


UNIVERSITY OF
ILLINOIS LIBRARY
AT URBANA-CHAMPAIGN
BOOKSTACKS



Digitized by the Internet Archive
in 2011 with funding from
University of Illinois Urbana-Champaign

30
885
STX
1158 COPY 2



BEBR

**FACULTY WORKING
PAPER NO. 1158**

COMMERCE LIB-ADM

JUL 23 1985

UNIVERSITY OF ILLINOIS
URBANA-CHAMPAIGN

A Knowledge-Based Approach to Flexible
Planning and Scheduling

Michael J. Shaw
Andrew B. Whinston

College of Commerce and Business Administration
Bureau of Economic and Business Research
University of Illinois, Urbana-Champaign

BEBR

FACULTY WORKING PAPER NO. 1158

College of Commerce and Business Administration

University of Illinois at Urbana-Champaign

July, 1985

A Knowledge-Based Approach to Flexible
Planning and Scheduling

Michael J. Shaw, Assistant Professor
Department of Business Administration

Andrew B. Whinston
Purdue University

Abstract

Planning in flexible manufacturing systems must take into account both the multiprocessing environment and the dynamically changing states. This paper uses a knowledge-based approach to handle such a planning system, where three levels--i.e., the control, data, and procedural levels--constitute its manufacturing knowledge. The construction of plans for multiple manufacturing jobs requires four steps: (1) linear plan generation, (2) conflict detection, (3) plan synthesis, and (4) plan revision. To achieve goals, these steps will search for planning operators using a backward chaining inference procedure.

To perform the scheduling and sequencing of the multiple jobs within a flexible manufacturing cell, the planning system modifies the nonlinear planning approach and adds resources and durations information to the action formalism. Because the plan-generation process is goal-directed, the dynamically adjustable plan can be constructed in real time. A prototype of this automatic planning system has been implemented in Franz LISP, an artificial intelligence programming language, and is applied to solve the real-time scheduling problem in flexible manufacturing cells.

1. Introduction

Flexible automation--automation that can handle a large and constantly changing variety of produced items--has played an increasingly important role in the efforts to improve the productivity of the manufacturing industry (Hutchingson [12], Merchang [14]). The recent progress in computer technologies has accelerated the realization of flexible automation. The use of computers in manufacturing, such as the numerically controlled (NC) machines, adds programmability and thus versatility into manufacturing systems. More important, computers also provide on-line execution of manufacturing planning and decision making. These two capabilities, computerized control and on-line planning, are integrated into a well-orchestrated, automated manufacturing system that can produce wide-ranging items efficiently.

Characterized by dynamically changing states and a multiprocessing environment, a flexible manufacturing system requires a planning approach adapted to distributed environments with interactive processes. Generally, a planning system develops a course of action or a "plan" for the processors that reaches the goals desired; the plan will then be used to guide the execution of activities. In such flexible manufacturing, the processors--which may be robots, computerized machine-centers, or the host computer of a flexible manufacturing cell--can carry out a variety of activities, including various types of machining, workpiece routing, loading, and unloading operations. The planning approach developed in this paper is designed to derive and coordinate manufacturing steps in real time to fulfill the goals of efficiently transforming raw material into finished products.

Within such a planning framework, the manufacturing process corresponding to each job is modeled by state-changing transformations. Since the system usually works on several different jobs at once, proper coordination is essential. If the manufacturing processes for different jobs are independent, then, in principle, they can be executed in parallel. In reality, however, workpieces usually have to share machines, tools, and other resources. Therefore, the manufacturing processes operating on them must interact. The idea is to coordinate the planned activities for each job so that each manufacturing operation is performed by the most capable machine available, thereby making efficient use of the machines.

Time is an important parameter for the planning system, and it must include not only the time span of the plan but also the times at which each activity occurs. Usually, the objective for such planning is to minimize the total time taken for completing the jobs (Baker [1]). To take this objective into account, the planning system represents time explicitly in the knowledge base and uses sensitivity analysis to ensure that due dates are satisfied. Finally, the planning system considers alternative operations and revises existing plans if any bottlenecks are detected.

The remainder of this paper is organized as follows. Section 2 characterizes the planning and scheduling problem in flexible manufacturing systems and discusses the approaches used in earlier research. Section 3 lays out the framework of the knowledge-based approach for real-time planning and scheduling in flexible manufacturing systems. lastly, Section 4 shows the extensions to the planning approach,

including the due-date consideration, dynamic scheduling and the two-level planning problem in computer integrated manufacturing systems.

2. The Planning and Scheduling in Manufacturing Cells

2.1 The Problem

As a flexible manufacturing system, a manufacturing cell is usually a modular unit in a computer-integrated manufacturing system (Cutkosky [6], Shaw and Whinston [20], [22]). A typical manufacturing cell has several computer-controlled machines and robots, with an automatic handling system transporting parts between machines. Such integrated systems are characterized by their ability to make different parts and to perform a wide range of operations.

The problem may be stated thus: A certain number, say n , parts are assigned to the manufacturing cell; each part requires a given set of linearly sequenced operations to be performed by the m machines. Since the machines have varying efficiencies for different operations, each part is routed among the machines so that every operation it needs is performed by the most appropriate machine available.

The objective of the problem is to schedule the n parts concurrently by developing a schedule for each part traveling among the machines; the makespan--the duration taken for completing all the required operations--should be minimized, while avoiding any conflicts arising from assigning parts to busy machines.

The scheduling problem in the flexible manufacturing cell has these characteristics:

- (1) Jobs consist of linearly ordered operation sequences (as a result of process planning).
- (2) A given operation can be performed on several alternative machines with different processing durations.
- (3) Each machine, while capable of performing a variety of operations, can execute only one operation at a time.

This scheduling problem can be formulated as an integer programming problem which captures all these characteristics:

Decision Variables:

- X_{ijk} : the completion time of operation j of job i on machine k .
- Y_{ijpq} = 1 if operation q of job p precedes operation j of job i ;
= 0 otherwise.
- Z_{ijpq} = 1 if operation j of job i precedes operation q of job p ;
= 0 otherwise.

Constants:

- t_{ijk} : The processing duration of operation j of job i on machine k .
- M : a very large positive number.
- $l(i)$: the last operation of job i .

(A) Minimize $\sum_{i=1}^n X_{il(i)k}$

Subject to

(B) $X_{ijk} - X_{i,j-1,h} \geq t_{ijk}$

(C) $X_{pqk} - X_{ijk} + M \cdot Y_{ijpq} \geq t_{pqk}$

$$(D) X_{ijk} - X_{pqk} + M \cdot Z_{ijpq} \geq t_{ijk}$$

$$(E) Y_{ijpq} + Z_{ijpq} \geq 1$$

$$(F) X_{ijk} \geq 0; Y_{ijpq}, Z_{ijpq} = 0 \text{ or } 1$$

$$1 \leq i, p \leq n; 1 \leq j, q \leq \max_l(i); 1 \leq h, k \leq m$$

The objective function described in (A) is to minimize the average duration needed for each job to complete. An alternative objective is to minimize the total duration of the schedule, i.e., the time when all the jobs are completed. This objective function can be described as:

$$\text{Minimize Maximize } \{X_{i, l(i), k}\}$$

The constraint set (B) represents the linear ordering of operations in a job. These constraints also impose the condition that an operation may not begin until its predecessors are completed.

Constraints (C), (D), and (E) are used to regulate the mutually exclusive condition of machine sharing. For a pair of manufacturing operations consisting of operation j of job i and operation q of job p, one of the following three situations may occur:

- (i) they are performed on the same machine; operation j of job i precedes operation q of job p.
- (ii) they are performed on the same machine; operation q of job p precedes operation j of job i.
- (iii) they are performed on different machines. In this case, there is no constraint on their precedence ordering.

In the formulation, condition (i) is represented by $Z_{ijpq} = 1$ and $Y_{ijpq} = 0$; similarly, condition (ii) is represented by $Z_{ijpq} = 0$ and $Y_{ijpq} = 1$; lastly, condition (iii) is represented by $Z_{ijpq} = Y_{ijpq} = 1$.

2.2 Related Research

In the operations research literature, the methods that have been applied to the planning and scheduling problems can be categorized into two types: (1) mathematical programming (Baker [1], French [11]), or (2) myopic dispatching heuristics determined by simulation (Moore and Wilson [15], Rochamadugu [17]). Both methods have been applied to the flexible manufacturing environment. Chang et al. [4] tested various dispatching rules for dynamic scheduling in a flexible manufacturing system and reported the performance of these rules produced by simulation programs. However, the simulation method has a critical restriction: it does not give any solutions. Rather, it only tests the resulting performance so that the best heuristic rule of the given system can be selected. In an effort to apply mathematical programming techniques, Stecke [23] argued that the planning problem in a flexible manufacturing system consists of five problems: (1) part-type selection, (2) machine grouping, (3) production ratio, (4) resource allocation, and (5) machine loading. Stecke uses linearized mixed integer programming methods to solve these problems. The results indicate that the linearization of the problem significantly reduces the computational complexity without seriously degrading the quality of the solution. She suggested that large problems cannot be feasibly handled by this method and may require the use of heuristics.

The inability of the operations research methods to handle such dynamic environments as the flexible manufacturing system can be remedied by the knowledge-based planning method. The STRIPS system, an early planning system developed in the artificial intelligence discipline (Fikes and Nilson [7], [8]), used an inference procedure to construct plans that can guide the robot's motions in accomplishing specified goals. Sacerdoti [18], Tate [24], and Nilson [16] extended the planning method used by STRIPS to deal with more complex situations where the planning system has to satisfy multiple goals. Since, in these situations, there are interactions between planning activities for the different goals, some mechanism is needed to account for these interactions (to avoid any potential conflicts); the planning system, in turn, establishes constraints between the activities involved. Because such plans are only partially ordered, these methods are referred to as nonlinear planning.

Vere [25] applied the nonlinear planning approach to scheduling space shuttles by a dynamic control scheme. In it, the time of the activities is described formally as a parameter in the planning system, and the goals are posted as functions of their ending times. Wilkins ([26], [27]) developed a domain-independent planning system featuring explicit resource coordination. For such planning domains as those in the flexible manufacturing system, where the detection of conflicts in resource assignment is essential, Wilkins' method can be more efficient than ordinary planning methods. Fox ([9], [10]) developed a planning and scheduling system by a constraint-directed approach for a large manufacturing system. In it, the key part of the search for

machine schedules is the application of constraints to reduce the search space, thus speeding up the planning process. Although his target system is a large-scale job shop, the environment actually resembles that of a flexible manufacturing system since it is automated and computer-controlled.

In our approach, an n -part- m -machine scheduling problem can be decomposed into n subproblems, with each subproblem defined as the routing of one part. The nonlinear planning method discussed above can thus be utilized to generate a plan for the n subproblems; the primary "interactions" between these subproblems are their sharing of the m machines. The objectives of the scheduling problem--to minimize makespan and to avoid conflicting assignments--can be translated into the criteria for the plan-generation problem: to maximize the parallelism and to avoid harmful interactions among the subplans. Specifically, our planning system uses a knowledge-based approach, as explained in the next section.

3. Knowledge-Based Planning

3.1 The Framework

The planning system, when organized as a knowledge-based system, treats knowledge on three levels: data, knowledge base, and control. By contrast, conventional programs treat knowledge on only two levels: data and program. At the data level a knowledge-based system stores declarative knowledge about the goals, the current situation of the world, and the semifinished plan. At the knowledge-base level is stored the domain-specific, procedural knowledge. This knowledge

models the actions of the world, and it is often represented by production rules or operators. Finally, at the control level the knowledge about the strategy of plan construction is stored; this is the control knowledge indicating how to select operators, and when to apply them.

Because planning involves exploration of alternative sequences of actions, a symbolic model of the real world, the "world model," represents the environment as the plans evolve. For any given planning problem, the initial condition and the stated goal condition are both treated as instances in the world model. The generation function of a planning system, then, is to construct a course of action that transforms one state of the world model, which contains an initial condition, to a state which satisfies the goal condition. Thus the planning system we have developed has three basic components:

(1) the world model, which contains a symbolic description of the real world. This world model is represented by the collection of first-order predicates in a database. Any instance of the database is called a state of the world model. Examples of such database elements in a world model are shown in Table 1.

Insert Table 1 Here

(2) The action model, which describes the transformational effects of actions that map states to other states. Such transformations are usually modeled by operators similar to the STRIPS operators defined in Fikes and Nilson [7]. In such an action model, each operator can be specified

< Action - name > : < list-of-arguments >
< Precondition > : < list-of-precondition-literals >
< Add list > : < list-of-add-list-literals >
< Delete list > : < list-of-delete-list-literals >
< Resource > : < resource-name >
< Duration > : < length-of-duration >

In addition to the standard STRIPS formalism--which specifies an action by the add list, delete list, and preconditions--we have also included two more descriptions for each action--the "resource" used during the action, and the "duration" of the action. There are two advantages to this addition: the increased representational power of the action model and the resulting acceleration of conflict detection and conflict resolution. An example is shown in Figure 1.

Insert Figure 1 Here

(3) The inference engine, which directs the plan generation process. It selects a sequence of operators to achieve the goal state from a given initial state.

The linear plans can be generated by any STRIPS-like plan generation system (Fikes and Nilson [7], [8]). Such a planning system can use a backward-chaining method in searching for the best actions--i.e., it works backward from the goal state and find a sequence of actions that could produce this goal state from the initial state. The process of plan generation, then, can be viewed as finding the solution path in a search tree. The root of the tree is the goal state, and instances of operators define the branches. The solution

path starts with the root (the goal state) and leads to the leaves (the initial state), thereby defining the plan. An example of the search trees generated by the planning system are depicted in Figure 2.

Insert Figure 2 Here

Within this planning framework, the manufacturing process corresponding to each task is modeled by state-changing transformations, represented by operators. If the manufacturing processes for different tasks are independent, then, in principle, they can be executed in parallel. In reality, however, different tasks usually are competing for machines, tools, and other resources; therefore, the planning system must take into account the interactions among the processes. In the planning literature, this interaction problem between tasks have been treated by imposing constraints between planning steps to avoid any potential conflicts (Sacerdoti [18], Vere [25], Wilkins [27]). Two kinds of schemes have been used for this purpose: the "critic" mechanism (Sacerdoti [16]), and the "reasoning about resource" scheme (Wilkins [27]). Because the plans generated by these methods are partially ordered, these methods are called "nonlinear planning."

3.2 The Conflict-Detection Mechanism

After a linear plan is constructed for each subgoal, the next step is to identify problematic interactions between parallel actions. The primary cause of such interactions is the potential conflicts in using resources. There are two possible approaches in this step: (1) a "critic mechanism," or (2) a "reasoning about resources" scheme. Let us consider each in turn.

The critic mechanism comes from the method used in NOAH (Sacerdoti [18]) and DCOMP (Nilson [16]). In these, the interaction-detection mechanism--called the "critic"--of the planning system identifies potentially harmful interactions between planning steps by checking the effects of the operators involved. If the preconditions of one operator are deleted by another, earlier operator, these preconditions must be added back by yet another operator, which will come between the two original operators. Thus, to test for potential conflicts facing an operator, two kinds of information are crucial: those operators in the plan that can delete its preconditions, and those operators in the plan that can result in its preconditions; the former are recorded in an "adder list," the latter are recorded in a "deleter list." These two lists are parts of a table kept by the planner, referred to as the table of multiple effects (TOME).

For instance, with the scheduling problem shown in the Appendix, if the plan developed so far is as follows:

$$\begin{array}{c} \rightarrow p1 \rightarrow p2 \rightarrow p3 \rightarrow p4 \\ \quad \quad \quad \downarrow \\ \quad \quad \quad \rightarrow q1 \rightarrow q2 \end{array}$$

According to the linearly-sequenced plan for the "q" process, the next operator to apply is q3. However, by checking with TOME, p3, in a parallel branch, deleted the preconditions of q3; thus, there is a risk of conflict between these two operators' actions if we do not restrict their invocations. And q3 must not be applicable until p5 is finished; otherwise, there would be a conflict. This can be explicitly

Mutual exclusion between critical sections can be enforced by semaphors, as used for concurrency management in multiprocessing operating systems (Brinch Hansen [3]). A semaphore is an integer variable shared by subplans; each resource is associated with one semaphore. The value of a semaphore, either zero or one, is used to signal the status of the resource. When the semaphore is one, the resource is available; when the semaphore is zero, the resource is occupied. Using such a semaphore mechanism, a conflict-detection procedure based on resource reasoning can be implemented by a program module called "resource manager."

3.3 Application to Real-Time Scheduling

For the flexible manufacturing environment, Shaw and Whinston [19], [20] used a plan-generation scheme, similar to the nonlinear planning approach, to derive the desired production plans within each cell. The scheme requires four steps:

The Nonlinear Planning Scheme

- Step 1. Generate a linearly-sequenced plan for each task.
- Step 2. Identify problematic interactions between the planning steps.
- Step 3. Use precedence constraints to avoid conflicts.
- Step 4. Identify alternative planning steps to improve the performance.

An example of the partially ordered plan resulting from the scheme is shown in Figure 3, where each of the nodes represents a manufacturing planning operator.

Insert Figure 3 Here

In step 1 of the planning scheme, the inference engine calls upon a backward chaining procedure to search for the best planning steps in constructing the manufacturing plan for each part. In our program, this is carried out by a procedure, called OPERATOR-SEARCH, based on "means-ends analysis" heuristic (Nilson [16]). The search tree generated by this step, as shown in Figure 2, results in a set of linearly ordered planning operators (some examples are shown in the Appendix).

"PLAN-AHEAD," as we have termed a plan-generation procedure that executes steps 2 and 3 of nonlinear planning for conflict detection and resolution, dynamically decides the precedence relationship between two conflicting actions. The underlying principle--based on the least commitment strategy--is not to impose any precedence constraint unless it is absolutely necessary, so that the parallelism among the subplans is maximized. Information about resources and duration of actions is crucial to the inference engine in making these decisions. The flow-chart of PLAN-AHEAD IS shown in Figure 4.

Insert Figure 4 Here

Step 4 of the nonlinear planning scheme employs a method that reassigns waiting jobs to alternative resources so as to achieve better utilization and performance as much as possible. This procedure, called Plan-Revision, can be described as follows.

The Plan-Revision Scheme

- step 1. Identify the resource for which this job is waiting.
- step 2. Locate the section of the subplan that would use this resource.
- step 3. Evaluate the expected waiting time vs. the additional processing time by an alternative, idle resource.
- step 4. Find out the initial conditions and the ending conditions of this section.
- step 5. Generate a plan that can transform the initial conditions to the goal conditions, using another idle resource.
- step 6. Modify the subplan by replacing the section identified in step 2 with the newly generated plan from step 4.

The planning steps embedded in the plan-revision scheme are shown in Figures 5 and 6. Specifically, they correspond to steps 4 and 5 of the scheme.

Insert Figures 5 and 6 Here

The functional organization of the knowledge-based program that executes the four-phased nonlinear planning procedure is summarized in Figure 7. This program has been implemented with Franz LISP, an artificial intelligence programming language, in VAX 11/780.

Insert Figure 7 Here

This approach displays some desirable characteristics for real-time planning and scheduling in the flexible manufacturing environment. First, it is goal-directed, i.e., users only need to specify the goals of the manufacturing process and the planning system would,

accordingly, derive the necessary steps on-line. Second, it is dynamically adjustable. New goals can be accommodated while the current production plan is still being executed; also, plans can be modified when unexpected events occur (e.g., tool or machine breakdowns). A plan-revision scheme is initiated when bottlenecks are detected; the scheme, in turn, seeks to use alternative resources to improve the throughput. Additional considerations should be given to the travel paths taken by guided carts or the arm movements of neighboring robots so that any potential conflicts or harmful interferences are avoided (Bourne and Fussell [2], Lozana-Perez [13]).

4. Extensions to the Planning System

4.1 Priority Scheduling with the Due-dates Consideration

If the jobs are ordered by their priorities, then EVENT-LIST can function as a priority queue. For every moment in time, every planned action holds a priority number. Whenever a resource becomes idle, PLAN-AHEAD will assign the resource to the action with the highest priority. When the due-date is critical, the planning system must perform such priority scheduling to determine the final plan.

First, a preliminary plan, a partially ordered network of actions, must be made for the jobs. Then, working backward from the due date, priority scheduling enables the planning system to assign the "latest starting time" for each action. Next, the planning system compares the actual starting time of an action with the latest starting time just calculated. The priority number of the job expresses how close it is to the due-date. In general, the more urgent a job is, the higher its

priority will be, thereby ensuring its on-time completion. Accordingly, the machine will start a low priority job only if no jobs of higher priority are present. And, to keep everything in order, once the machine is assigned a job, under the non-preemptive assumption, it is committed to serve that job to completion even if jobs of higher priority arrive in the meantime.

4.2 Dynamic Scheduling

In flexible manufacturing systems, jobs arrive at a manufacturing cell dynamically, each requiring a variety of operations. For such systems, the knowledge-based planning approach works best because of its structured knowledge representations and the conflict-resolving inference capability. The current status of the cell, the progress of the on-going plan, and the utilization of the machines are all included and updated in the world model. When new jobs need to be scheduled during the execution of existing jobs, a simple plan-modification procedure, such as the following one, can be invoked to accommodate the new jobs.

Step 1. Establish plans for the new jobs based on the currently available machines.

Step 2. Use the conflict-resolution scheme to coordinate the actions for the new jobs and the remaining actions for the old jobs.

Step 3. Improve the modified plan by the same plan-revision scheme.

Because previously generated plans are stored in the data base with structured knowledge representation, new jobs change plans only where they interact with the new jobs. This concept originated in the STRIPS

planning system, where macro-operators and structured plan-representation assisted both in the solution of similar problems and also in the intelligent monitoring of the plans' execution (Fike and Nilson [9]). Dynamically adjustability, in short, is one of the primary advantages of the knowledge-based planning approach.

4.3 The Generalized Planning Problem in the Computer Integrated Manufacturing System

An emerging architecture for computer integrated manufacturing systems is the cellular system which consists of a collection of flexible manufacturing cells, each of which serves a specific part family. The planning and scheduling system discussed in the present paper is executed by each cell's host computer, its control unit. The planning problem for the whole CIMS is itself a two-level problem: the first-level planner distributes jobs among cells according to the capabilities and the set-up of each individual cell; the second-level planner--the one described in this paper--in turn performs the planning and scheduling within each cell. To achieve good performance, flexible manufacturing cells must coordinate and communicate with each other through a local area network. Shaw and Whinston [19] analyzed distributed task allocation and modeled the first level planning problem by a distributed bidding algorithm. That paper developed a variant of the knowledge-based system that, guided by augmented Petri nets, acts as the first-level planner. Integration of the two planning systems can thus result in a distributed knowledge-based system, with the knowledge for both the job allocation and the inter-cell scheduling incorporated into the knowledge base in each cell.

5. Conclusion

In this paper, planning in the flexible manufacturing environment has been investigated in the context of a knowledge-based approach in order to handle the dynamically changing environment, interactions between manufacturing processes, and the versatility of processors. The nonlinear planning method developed in artificial intelligence has been extended so that the duration and resource information of each action is immediately derivable. The planning system can effectively schedule jobs in a flexible manufacturing cell and dynamically determine the routing of workpieces among the processors. The resulting plan, a partially ordered network, demonstrates maximal parallelism and shortest duration.

Appendix

Shaw and Whinston [20] shows the application of the nonlinear planning approach to the scheduling problem in a three-machine manufacturing cell. To simplify the situation, suppose there are two independent jobs to be scheduled. The first job, requiring operations OP1, OP2, and OP3, is assigned the following routing by step 1 of nonlinear plan construction-linear planning:

```
p1  ENTER
p2  EXECUTE(LOAD,DOCK)
p3  TRANSFER(DOCK,M1)
p4  EXECUTE(M1,OP1)
p5  TRANSFER(M1,M2)
p6  EXECUTE(M2,OP2)
p7  TRANSFER(M2,M3)
p8  EXECUTE(M3,OP3)
p9  UNLOAD(M3,DOCK)
p10 EXIT
```

Similarly, job 2, requiring operations OP1, OP3, will be assigned the following linear routing

q1 ENTER
q2 EXECUTE(LOAD,DOCK)
q3 TRANSFER(DOCK,M1)
q4 EXECUTE(M1,OP1)
q5 TRANSFER(M1,M3)
q6 EXECUTE(M3,OP3)
q7 UNLOAD(M3,DOCK)
q8 EXIT

A portion of the planning steps generated by PLAN-AHEAD is shown in Figure A.1. The resulting schedule and the corresponding machine loading is depicted in Figure A.2. After plan revision, the sechedule is improved to the one shown in Figure A.3.

Insert Figures A.1, A.2, and A.3 Here

References

1. Baker, K. R., Introduction to Sequencing and Scheduling (New York: John Wiley & Sons), 1974.
2. Bourne, D. and Fussell, P., "Designing Programming Languages for Manufacturing Cells," The Robotics Institute, CMU-R1-Tr-82-5, Carnegie-Mellon University, 1982.
3. Brinch Hansen, P., Operating System Principles (Englewood Cliffs, New Jersey: Prentice-Hall), 1973.
4. Chang, Y. L., Sullivan, R. S., and Bagchi, U., "Experimental Investigation of Quasi-Realtime Scheduling in Flexible Manufacturing Systems," Proc. First ORSA/TIMS Conf. on FMS, 1985.
5. Conway, R. W., Maxwell, W. L., and Miller, L. W., Theory of Scheduling (Massachusetts: Addison-Wesley), 1967.
6. Cutkosky, et. al., "Precision Machining Cells within a Manufacturing System," The Robotics Institute, Carnegie-Mellon University, 1983.
7. Fikes, R. E. and Nilson, N. J., "STRIPS: a new approach to the application of theorem proving to problem solving," Artificial Intelligence 2 (3/4), (1971), p. 189-208.
8. Fikes, R. E., Hart, P. E. and Nilson, N. J., "Learning and executing generalized robot plans," Artificial Intelligence, 3(4), (1972), p. 251-288.
9. Fox, M. S., "The Intelligent Management System: An Overview," The Robotics Institute, Carnegie-Mellon University, 1982.
10. Fox, M. S., Allen, B. P., Smith, S. F. and Strohm, G. A., "ISIS: A Constraint-Directed Reasoning Approach to Job Shop Scheduling," CMU-R1-Tr-83-8, The Robotics Institute, Carnegie-Mellon University, 1983.
11. French, S., Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop (New York: John Wiley), 1982.
12. Hutchingson, G. K., "Flexibility is Key to Economic Feasibility to Automating Small Batch Manufacturing," Industrial Engineering, (1984), p. 77-84.
13. Lozana-Perez, T., "Robot Programming," Artificial Intelligence Laboratory, Massachusetts Institute of Technology, A.I. Memo No. 698, 1982.

14. Merchang, M. E., "Production: A Dynamic Challenge," IEEE Spectrum, May 1983, p. 36-39.
15. Moore, J. M., and Wilson, R. C., "A review of simulation research in job shop scheduling," Production and Inventory Management, (1967), 8:1-10.
16. Nilson, N., Principles of Artificial Intelligence (Palo Alto: Tioga), 1980.
17. Rachamadugu, R. R., Myopic Heuristics in Job Shop Scheduling, Ph.D. Thesis, Carnegie-Mellon University, 1982.
18. Sacerdoti, E. D., A Structure for Plans and Behavior (New York: North-Holland), 1977.
19. Shaw, M. J. P. and Whinston, A. B., "Automatic planning and flexible scheduling: A knowledge-based approach," Proceedings International Conference on Automation and Robotics, St. Louis, 1985.
20. Shaw, M. J. P. and Whinston, A. B., "A Knowledge-Based Approach to Planning and Scheduling in Flexible Cells," Technical Report, Department of Business Administration, University of Illinois, 1985.
21. Shaw, M. J. P., The Design of a Distributed Knowledge-Based System for Intelligent Manufacturing Information Systems, Ph.D. Thesis, Purdue University, 1984.
22. Shaw, M. J. P. and Whinston, A. B., "Distributed planning in cellular flexible manufacturing systems," Management Information Research Center, Purdue University, 1983.
23. Stecke, K. E., "Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems," Management Science, Vol. 29, No. 3, March 1983, p. 273-288.
24. Tate, A., "Generating project networks," Proc. International Joint Conference on Artificial Intelligence, 1977.
25. Vere, S., "Planning in time: windows and durations for activities and goals," Pattern Analysis and Machine Intelligence, Vol. PAMI-5, No. 3, (1983), pp. 246-266.
26. Wilkins, D., "Parallelism in planning and problem solving: reasoning about resources," Tech. Note 258, AI Center, SRI, 1982.
27. Wilkins, D. E., "Domain independent planning: representation and plan generation," Artificial Intelligence, 22, (1984), p. 269-301.

List of Figures

| <u>Figures</u> | <u>Captions</u> |
|----------------|--|
| 1 | Examples of Operators. |
| 2 | Search Tree Generated by the Linear Planner. |
| 3 | A Partially Ordered Plan for Two Jobs. |
| 4 | Flow-Chart of PLAN-AHEAD. |
| 5 | Step 4 of Plan-Revision. |
| 6 | Step 5 of Plan-Revision. |
| 7 | Organization of the Planning System. |
| A.1 | A Portion of the Planning Cycles. |
| A.2 | The Schedule Before Plan-Revision. |
| A.3 | The Schedule After Plan-Revision. |

Figure 1

TRANSFER(M, M', PT, t) : Transfer part PT from machine M to machine M' at time t.

Precondition : FINISH-OP(M, OP, PT, t)

DIFFERENT(M, M')

PT-NEXTOP(OP, OP', PT)

MACH-OP(M', OP')

IDLE(M', t)

Add-list : MACH-PT(M', OP', PT, t)

IDLE(M, t)

Delete-list : FINISH-OP(M, OP, PT, t)

IDLE(M', t)

Resource : M'

Duration : 2

NEXTOP(M, OP, OP', PT, t) : Perform operation OP' on part PT following operation OP on the same machine M.

Precondition : FINISH-OP(M, OP, PT, t)

PT-NEXTOP(OP, OP', PT)

MACH-OP(M, OP')

Add-list : MACH-PT(M, OP', PT, t)

Delete-list : FINISH-OP(M, OP, PT, t)

Resource : M

Duration : 0

Figure 2

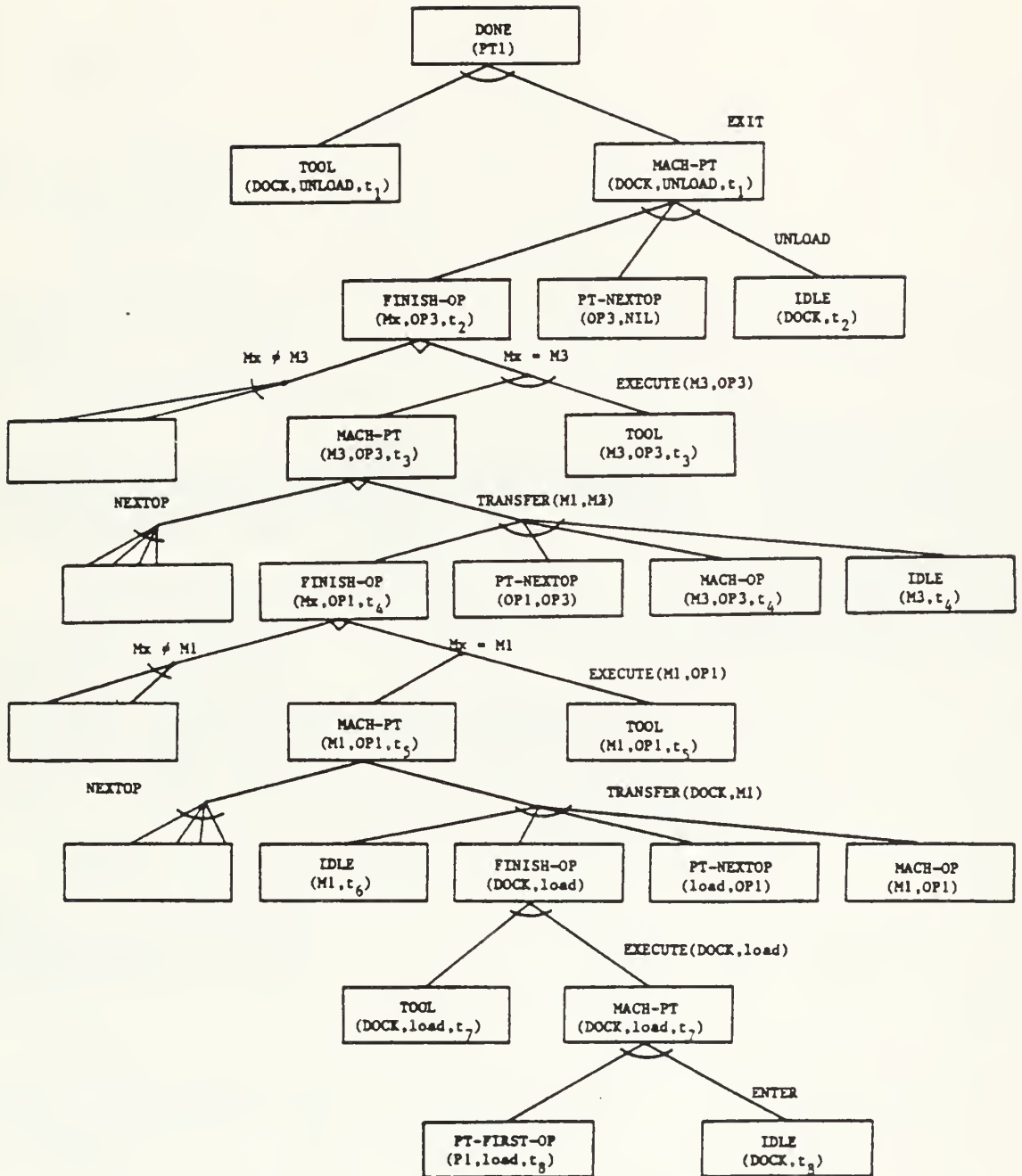


Figure 3

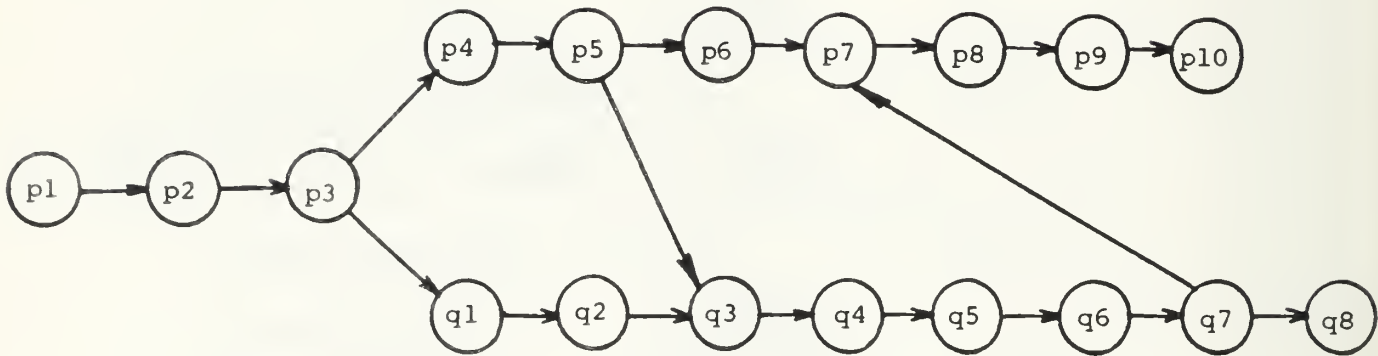


Figure 4

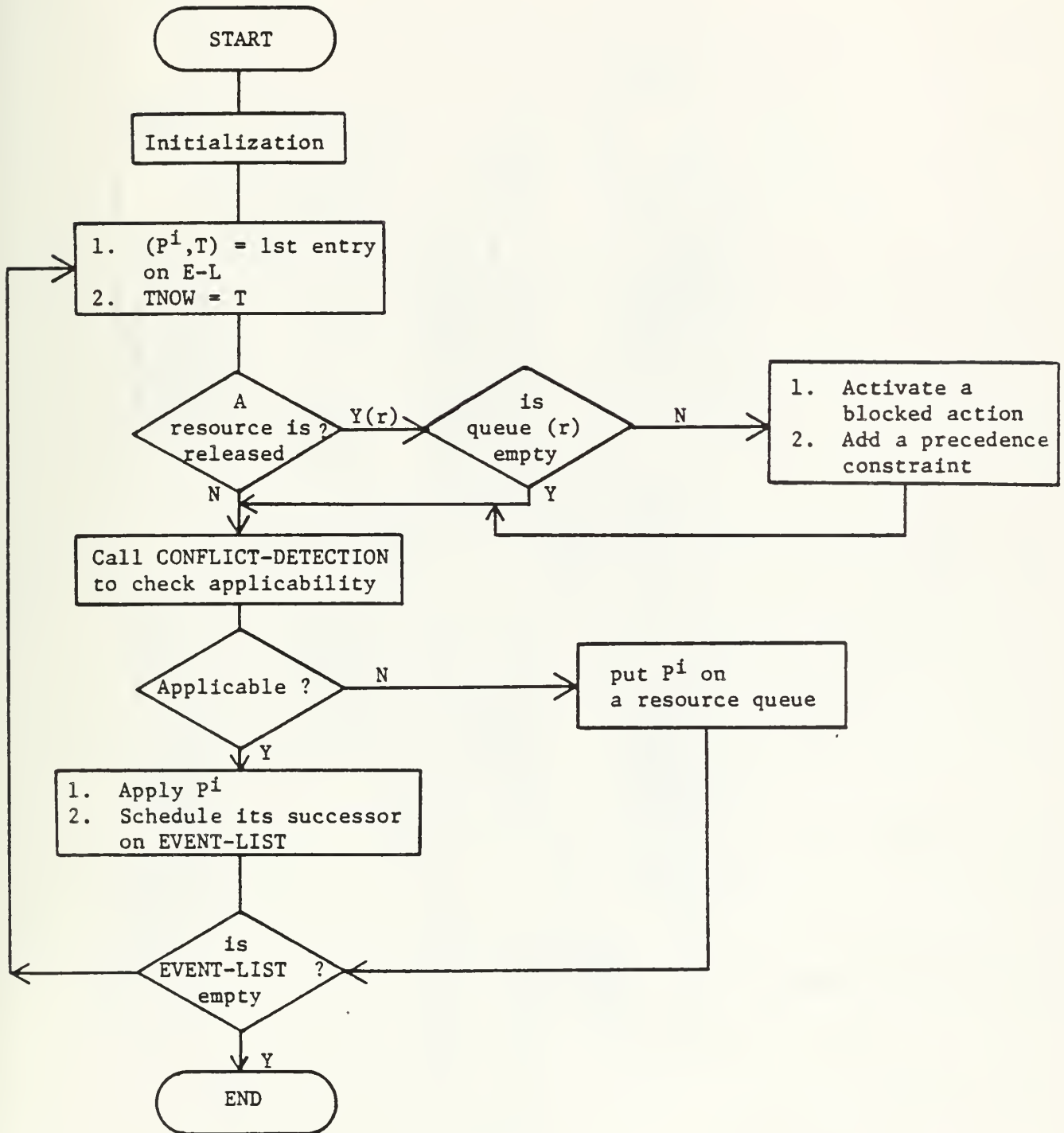


Figure 5

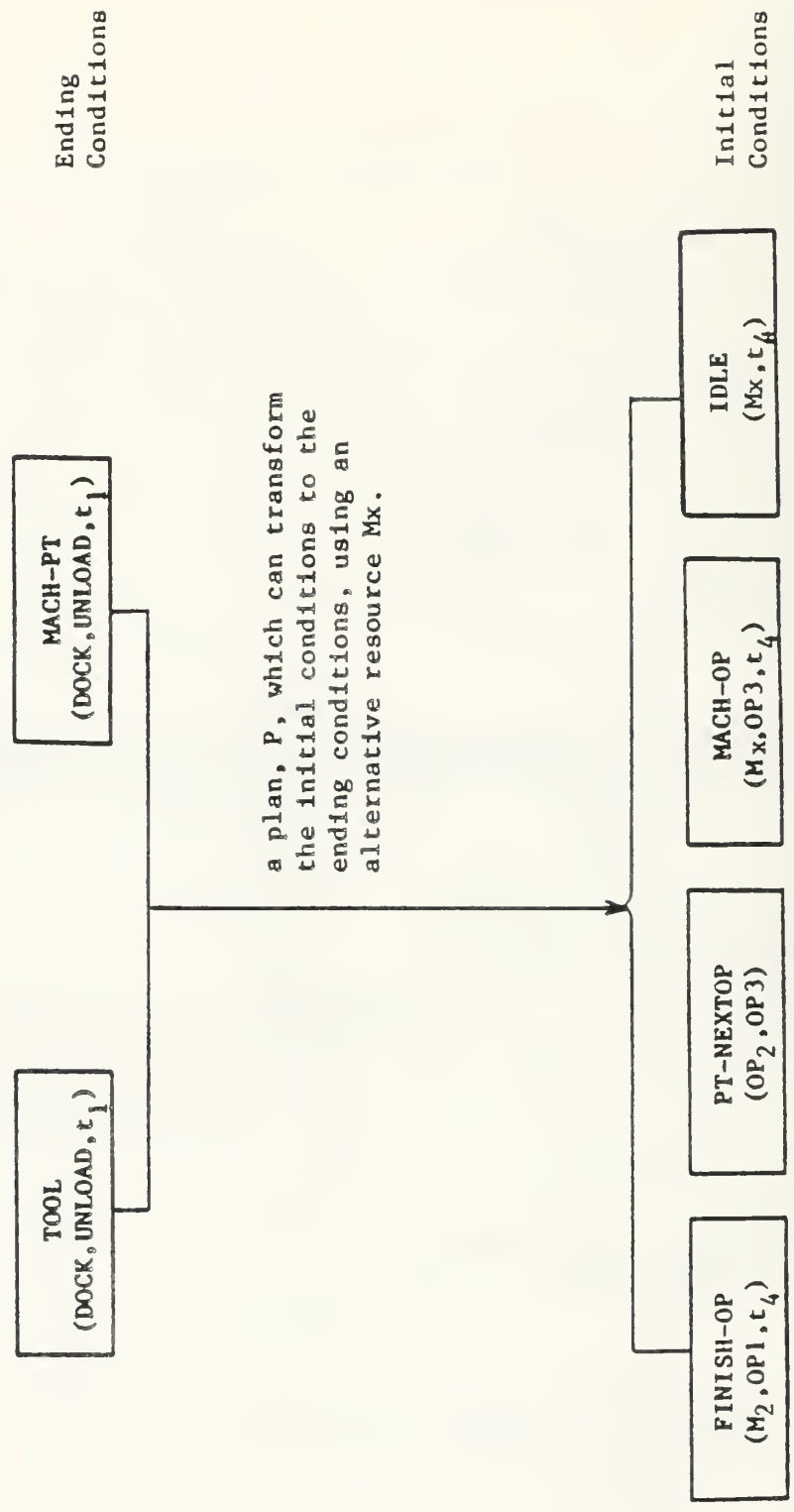


Figure 7

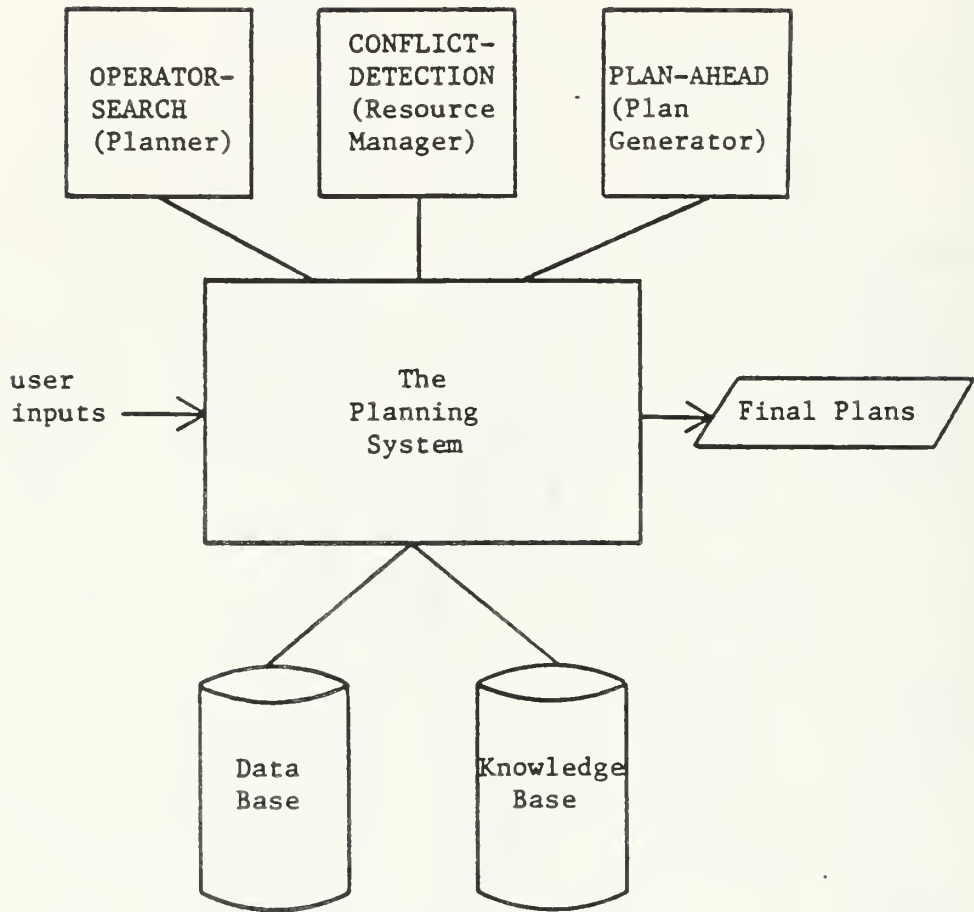
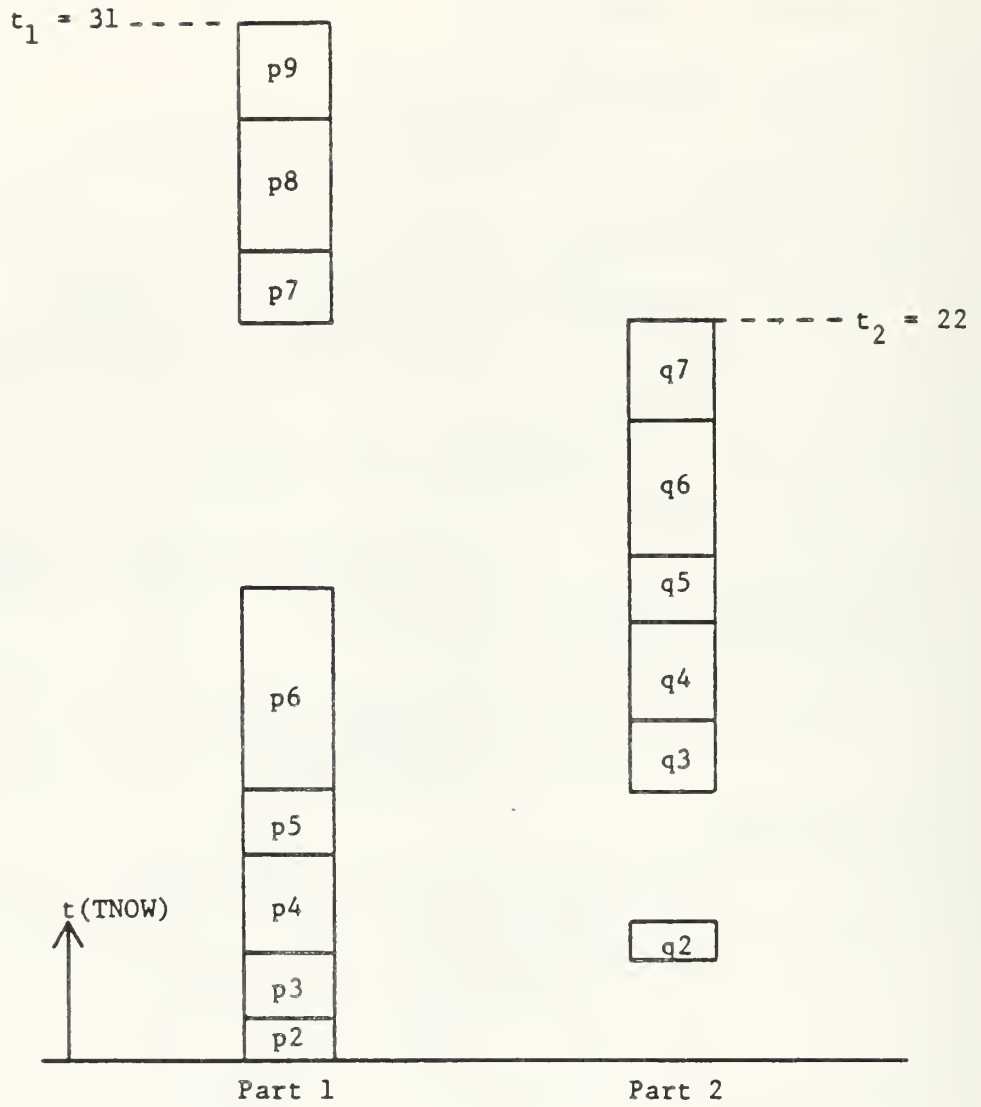


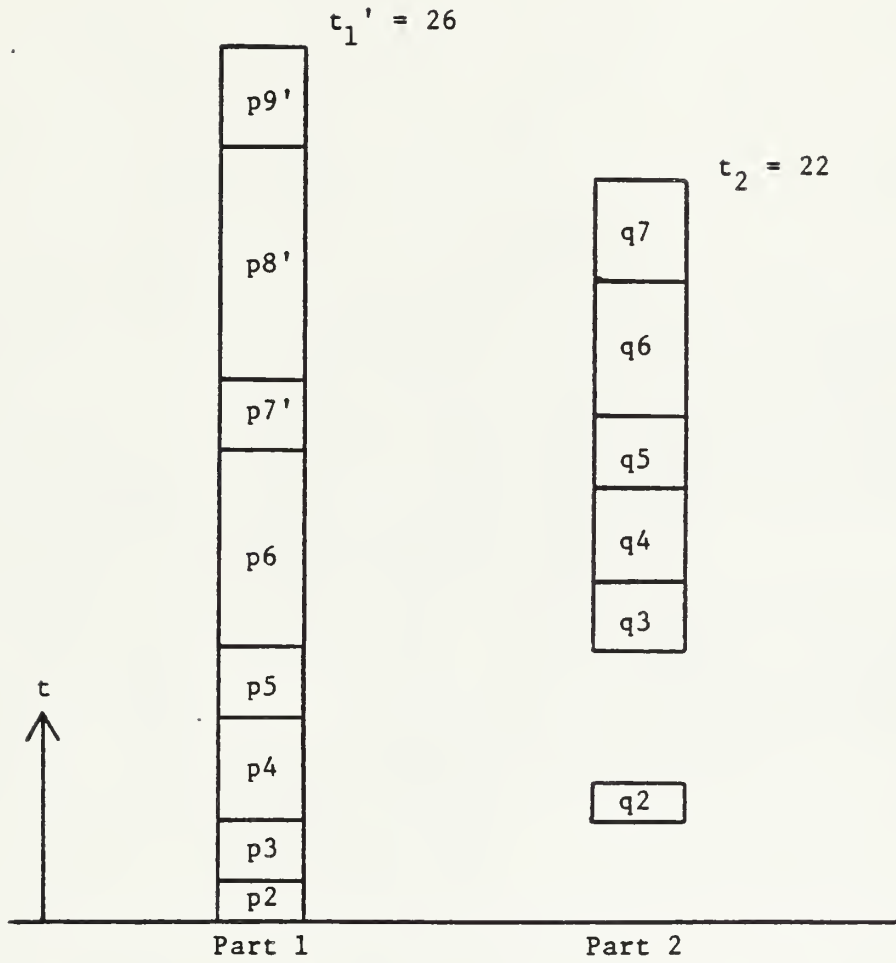
Figure A.2



Total Duration = 31

Average Duration = 26.5

Figure A.3



Total Duration = 26

Average Duration = 24

Table 1

IDLE(M,t): Machine M is idle at time t

MACH-PT(M,OP,PT,t): Machine M begins operation OP on part PT at time t

FINISH-OP(M,OP,PT,t): Machine M completes operation OP on part PT at
time t

DIFFERENT(M,M'): Machine M is a machine different from machine M'

MACH-OP(M,OP): Machine M is capable of performing operation OP

PT-FIRST-OP(OP,PT): Operation OP is the first operation on part PT

PT-NEXTOP(OP,OP',PT): Operation OP' should be performed on part PT
immediately after operation OP

DONE(PT,t): All operations on part PT are completed at time t

TOOL(M,OP,t): The tool for operation OP is available to the machine M
at time t

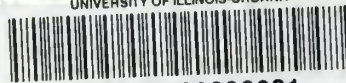
HECKMAN
BINDERY INC.



JUN 95

Bound-To-Please® N. MANCHESTER,
INDIANA 46962

UNIVERSITY OF ILLINOIS-URBANA



3 0112 060296081