Division of Research
Graduate School of Business Administration
The University of Michigan

August 1979

# OPTIMAL METHODS FOR SCHEDULING PROJECTS UNDER RESOURCE CONSTRAINTS

Working Paper No. 191

F. Brian Talbot
University of Michigan
and
James H. Patterson
University of Missouri-Columbia

# OPTIMAL METHODS FOR SCHEDULING PROJECTS

# UNDER RESOURCE CONSTRAINTS

by

F. Brian Talbot
University of Michigan

and

James H. Patterson
University of Missouri-
Columbia

## Abstract

Project network techniques such as PERT and CPM have been widely used in the planning, scheduling, and control of large projects. Refinements of these techniques on the part of practitioners and researchers have made existing procedures easier and less costly to use, and new developments in related computer software and hardware have met the needs of such special project environments as cost control, the desirability of graphics, resource allocation, and so on.

It is the purpose of this paper to discuss briefly two state-of-the-art techniques that have been developed for the optimal scheduling, under conditions of limited resources, of jobs (activities) which cannot be interrupted once they are started. Recent advances in this area have made it possible to obtain schedules that are significantly more cost-effective than those previously obtainable.

## The Impact of Limited Resources

Basic project scheduling methods provide start and finish times for jobs within a project such that precedence relationships only are not violated. For example, the application of the critical path method (CPM) to jobs in the project given in Figure 1 and Table 1 (ignoring crew or resource requirements) results in the early-start schedule illustrated in Figure 2. Here the shaded bars extend from the critical-path-determined early-start time (ES) of each job to the critical-path-determined early-finish time (EF) of each job.

Basic PERT and CPM scheduling techniques such as those illustrated have proven to be very useful in practice, but are often inadequate when resources are limited and when the activities of the project require the simultaneous use of common resources from a resource "pool." Under these conditions, the competition for resources results in scheduling conflicts. Ultimately, the start of certain jobs may be delayed beyond their early-start times until the resources they require are freed by the completion of competing jobs. The process of determining which jobs to schedule and which to delay is termed resource conflict resolution. The manner in which conflicts are resolved influences the length of time required to complete the entire project, or different projects of a project set.

Resource conflict resolution, and the complexity involved in scheduling jobs in this type of environment, can be appreciated by examining the project given in our example, in which six resources are in limited supply. In particular, we will assume that these limited resources are labor crews of various types, as described in Table 2. Other resources needed to complete the project

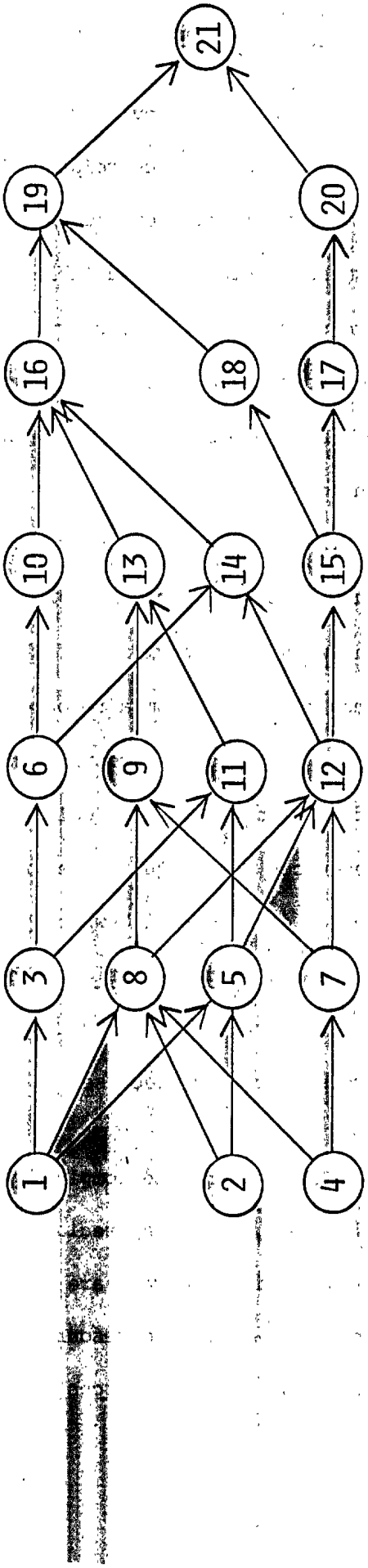



Figure 1

Project Network Diagram

Table 1

Job Durations and Crew Requirements
(Maintenance Project)

| Job Number | Expected Duration (Days) | Crew Requirements Per Day | | | | | |
|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 6 | 5 | 2 | 2 | 2 | 7 | 4 |
| 2 | 3 | 3 | 5 | 2 | 3 | 9 | 6 |
| 3 | 4 | 2 | 4 | 4 | 2 | 3 | 1 |
| 4 | 6 | 5 | 4 | 3 | 5 | 5 | 4 |
| 5 | 7 | 3 | 5 | 2 | 3 | 8 | 0 |
| 6 | 5 | 4 | 1 | 4 | 9 | 2 | 5 |
| 7 | 2 | 4 | 1 | 4 | 3 | 9 | 8 |
| 8 | 2 | 5 | 5 | 4 | 0 | 9 | 1 |
| 9 | 2 | 3 | 2 | 4 | 3 | 4 | 2 |
| 10 | 6 | 1 | 5 | 4 | 6 | 7 | 3 |
| 11 | 1 | 3 | 3 | 2 | 4 | 5 | 1 |
| 12 | 2 | 3 | 2 | 2 | 8 | 3 | 4 |
| 13 | 4 | 2 | 2 | 2 | 2 | 4 | 8 |
| 14 | 2 | 1 | 4 | 4 | 3 | 4 | 1 |
| 15 | 3 | 5 | 5 | 4 | 6 | 2 | 3 |
| 16 | 5 | 3 | 2 | 3 | 4 | 7 | 8 |
| 17 | 8 | 4 | 5 | 4 | 2 | 3 | 4 |
| 18 | 2 | 5 | 3 | 3 | 3 | 7 | 8 |
| 19 | 6 | 2 | 4 | 6 | 2 | 3 | 4 |
| 20 | 2 | 1 | 6 | 2 | 7 | 5 | 2 |
| 21 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

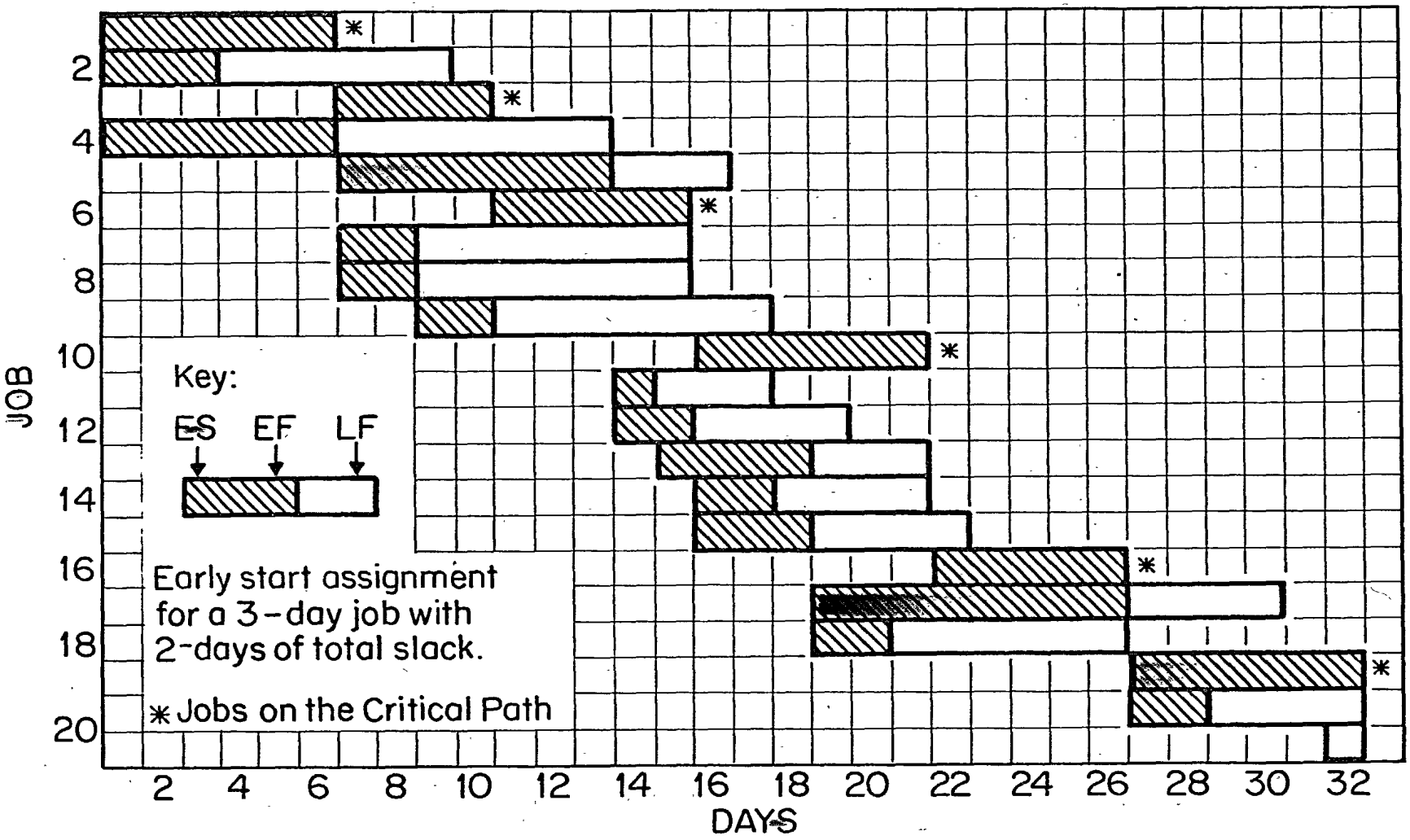



Figure 2
Critical Path Method Early Start Schedule

Table 2

Maximum Number of Crews Available Each Day

| | Crew Type | Number Available Each Day |
|---|---|---|
| 1. | Electrician | 7 |
| 2. | Carpenter | 10 |
| 3. | Sheet Metal | 10 |
| 4. | Plumber | 16 |
| 5. | General Labor | 18 |
| 6. | Structural Steel | 13 |

are assumed to be in unlimited supply, and hence can be ignored at our level of scheduling detail. Each of the twenty jobs requires a specific complement of crews in order to be completed in the expected time period (duration) shown in Table 1.

If one were to attempt to schedule these jobs manually, given the objective of completing the project as soon as possible, he or she would immediately find that jobs 1, 2, and 4 cannot be scheduled simultaneously (in parallel) according to the critical path schedule. Electrician, carpenter, general labor, and structural steel crews are not available in sufficient numbers to schedule these jobs at their early-start times. Hence, the initiation of one or more of these jobs must be delayed until the required crews become available. The question is, which jobs should be delayed? Obviously, we would select to delay that job which would have the least impact on the completion time of the project. But herein lies the difficulty. Given the interrelated nature of jobs in terms of precedence requirements and resource consumption, there is no straightforward method for identifying which jobs to schedule and which to delay. Each job that is scheduled has the potential of affecting the start time of all parallel jobs that use the same types of crews. Furthermore, any job that is delayed has the potential of delaying the starting time of all of its successor activities.

## Heuristic Project Scheduling

In response to the complexity of scheduling in this environment, practitioners have developed computer programs [6] which schedule job assignments using preprogrammed heuristics, or "rules of thumb," to resolve these resource conflicts. One of the more effective of these heuristics [5] is called MINSLK or minimum total slack. Scheduling by this rule is accomplished as follows: when activities "compete" for resources which are not available in sufficient quantities to schedule all activities simultaneously, schedule those activities

with the least amount of slack or total float (as determined by a critical path analysis of the problem). Figure 3 is a flow chart of the procedure used to implement this heuristic decision rule. Note that a simulated clock advances time and, in so doing, considers sorted lists of activities for resource assignment at various stages of schedule completion. The sorted list is comprised of those activities (1) that haven't yet been scheduled and (2) whose predecessors are all completed. The sort is determined by the logic of the heuristic decision rule: activities first on the list are those with the least amount of total float or slack and hence are considered "first" for resource assignment.

Figure 4 is a bar chart of the schedule developed using the MINSLK scheduling heuristic for the Figure 1 project. Table 3 lists the corresponding resource usage for these 49 days. The heuristically determined project completion time of 49 days is 17 days longer (49 - 32) than the critical path completion-time estimate, due to the fact that activities of the project must compete for the six types of labor crews available. The late finish times (LF*) shown in Figure 4 are simply the critical-path-determined LF times extended by 17 days. The finish times of any of the jobs can be delayed up to LF* without extending the project completion time beyond 49 days. In this sense, the notion of total slack, represented by the nonshaded area of the open bars, is the same as it was in Figure 2. Due to limitations on resource availability, however, it may not be possible to find resource-feasible job completions times prior to LF* other than the assignments shown in Figure 4.

## Heuristic versus Optimal Methods

Heuristics such as the MINSLK rule have been widely adopted for scheduling jobs when resources are limited because they rationalize the scheduling process and make it manageable. They have the advantages of being simple to understand,

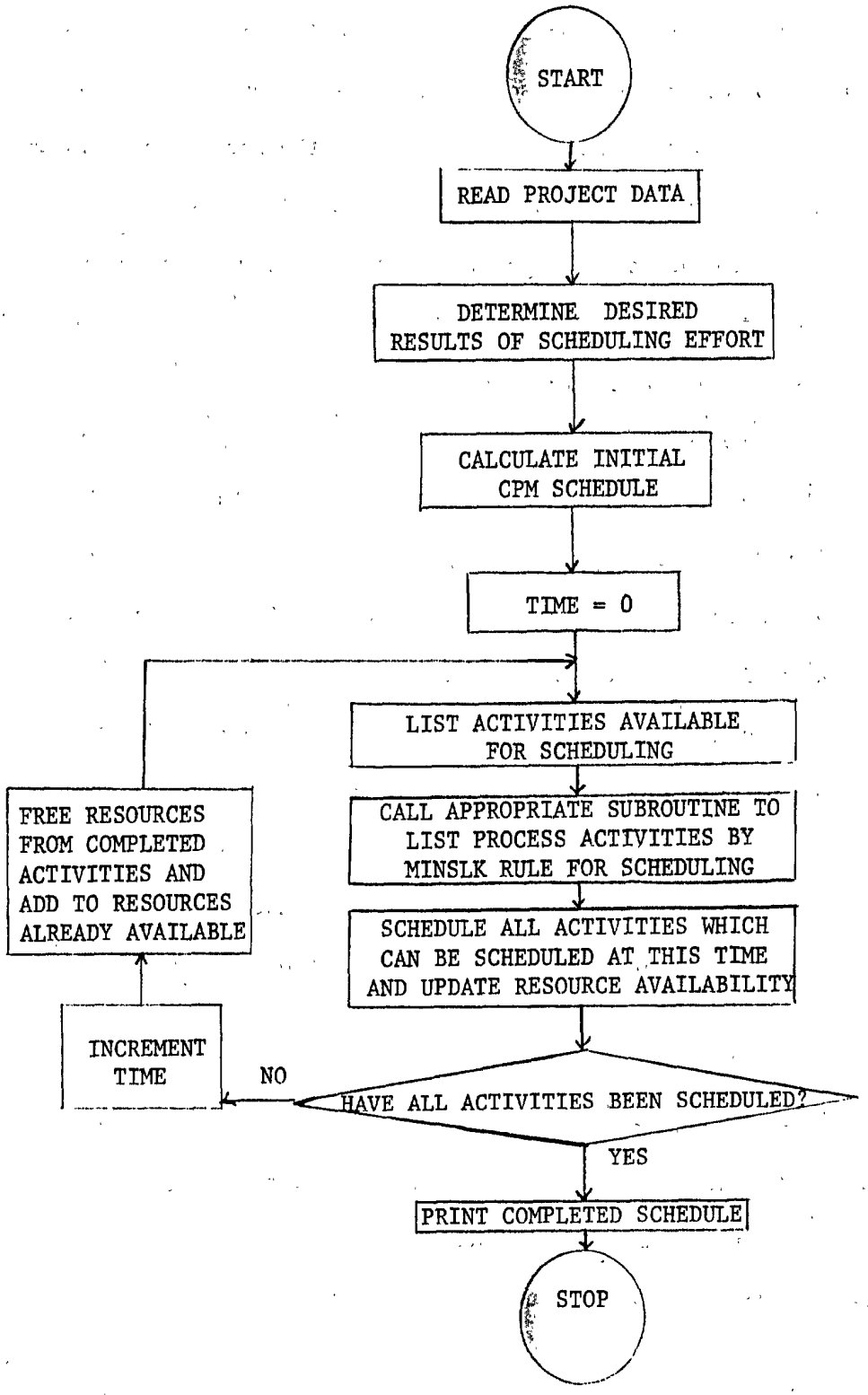



Fig. 3. Flow Chart of Heuristic Scheduling Program using MINSLK Decision Rule.

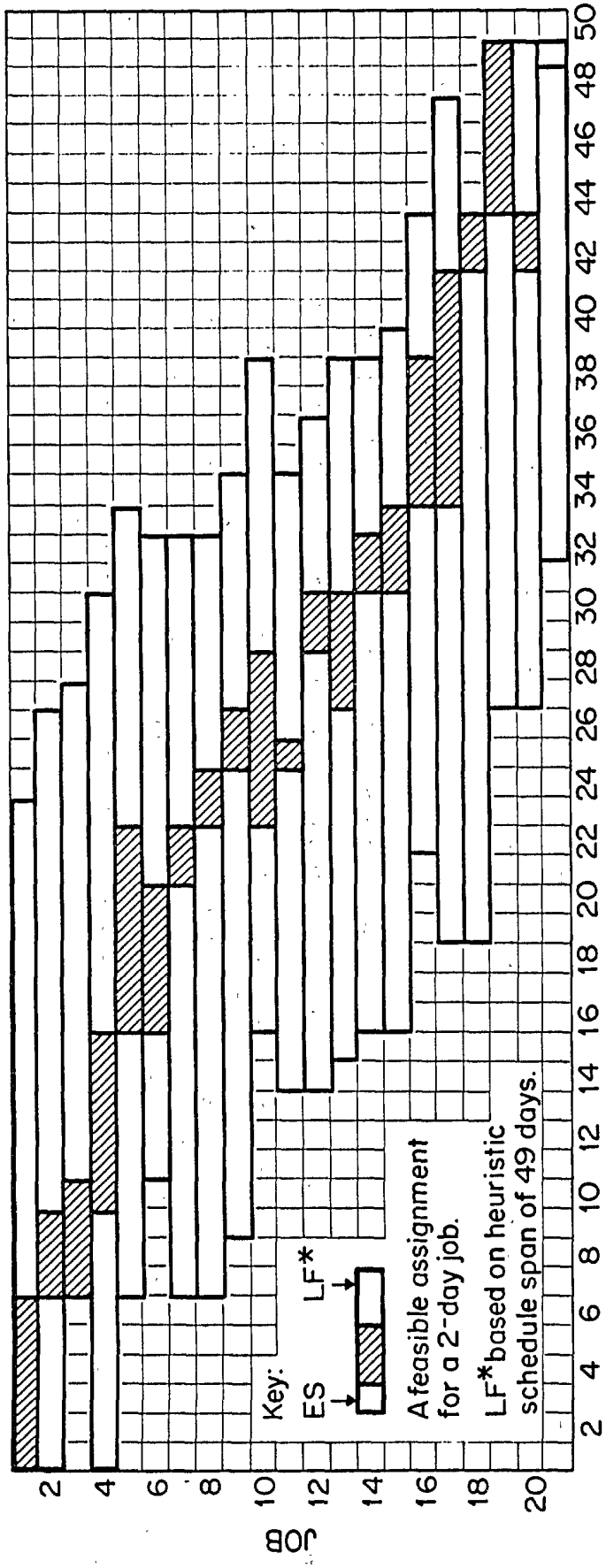
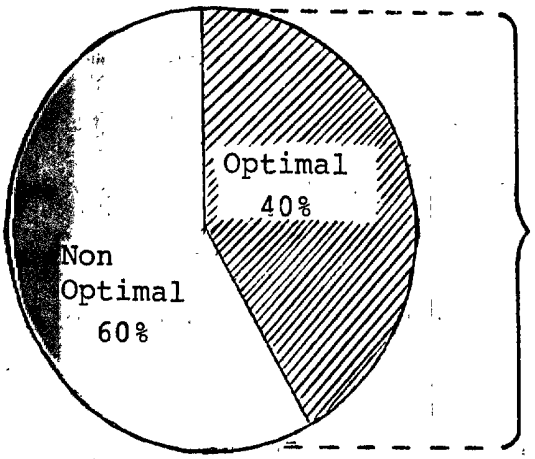



Fig. 4. Forty-nine-day Heuristic Solution to Problem given in Fig. 1.

Table 3

A 49-Day Heuristic Schedule

| Day | Crews Used Each Day | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 5 | 2 | 2 | 2 | 7 | 4 |
| 2 | 5 | 2 | 2 | 2 | 7 | 4 |
| 3 | 5 | 2 | 2 | 2 | 7 | 4 |
| 4 | 5 | 2 | 2 | 2 | 7 | 4 |
| 5 | 5 | 2 | 2 | 2 | 7 | 4 |
| 6 | 5 | 2 | 2 | 2 | 7 | 4 |
| 7 | 5 | 9 | 6 | 5 | 12 | 7 |
| 8 | 5 | 9 | 6 | 5 | 12 | 7 |
| 9 | 5 | 9 | 6 | 5 | 12 | 7 |
| 10 | 7 | 8 | 7 | 7 | 8 | 5 |
| 11 | 5 | 4 | 3 | 5 | 5 | 4 |
| 12 | 5 | 4 | 3 | 5 | 5 | 4 |
| 13 | 5 | 4 | 3 | 5 | 5 | 4 |
| 14 | 5 | 4 | 3 | 5 | 5 | 4 |
| 15 | 5 | 4 | 3 | 5 | 5 | 4 |
| 16 | 7 | 6 | 6 | 12 | 10 | 5 |
| 17 | 7 | 6 | 6 | 12 | 10 | 5 |
| 18 | 7 | 6 | 6 | 12 | 10 | 5 |
| 19 | 7 | 6 | 6 | 12 | 10 | 5 |
| 20 | 7 | 6 | 6 | 12 | 10 | 5 |
| 21 | 7 | 6 | 6 | 6 | 17 | 8 |
| 22 | 7 | 6 | 6 | 6 | 17 | 8 |
| 23 | 6 | 10 | 8 | 6 | 16 | 4 |
| 24 | 6 | 10 | 8 | 6 | 16 | 4 |
| 25 | 7 | 10 | 10 | 13 | 16 | 6 |
| 26 | 4 | 7 | 8 | 9 | 11 | 5 |
| 27 | 3 | 7 | 6 | 8 | 11 | 11 |
| 28 | 3 | 7 | 6 | 8 | 11 | 11 |
| 29 | 5 | 4 | 4 | 10 | 7 | 12 |
| 30 | 5 | 4 | 4 | 10 | 7 | 12 |
| 31 | 6 | 9 | 8 | 9 | 6 | 4 |
| 32 | 6 | 9 | 8 | 9 | 6 | 4 |
| 33 | 5 | 5 | 4 | 6 | 2 | 3 |
| 34 | 7 | 7 | 7 | 6 | 10 | 12 |
| 35 | 7 | 7 | 7 | 6 | 10 | 12 |
| 36 | 7 | 7 | 7 | 6 | 10 | 12 |
| 37 | 7 | 7 | 7 | 6 | 10 | 12 |
| 38 | 7 | 7 | 7 | 6 | 10 | 12 |
| 39 | 4 | 5 | 4 | 2 | 3 | 4 |
| 40 | 4 | 5 | 4 | 2 | 3 | 4 |
| 41 | 4 | 5 | 4 | 2 | 3 | 4 |
| 42 | 6 | 9 | 5 | 10 | 12 | 10 |
| 43 | 6 | 9 | 5 | 10 | 12 | 10 |
| 44 | 2 | 4 | 6 | 2 | 3 | 4 |
| 45 | 2 | 4 | 6 | 2 | 3 | 4 |
| 46 | 2 | 4 | 6 | 2 | 3 | 4 |
| 47 | 2 | 4 | 6 | 2 | 3 | 4 |
| 48 | 2 | 4 | 6 | 2 | 3 | 4 |
| 49 | 2 | 4 | 6 | 2 | 3 | 4 |




| Heuristic | Percentage of Problems for Which Optimum Duration Found* |
|---|---|
| MINSLK | 29% |
| LFT | 20% |
| RSM | 14% |
| GRD | 13% |
| RAN | 5% |
| GRU | 2% |
| MJP | 2% |
| SIO | 1% |

*Percentages computed include problems in which one or more heuristic sequencing rules produced a minimum duration solution.

Source: Davis, E. W., and J. H. Patterson, "Resource Based Project Scheduling: Which Rules Perform Best?", Project Management Quarterly, December, 1975.

Fig. 5. Percent of Problems for which Optimal Minimum Duration Solution Obtained.

easy to apply, and very inexpensive to use in computer programs. Furthermore, research has identified rules which generally provide good solutions [4,5].

The difficulty in relying on heuristic methods, however, is that there is no guarantee that the schedules developed are of minimum duration. Figure 5, taken from [5], shows the results obtained using a set of eight different heuristic procedures to schedule activities of a project. Optimal (minimum duration) solutions were found for only 40 percent of the problems attempted, and the best heuristic procedure, the MINSLK rule described above, found optimal solutions for only 29 percent of the problems attempted. One common response to this problem is to try a series of heuristic procedures and then to select the schedule with minimum duration, an approach which the authors fully endorse. As Figure 5 illustrates, however, while this procedure reduces the uncertainty about the "goodness" of the schedule, it by no means guarantees that the schedule of shortest duration will be found.

Recent advances in the development of optimal solution procedures have given practitioners more powerful tools with which to overcome the weaknesses of these heuristics. The declining cost of computing and the proliferation of minicomputers have removed some of the financial constraints on their use. Techniques now exist which can be used to provide optimal or near-optimal solutions to many industrial and engineering resource-based scheduling problems. We will now examine two of the more promising optimizing techniques and compare the heuristic and the optimal schedules for the project in our example.

## Optimal Procedures

The two optimal procedures we will consider [7,8] seek minimum-duration schedules through the systematical evaluation of all possible job (activity) assignments. Since the number of assignment combinations is large even for

small problems, however, many combinations must be evaluated implicitly rather than explicitly, using rules based on network characteristics.

Stinson's Approach

Joel Stinson's [7] approach to the problem of determining the minimum-duration schedule for a set of activities uses a method that examines subsets of the jobs until all "good" combinations of these subsets have been evaluated. Stinson's procedure systematically generates a "solution's tree" which will ultimately contain the minimum-duration schedule. Nodes on this tree represent partial project schedules, that is, feasible assignments for some of the jobs in the project. Branches extend from a "parent" node to descendent nodes that represent the same "parent" partial schedule appended by the feasible assignment of one or more additional jobs.

Generating such a solutions tree will ultimately result in an exhaustive enumeration of all possible job assignments from which the schedule of shortest duration can be selected. However, not only is such an enumeration impractical due to computer time and storage requirements, it isn't necessary. By identifying partial schedules that dominate others, Stinson is able to prune many nodes from the tree, a procedure which has the effect of implicitly enumerating all completions of the dominated schedules.

Figure 6 shows an example of a solutions tree on which we have indicated possible assignments (partial solutions) for several of the first jobs of our example project. As it relates to artificial intelligence, such a search scheme is termed "breadth-first search"; that is, the procedure attempts to "look 'wide'" for improved solutions to the problem and to identify as early as possible partial schedules that dominate others. Looking at level 1, for example, we may find that, given a limitation on resource (crew) availablity like the one described,

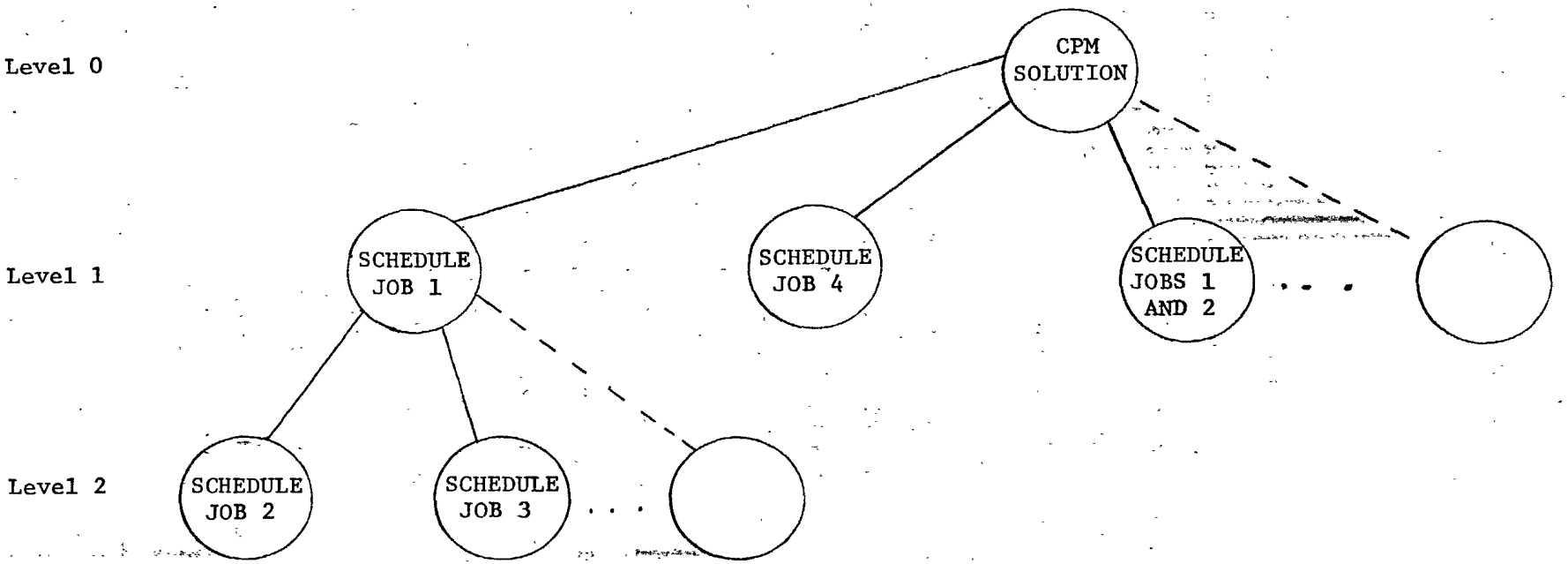



Figure 6

Branch-and-Bound Solutions Tree

the solution in which job 1 is scheduled in the first time period dominates (in the sense of producing a minimum duration solution) the solution in which job 2 or job 4 is scheduled in the first period. In this case, we would have implicitly enumerated all possible solutions in which job 2 or job 4 would be scheduled during the first time period; we need not consider descendent nodes from these parent partial schedules any further. This is the key concept in devising efficient branch-and-bound procedures.

Talbot and Patterson's Approach

Our approach [8] as opposed to the one developed by Stinson, begins by renumbering the jobs of the project such that jobs with the lower numbers possess the least amounts of slack or total float (similar to the MINSLK heuristic rule). Jobs are then considered in numerical order for resource assignment and scheduling. Whenever a job cannot be assigned in such a way as to be completed within LF* determined by a heuristic procedure such as the MINSLK heuristic, we backtrack to a preceding job. An attempt is then made to schedule the preceding job for completion later in its LF* time period, with the provision that other jobs numbered higher on the list can also be scheduled within their current LF* times. As each improved (shorter duration) solution is found, the LF* for each activity is reduced accordingly, and an attempt is made to schedule (shift) all jobs within their revised LF* time periods. If all jobs cannot be assigned within the revised LF* estimate, the minimum duration schedule has been found. It is given by the previous solution in which all jobs were scheduled.

Figure 7 gives a partial solutions tree for this procedure. In the language of artificial intelligence, such a search procedure is termed "depth-first search"; that is, the procedure attempts to find a feasible solution rapidly by transversing down the solutions network, considering each of the jobs in the order given.

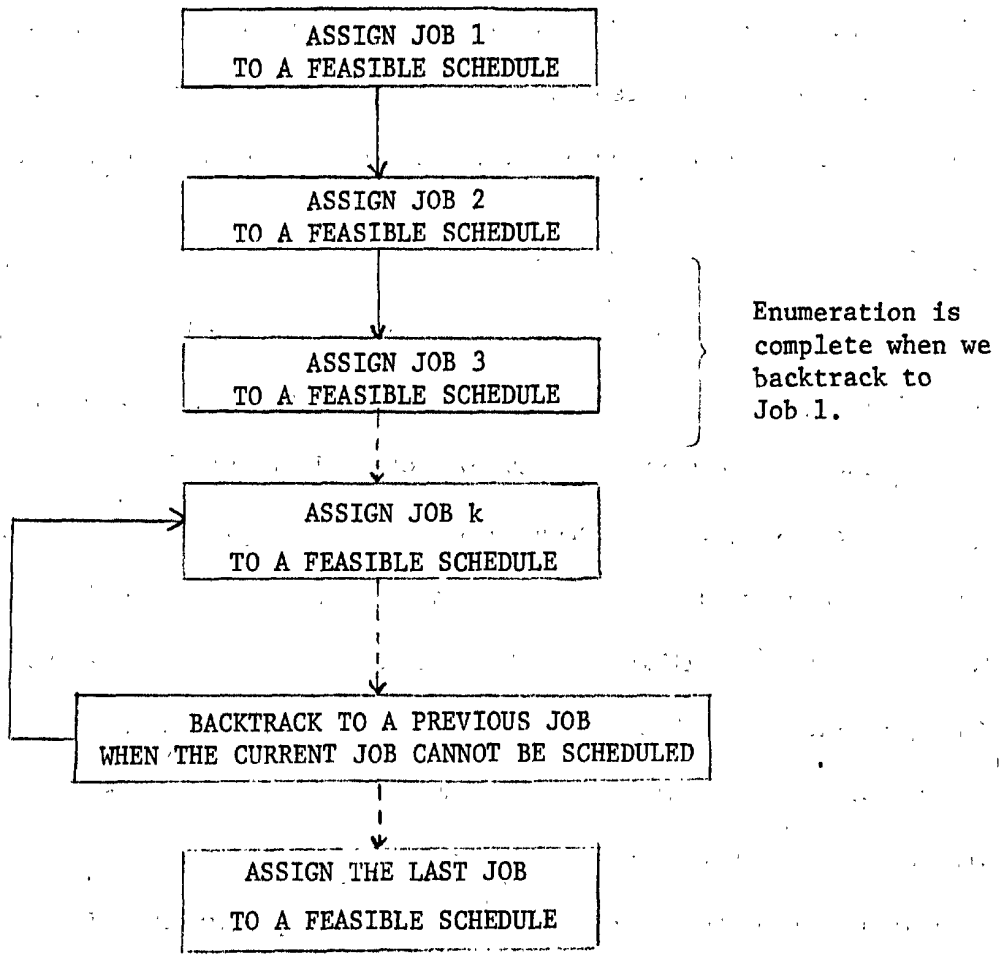



Figure 7

Depth-First Search Solutions Tree

The solution network for our procedure tends to branch out less than a branch-and-bound procedure such as that developed by Stinson. Both techniques, however, use special "pruning" rules to eliminate partial solutions and hence to implicitly enumerate the majority of possible solutions.

Our implicit enumeration approach differs from Stinson's primarily in the manner in which we store schedules that have been evaluated and the way we seek new job assignments. Basically, we store only the current best schedule and the schedule currently under evaluation, whereas Stinson stores the entire (pruned) solution tree. Consequently, our approach requires significantly less computer storage and may be used on a number of the minicomputers available today.

Recent experience indicates, however, that Stinson's procedure seems to require less computer time on the average for larger projects than does ours. Since both techniques can produce systematically improved heuristic schedules as well as optimal schedules, the trade-off facing a potential user thus involves computer time versus computer storage. Where large computers are available to a project manager, Stinson's procedure is likely to be preferred; where only small or minicomputers are available, our technique is likely to be favored.

Figure 8 and Table 4 give the optimal solution found by applying our approach to the project given in Figure 1 and Table 1. The project completion time of 43 days has the obvious advantage over the heuristic solution of freeing resources 6 days earlier for use in other projects. Additional potential advantages would be related to the good-will or revenue (bonuses) gained and the costs or penalties avoided by finishing a project earlier than originally scheduled.

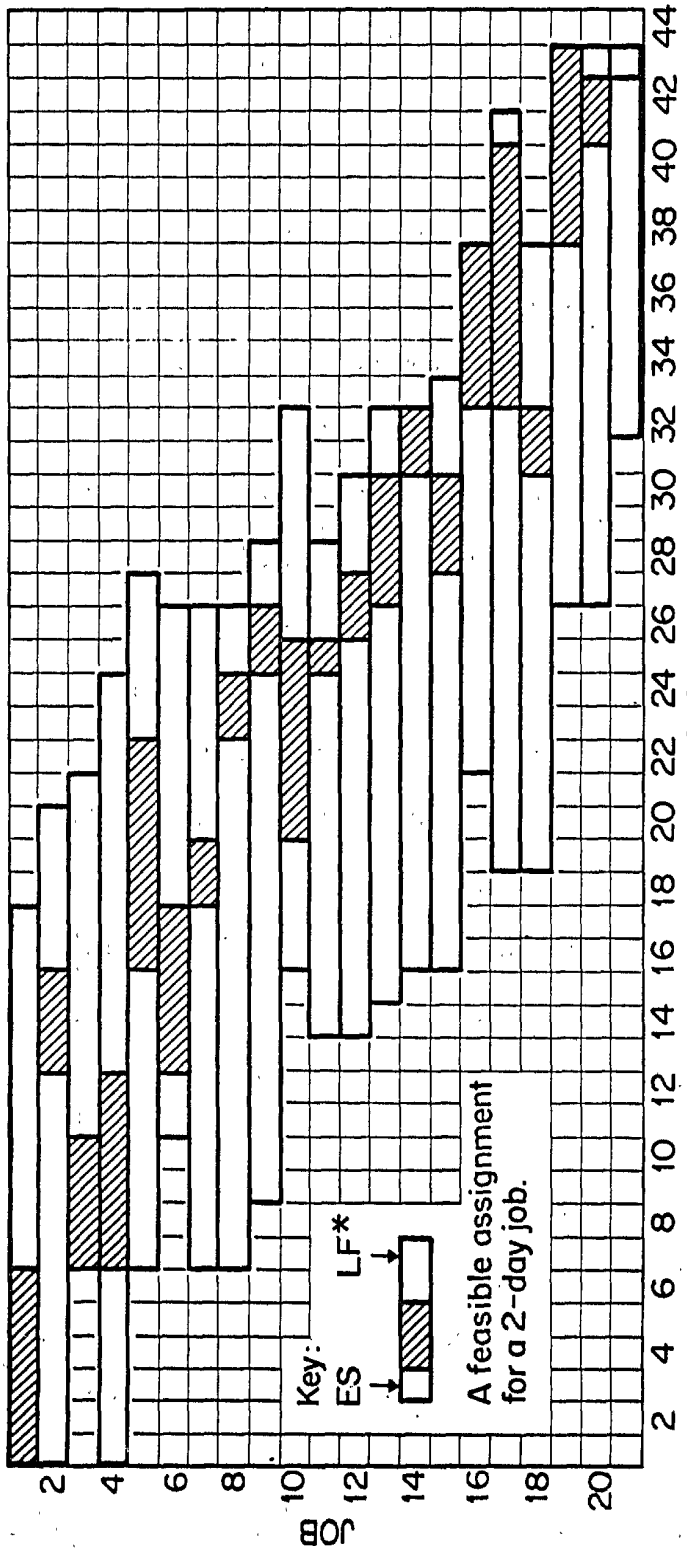



Fig. 8. Forty-three-day Optimal Schedule

Table 4

A 43-Day Optimal Schedule

| Day | Crews Used Each Day | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 5 | 2 | 2 | 2 | 7 | 4 |
| 2 | 5 | 2 | 2 | 2 | 7 | 4 |
| 3 | 5 | 2 | 2 | 2 | 7 | 4 |
| 4 | 5 | 2 | 2 | 2 | 7 | 4 |
| 5 | 5 | 2 | 2 | 2 | 7 | 4 |
| 6 | 5 | 2 | 2 | 2 | 7 | 4 |
| 7 | 7 | 8 | 7 | 7 | 8 | 5 |
| 8 | 7 | 8 | 7 | 7 | 8 | 5 |
| 9 | 7 | 8 | 7 | 7 | 8 | 5 |
| 10 | 7 | 8 | 7 | 7 | 8 | 5 |
| 11 | 5 | 4 | 3 | 5 | 5 | 4 |
| 12 | 5 | 4 | 3 | 5 | 5 | 4 |
| 13 | 7 | 6 | 6 | 12 | 11 | 11 |
| 14 | 7 | 6 | 6 | 12 | 11 | 11 |
| 15 | 7 | 6 | 6 | 12 | 11 | 11 |
| 16 | 7 | 6 | 6 | 12 | 10 | 5 |
| 17 | 7 | 6 | 6 | 12 | 10 | 5 |
| 18 | 7 | 6 | 6 | 6 | 17 | 8 |
| 19 | 7 | 6 | 6 | 6 | 17 | 8 |
| 20 | 4 | 10 | 6 | 9 | 15 | 3 |
| 21 | 4 | 10 | 6 | 9 | 15 | 3 |
| 22 | 4 | 10 | 6 | 9 | 15 | 3 |
| 23 | 6 | 10 | 8 | 6 | 16 | 4 |
| 24 | 6 | 10 | 8 | 6 | 16 | 4 |
| 25 | 7 | 10 | 10 | 13 | 16 | 6 |
| 26 | 6 | 4 | 6 | 11 | 7 | 6 |
| 27 | 5 | 4 | 4 | 10 | 7 | 12 |
| 28 | 7 | 7 | 6 | 8 | 6 | 11 |
| 29 | 7 | 7 | 6 | 8 | 6 | 11 |
| 30 | 7 | 7 | 6 | 8 | 6 | 11 |
| 31 | 6 | 7 | 7 | 6 | 11 | 9 |
| 32 | 6 | 7 | 7 | 6 | 11 | 9 |
| 33 | 7 | 7 | 7 | 6 | 10 | 12 |
| 34 | 7 | 7 | 7 | 6 | 10 | 12 |
| 35 | 7 | 7 | 7 | 6 | 10 | 12 |
| 36 | 7 | 7 | 7 | 6 | 10 | 12 |
| 37 | 7 | 7 | 7 | 6 | 10 | 12 |
| 38 | 6 | 9 | 10 | 4 | 6 | 8 |
| 39 | 6 | 9 | 10 | 4 | 6 | 8 |
| 40 | 6 | 9 | 10 | 4 | 6 | 8 |
| 41 | 3 | 10 | 8 | 9 | 8 | 6 |
| 42 | 3 | 10 | 8 | 9 | 8 | 6 |
| 43 | 2 | 4 | 6 | 2 | 3 | 4 |

Table 5 gives an indication of the relative cost involved in using an optimal-seeking technique. The heuristic solution of 49 days incurred about $.62 in related computer costs. Improved solutions of 46 and 45 days were obtained for an additional $.02 each before the optimal solution of 43 days was found, at an additional cost of $2.58. Thus, the total estimated cost of obtaining a minimum-duration schedule for the example project is $3.23. The optimal schedule obtained is six days shorter than the heruistically determined schedule, at an additional computation cost of only $2.61. Hence the benefits of optimal project scheduling in this case would likely far exceed the costs.

Unfortunately, methods have not yet been developed for efficient optimal solving of all project scheduling problems. As a rule of thumb, we have found that projects containing more than 50 jobs and six resource types cannot often be optimally solved in a cost-effective manner. However, projects containing more than 50 jobs can benefit from the application of the optimal procedures described. Both our approach and Stinson's can be stopped prematurely when the computational cost of continuing becomes unattractive. A good, if perhaps suboptimal, solution is always available. For example, Table 5 indicates that if computation had been terminated at 1.0 seconds of computer time, an improved solution of 45 days would have been obtained.

## Summary

Recent advances in techniques of integer programming known as branch-and-bound and implicit enumeration have made available to the project manager tools for optimally scheduling the activities of a project where a limit on resources causes some activities to be delayed. These procedures, when combined with the greater speeds of current computers, are able to aid the project

Table 5

Cost of Improved Solutions

| Project Duration (Days) | Time to Obtain (Seconds)[a] | Computer Cost[b] |
|---|---|---|
| 49 | .009 | $ .62 |
| 46 | .014 | .63 |
| 45 | .033 | .65 |
| 43 | 5.163 | 3.23 |

[a] Cumulative CPU time on an Amdahl V/6 computer. Includes time for some printed output.

[b] This is an estimate of the cumulative total cost at commercial rates running our program on The University of Michigan's Amdahl V/6 computer in February 1979.

manager in allocating scarce resources to the competing activities of a project such that the allocation given is the optimal one for the criterion of completing the project in as short a time as possible. The costs of obtaining these optimal solutions have witnessed a dramatic decrease in recent years, especially when compared to the cost savings which can result from a better assignment of scarce resources.

## References


1. Davis, E.W., "An Exact Algorithm for the Multiple Constrained-Resource Project Scheduling Problem," Ph.D. Dissertation, Department of Administrative Sciences, Yale University, 1968.

2. Davis, E.W., "Project Scheduling Under Resource Constraints - Historical Review and Categorization of Procedures," AIIE Transactions, December, 1973.

3. Davis, E.W., and G.E. Heidorn, "Optimal Project Scheduling Under Multiple Resource Constraints," Management Science, August, 1971.

4. Davis, E.W., and J.H. Patterson, "A Comparison of Optimal and Heuristic Solutions in Resource-Constrained Project Scheduling," Management Science, April, 1975.

5. Davis, E.W., and J.H. Patterson, "Resource-based Project Scheduling: Which Rules Perform Best?," Project Management Quarterly, December, 1975.

6. Jenett, E., "Availability of CPM Programs," Project Management Quarterly, July, 1970.

7. Stinson, J.P., "A Branch and Bound Algorithm for a General Class of Resource-Constrained Scheduling Problems," AIIE Conference Proceedings, Las Vegas, Nevada, 1975.

8. Talbot, F.B., and J.H. Patterson, "An Efficient Integer Programming Algorithm with Network Cuts for Solving Resource-Constrained Scheduling Problems," Management Science, July, 1978.