080 FOR P = 1 TO NPROD
```

```
7085 LPRINT
7090 LPRINT:LPRINT TAB(40) "FOR PRODUCT";P
7100 MAXACCTIME = -100
7104 LPRINT:LPRINT"                                   ACCUMULATED"
7105 LPRINT"         STATION   RESOURCE       STATION TIME      TASKS"
7110 FOR I = NUMARCS TO 1 STEP -1
7115 J = NUMARCS - I + 1
7120 R = MINRESOURCE(ARC(I))
7130 ACCTIME = TRANSTIME
7140   FIRSTTOOL$ = "0"
7150   PRIORTOOL$ = "0"
7160 TASK$ = ""
7170 FOR T = 1 TO NTASK
7180 IF INTASK(I,T) = 0 OR MID$(SEQ$(P),T,1) = "0" THEN 7280
7190 IF TASK$ = "" THEN TASK$=TASK$+RIGHT$(STR$(T),LEN(STR$(T))-1):GOTO 7210
7200 TASK$ = TASK$ + STR$(T) + " "
7210 ACCTIME = ACCTIME + OPTIME(T,R)
7220    IF FIRSTTOOL$ = "0" THEN FIRSTTOOL$ = TOOLNUM$(T,R)
7230    THISTOOL$ = TOOLNUM$(T,R)
7240    IF PRIORTOOL$ = "0" THEN 7270
7250    IF THISTOOL$ = PRIORTOOL$ THEN 7280
7260   ACCTIME = ACCTIME + TCHANGETIME(R)
7270   PRIORTOOL$ = THISTOOL$
7280 NEXT T
7290 IF TCHANGETIME(R)<= TRANSTIME THEN 7320
7300 IF THISTOOL$ = FIRSTTOOL$ THEN 7320
7310 ACCTIME = ACCTIME + TCHANGETIME(R) - TRANSTIME
7320 LPRINT
7325 STATIME = ACCTIME -TRANSTIME
7330 LPRINT TAB(14) J; TAB(24) RESNAME$(R); TAB(34);
7340 LPRINT USING "###.##";STATIME;
7350 LPRINT " seconds";
7360 IF LEN(TASK$) <= 25 THEN LPRINT TAB(53) TASK$ : GOTO 7410
7370 ZZ = 0
7380 IF MID$(TASK$,25-ZZ,1) = "  " THEN LPRINT TAB(53)LEFT$(TASK$,25-ZZ)
       ELSE ZZ = ZZ + 1 : GOTO 7380
7390 TASK$ = RIGHT$(TASK$,LEN(TASK$)-25+ZZ)
7400 GOTO 7360
7410 IF ACCTIME > MAXACCTIME THEN MAXACCTIME = ACCTIME
7420 NEXT I
7430 TOTALTIME = TOTALTIME + (MAXACCTIME*ANVOL(P))
7440 NEXT P
7450 LPRINT CHR$(12)
7455 LPRINT
7460 LPRINT:LPRINT TAB(35) "GENERAL REPORT":LPRINT
7462 LPRINT:LPRINT "          ANNUAL VOLUME: ";TOTVOL;"
     NUMBER OF SHIFTS/DAY: ";SDAY
7463 FOR P = 1 TO NPROD
7464 LPRINT: LPRINT"                    CYCLETIME PRODUCT(";P;"): ";
     CYCTIME(P);" SECONDS"
7465 NEXT P
7466 LPRINT
7470 '
7472 LPRINT
7475 LPRINT "                               APPARENT   ADJUSTED
```

```
7470 LPRINT                 STATION    RESOURCE      COST         COST
     TASKS         TOOLS"
7480 FOR I = NUMARCS TO 1 STEP -1
7482 LPRINT
7485 J = NUMARCS - I + 1
7490 R = MINRESOURCE(ARC(I))
7500 VARSTATCOST = TAT*(VCOSTS(R)+COSTLABOR/STATPERWORK(R))
7510 'ADJUST VARIABLE COSTS
7520 ACCCOST = ACOST(ARC(I)) - VARSTATCOST
7530 VARSTATCOST = TOTALTIME*(VCOSTS(R)+COSTLABOR/STATPERWORK(R))/3600
7540 VARSTATCOST = VARSTATCOST/(UPTIME(R)/100*REL/100)
7550 ACCCOST = ACCCOST + VARSTATCOST
7580 LPRINT TAB(14) J; TAB(24) RESNAME$(R); TAB(32) "$";ACOST(ARC(I));
     TAB(43) "$";ACCCOST;
7590 TOOL$ = "":TASK$ = "":FOR T = 1 TO NTASK
7600 IF INTASK(I,T) <> 1 THEN 7690
7610 IF TASK$="" THEN TASK$=RIGHT$(STR$(T),LEN(STR$(T))-1):GOTO 7630
7620 TASK$ = TASK$ + STR$(T)
7630 IF RIGHT$(TOOLNUM$(T,R),1) = " " THEN TOOLNUM$(T,R) =
     LEFT$(TOOLNUM$(T,R),LEN(TOOLNUM$(T,R))-1):GOTO 7630
7640 IF LEN(TOOL$) = 0 THEN 7680
7650 FOR QQ = 1 TO LEN(TOOL$)
7660 IF MID$(TOOL$,QQ,LEN(TOOLNUM$(T,R))) = TOOLNUM$(T,R) THEN GOTO 7690
7670 NEXT QQ
7680 TOOL$ = TOOL$ + TOOLNUM$(T,R) + " "
7690 NEXT T
7700 IF LEN(TASK$) < 11 THEN LPRINT TAB(56) TASK$; : TASK$ = "" : GOTO 7740
7710 ZZ = 0
7720 IF MID$(TASK$,11-ZZ,1) = "  " THEN LPRINT TAB(56) LEFT$(TASK$,11-ZZ);
     ELSE ZZ = ZZ + 1 : GOTO 7720
7730 TASK$ = RIGHT$(TASK$,LEN(TASK$)-11+ZZ)
7740 IF LEN(TOOL$) < 8 THEN LPRINT TAB(70) TOOL$ : TOOL$ = "" : GOTO 7780
7750 ZZ = 0
7760 IF MID$(TOOL$,8-ZZ,1) = " " THEN LPRINT TAB(70) LEFT$(TOOL$,8-ZZ)
     ELSE ZZ = ZZ + 1 : GOTO 7760
7770 TOOL$ = RIGHT$(TOOL$,LEN(TOOL$)-8+ZZ)
7780 IF TOOL$ <> "" OR TASK$ <> "" THEN GOTO 7700
7790 APPTOTAL = APPTOTAL + ACOST(ARC(I))
7800 ACTTOTAL = ACTTOTAL + ACCCOST
7810 NEXT I
7815 LPRINT
7820 LPRINT:LPRINT USING "          TOTAL APPARENT COST       $$##########";
     APPTOTAL
7830 LPRINT:LPRINT USING "          ADJUSTED SYSTEM COST      $$##########";
     ACTTOTAL
7840 LPRINT:LPRINT "          TOTAL TIME FOR REPORT RUN ==> ";TIME$:END
7850 IFLAG1(NMAX) = 1
7860 FOR R = 1 TO NRES
7870 IF R = NMAX THEN 7890
7880 IF RCHANGETIME(R) = MAX THEN IFLAG1(R) = 1
7890 NEXT R
7900 ITER = ITER + 1
7910 GOTO 5070
```

BIBLIOGRAPHY

1. Baybars, Ilker, "A Survey of Exact Algorithms for the Simple Assembly Line Balancing Problem," Management Science, 32(1986), 909-932.

2. DeFazio, T.L., and D.E. Whitney, "Simplified Generation of All Mechanical Assembly Sequences," C.S. Draper Laboratory Report No. P-2709, June 1986.

3. Dijkstra, E. W., "A note on Problems in Connexion with Graphs," Numerical Mathematics, 1 (1959), 269-271.

4. Graves, S. C. and B. W. Lamar, "An Integer Programming Procedure for Assembly System Design Problems," Operations Research, 31 (1983), 522-545.

5. Graves, S. C. and D. E. Whitney, "A Mathematical Programming Procedure for Equipment Selection and System Evaluation in Programmable Assembly," Proceedings of the Eighteenth IEEE Conference on Decisions and Control (1979), 531-536.

6. Gustavson, R. E., "Multiple Product Assembly System Design," C.S. Draper Laboratory MAT MEMO NO: 1303, Cambridge, 1986.

7. Gustavson, R. E., "Design of Cost-Effective Assembly Systems," C.S. Draper Laboratory Report No. P-2661, Cambridge, 1986.

8. Halevi, G., The Role of Computers in Manufacturing Processes, John Wiley & sons, New York, 1980.

9. Hollingum, J., "A Philosophical View of the Need for Flexibility," Assembly Automation, February 1986.

10. Kilbridge and Webster., "A Heuristic Method of Assembly Line Balancing," The Journal of Industrial Engineering, Vol. XII, 4, July-August 1961.

11. Lamar, B. W., Optimal Machine Selection and Task Assignment in an Assembly System Design Problem, Master's Thesis, Operations Research Center, Massachusetts Institute of Technology, 1980.

12 Pinto, P. A., D. G. Dannenbring and B. M. Khumawala, "Assembly Line Balancing with Processing Alternatives: An Application," Management Science, 29, July 1983.

13. Schrage, L. and K.R. Baker, "Dynamic Programming Solution of Sequencing Problems with Precedence Constraints," Operations Research, 26, June 1978.

14. Talbot, F. B., J. H. Patterson and W. V. Gehrlein, "AComparative Evaluation of Heuristic Line Balancing Techniques," Management Science, 32, April 1986.

15. Thomopolous, N. T., "Line Balancing-Sequencing for Mixed-model Assembly," Management Science, 14, October 1967.