1972

A Heuristic Routine for Project-Network Scheduling with Resource Constraints

Hector J. Alvarez
University of Rhode Island

A HEURISTIC ROUTINE FOR PROJECT-NETWORK SCHEDULING WITH RESOURCE CONSTRAINTS

BY

HECTOR J. ALVAREZ

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
INDUSTRIAL ENGINEERING

UNIVERSITY OF RHODE ISLAND

1972

MASTER OF SCIENCE THESIS

OF

HECTOR J. ALVAREZ

Approved:

Thesis Committee:

Chairman David M. Shao

R Chaudary Hanumara

Charles F. James Jr

Aloys A. Michel

Dean of the Graduate School

UNIVERSITY OF RHODE ISLAND

1972

ABSTRACT

A heuristic scheduling routine was developed in this study for scheduling the activities of a project-network. The objectives of the scheduling process are to minimize the daily allocation of resources, satisfy constraints on the availability of resources, and achieve total completion of the project within a given due date.

Since this type of a problem is a large combinatorial one, an analytical solution is almost impossible, and always impractical, even for small project-networks.

The scheduling procedure developed in this study consists of a series of computations and heuristic decisions based on functional properties of the networks and assembled into a logical sequence of steps designed to originate successive schedules that converge toward the optimal or near-optimal solution.

The heuristic scheduling routine was tested with several artificially prepared project-networks for which the optimal solutions were already known. Either the originally prepared optimal schedules or resource equivalent ones were obtained.

The solution of this problem will find real-life applications in the maintenance functions of varying industrial organizations, and in the construction industry among others.

ACKNOWLEDGMENTS

The author wishes to express his sincere gratitude to Dr. David Shao, professor of Industrial Engineering, for his guidance, valuable suggestions and constant encouragement throughout the entire preparation of this thesis.

The author is indebted to Dr. Charles F. James Jr., chairman of the Industrial Engineering Department, for his friendship, encouragement, and valuable review of the manuscripts.

The author expresses his appreciation to Dr. Choudary Hanumara, professor of Statistics and Computer Science, for his constructive criticism of this thesis.

Last but not least, the author is grateful to all members of the staff of the Industrial Engineering Department of URI and fellow students, who made me feel at home among them.

DEDICADO CON TODO CARIÑO A MIS PADRES:

Sr. Héctor Guillermo Alvarez

y

Sra. Ana Herrera de Alvarez

TABLE OF CONTENTS

Page

ABSTRACT ii

ACKNOWLEDGMENTS iii

DEDICATION iv

TABLE OF CONTENTS v

LIST OF TABLES vi

LIST OF FIGURES vii

Chapter

I. INTRODUCTION 1

II. LITERATURE REVIEW 13

III. SINGLE-PROJECT SINGLE-RESOURCE SCHEDULING 30

IV. GENERALIZED PROJECT SCHEDULING PROBLEM 81

V. CONCLUSIONS 89

VI. REFERENCES 92

APPENDICES 96

A. COMPUTER PROGRAM OF THE SCHEDULING ROUTINE

B. COMPUTER SOLUTION OF THE EXAMPLE PROJECT

C. COMPUTER SOLUTION OF THE CONSTRUCTION OF A GAS STATION

LIST OF TABLES

Number | | Page
I DATA OBTAINED IN THE PLANNING STEP 53

II NETWORK CHARACTERISTICS (ACTIVITIES) 57

III NETWORK CHARACTERISTICS (EVENTS AND PROJECT) 59

LIST OF FIGURES

Figure Page

1. RESOURCE LEVELING PROBLEM. 16

2. RESOURCE ALLOCATION PROBLEM 21

3. ILLUSTRATION OF HU'S METHOD (when M = 3) 29

4. FLOW CHART OF THE SCHEDULING ROUTINE 49

5. EXAMPLE PROJECT-NETWORK 52

6. INITIAL SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT. . . 62

7. SECOND SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT . . . 64

8. THIRD SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT . . . 66

9. FOURTH SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT . . . 68

10. FIFTH SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT . . . 70

11. SIXTH SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT . . . 72

12. SEVENTH SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT. . . 73

13. HEURISTIC BEST SOLUTION SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT. 75

I. INTRODUCTION

In 1957 a team of engineers and mathematicians from Du Pont and Sperry Rand Corporation developed a planning, scheduling, and control technique that became known as the Critical Path Method (CPM). At about the same time, the U.S. Navy Special Projects Office, working with a firm of management consultants and with Lockheed, developed a management control system that was used successfully for coordinating the work of some 3,000 contractors, suppliers, and government agencies involved in design, development, and fabrication of Polaris missiles under the Navy's Fleet Ballistic Missiles Program; this control system was named PERT (Program Evaluation and Review Technique) and proved to be so successful that the Navy credited it for advancing the completion of the Polaris program by more than two years.

Originally, PERT's methodology was probabilistic due to the uncertainty of the time estimates for the duration of the activities involved in the Polaris program. On the other hand, CPM's methodology was deterministic, which is understandable since CPM was developed in an environment dominated by construction engineering and maintenance activities where time estimates are fairly well defined by experience. As actually practiced today, however, either methodology can use the probabilistic as well as the deterministic model, and neither one seems to be overwhelmingly superior to the other.

PERT and CPM are widely used project-network planning and scheduling techniques. Both make use of the fundamental approach of dividing the project into two kinds of basic elements: activities or time consuming elements, and events or time points defined as the start or the end of an activity. The project is represented graphically by a "network" of arrows and circles showing the time-precedence relations of the activities and events. The "critical path" is found by special calculations depending only on the time-precedence relations of the activities and events and is defined as the longest time consuming chain of activities and events connecting the start event and the end event of a given project-network. The length of the critical path also represents the minimum time needed to complete the project; e.g.: the "earliest project completion time" is determined by the critical path. The problem of scheduling a project is actually concerned with the determination of the starting date and the finish date of each activity in the project. Since the earliest starting time, earliest finish time, latest starting time, latest finish time, activity slack time, and other related information are obtained through PERT-CPM computations, it is fairly easy to obtain an initial schedule of the project, simply by scheduling the activities without disrupting the time-precedence relations embedded in the network.

PERT-CPM techniques have proved to be very useful when the major concern is on the variable time, and when it can be assumed that there is an infinite availability of resources and an open budget. However, these assumptions are not always valid.

From the very beginning of the development of PERT-CPM techniques, there have been suggestions to extend the general utility of network planning and scheduling by including other variables besides time. The major concerns were with the problems of time-cost tradeoff, and of scheduling with limited resources. Project time-cost tradeoff[10,26] is the problem of determining the cost of reducing the project completion time by "crashing" a selected combination of activities. Crashing an activity means reducing its duration, usually accomplished by allocating more resources to this activity.

Project scheduling with limited resources comprises two problems: the first one consists of the allocation of resources to the activities, on a day-by-day basis, up to the limit of available resources, trying to find the earliest project completion time that still satisfies the resource constraints; the second problem is finding a schedule that minimizes the daily resource allocations and at the same time completes the project within the given due date. This second problem is most commonly known as the "resource leveling problem" of project scheduling with limited resources.

Insofar as is known, no generalized analytic technique has proved to be successful in solving a generalized problem of project-network scheduling with limited resources, and only "heuristic" techniques applied to varying situations are widely used in practice. A heuristic is a guide or a method of reducing search in a problem solving situation; the phrase "rule of thumb" is often used synonymously with "heuristic".

The following definitions and/or assumptions are given to form a basis for the statement and formulation of the problem of leveling resource allocations when scheduling large project-networks. However, more definitions and/or assumptions might be given later on, as a part of the text, when deemed necessary.

1. The project is a "one time project-network with limited resources", which means that:
   1) once the project is started it will be continued without interruption until its total completion is achieved,
   2) the project can be represented by a network showing the time-precedence relations of its activities and events,
   3) the project may require several kinds of resources, but at least one kind is subjected to availability constraints,
   4) there is no exchange or sharing of allocated resources with another project being executed during the same period of time by the same company or by an affiliate company.
2. The variable <u>time</u> (t) is considered to be an <u>integer variable</u> expressed in days (time-units).
3. "Resources requirements" of an activity (or project) stand for the daily amounts of resources that are needed for its successful completion, "resources availability" refers to the pools of available resources from which the resources requirements can be satisfied, and "resources allocations" refer to the amount of resources drawn from the pools of available resources and assigned to scheduled activities on a day-by-day basis.
4. Each activity behaves like a complete and separate entity within

the network, that is:

1) once the activity is started it has to be continued until its completion is achieved,

2) any activity requires the same amount of resources during any day of its entire duration,

3) resource requirements of one activity may or may not be equal to the resource requirements of another activity.

4) amounts of resources allocated to one activity cannot be allocated to another activity during the same day.

5. The duration and daily resource requirements of all activities in the network have been calculated through time-cost tradeoff computations and can be considered as fixed values.

6. The total amount of resources allocated in any day is equal to the summation of the individual resource requirements of all the activities scheduled on the same day.

Statement of the Problem

The problem to be solved in this thesis is the following:

> "To find a schedule for the activities of a project-network that minimizes the daily allocations of resources, satisfying stated constraints on the availability of resources, and achieving total completion of the project within a given due date."

The problem as stated above represents a restricted version of the problem of project scheduling with limited resources; restricted because we are interested mainly in minimizing peak resource allocations within a given project due date (e.g., resource leveling problem), even if it means to idle some of the available resources; this is what makes our problem different from an allocation problem where the main interest is to obtain the earliest project completion by allocating all available resources to suitable subsets of activities, in a day by day basis, until all activities are scheduled.

Some heuristic techniques have been developed for the solution of related problems* which could give a solution to our problem, but the last word is not yet in on project scheduling with limited resources, and continued research leading to the development of new approaches will help to measure the worth of the various scheduling rules that have been suggested. We are safe in saying that new heuristic techniques in this area will be, in fact, valuable for the final development of an analytical solution in the future.

Mathematical Formulation of the Problem

First we define the required variables and parameters, then we give the formulation of the objective function, and finally we present the integer programming formulation of our problem.

*These problems are discussed at the beginning of Chapter III. The techniques available in the open literature, and which are interesting to us, are also presented in Chapter III.

Definition of Variables and Parameters

1. $t = 1,2,3,\ldots,t_n$ integer variable representing days of schedule; where t_n is the project completion day, e.g., the n^{th} day of schedule.

2. t_d is a parameter representing the fixed project due date.

3. $j = 1,2,3,\ldots,a$ integer variable representing activities' identification numbers (AIN's), where a is the total number of activities in the network.

4. $i = 1,2,3,\ldots,g$ integer variable representing the g kinds of resources required by the activities.

5. r_{ij} represents the requirements for resource i of activity j; it is expressed in units of resource per day.

6. d_j represents the duration of activity j expressed in days.

7. X_{jt} is an integer variable that is either one or zero:

   $X_{jt} = 1$ if the activity j is scheduled during day t;

   $X_{jt} = 0$ if the activity j is not scheduled during day t.

8. Q_{it} is an integer variable representing the total amount of resources i allocated during the day t of the schedule.

9. K_i is a parameter representing the fixed amount of resources i available to any combination of activities during any day of the schedule.

10. Since the resource requirements (r_{ij}) of any activity have to be satisfied by the resource allocation on the schedule, it will always be true that:

$$Q_{it} = \sum_{j=1}^{a} r_{ij} X_{jt}$$

for $t = 1,2,3,\ldots,t_n$

11. Q_i^* represents the largest current peak allocation of resource i on the schedule; that is:

$$Q_i^* \geq Q_{it}$$

for any day t.

Formulation of the Objective Function

The objective of our scheduling problem is to minimize the daily allocations of resources, satisfying the stated availability constraints (K_i for $i = 1,2,3,\ldots,g$), and achieving total completion of the project within the given due date (t_d). This objective will be accomplished by minimizing the following objective function (O.F.):

$$\text{O.F.} = Q_i^* = (\text{maximum} \sum_{j=1}^{a} r_{ij} X_{jt})_i$$

where:

$$t = 1,2,3,\ldots,t_n$$

$$t_n \leq t_d$$

$$i = 1,2,3,\ldots,g$$

$$Q_i^* \leq K_i$$

Integer Programming Formulation of our Scheduling Problem

The integer programming formulation presented next would provide an analytic solution to the problem of project scheduling with limited resources. We are using here an approach similar to that of Bowman[5] for the job shop problem.

Integer-Linear Programming formulation:

$$\text{minimize } \left(\text{maximum } \sum_{j=1}^{a} r_{ij}\, X_{jt}\right)_i$$

for all and every $i = 1,2,3,\ldots,g$

in the range: $t = 1,2,3,\ldots,t_n$

subject to the following constraints:

1) time constraint; project due date is not exceeded.

$$t_n \leq t_d$$

2) resource constraints; resources availabilities are not exceeded.

$$\sum_{j=1}^{a} r_{ij}\, X_{jt} \leq K_i$$

for any $i = 1,2,3,\ldots,g$

and for any: $t = 1,2,3,\ldots,t_n$

3) All activities will be performed.

$$\sum_{t=1}^{t_n} X_{jt} = d_j$$

for any $j = 1,2,3,\ldots,a$

4) No activity will be split.

$$d_jX_{jt} - d_jX_{j(t+1)} + \sum_{s=t+2}^{t_n} X_{js} \leq d_j$$

for any $j = 1,2,3,\ldots,a$

5) No activity will be started before its predecessors are completed; time-precedence relations in the network are not broken.

$$d_pX_{jt} \leq \sum_{s=1}^{t-1} X_{ps}$$

where: p = any predecessor of j

for any: $t = 1,2,3,\ldots,t_n$

and for any $j = 1,2,3,\ldots,a$

It should be noted that this integer programming model does not allow for crashing or stretching of activities. Also it does not assure the finding of the earliest schedule satisfying the above constraints as the Bowman model does (suitable for the resource allocation problem). It does assure, however, the finding of a feasible schedule that completes the project in due date with the minimum daily resource allocation (resource leveling problem).

The large number of variables and equations involved, coupled with additional equations and slack variables necessary to assure an integer solution, would lead to such computational complexities that attempting a solution with this technique is almost impossible, and always impractical, even for a small project. Furthermore, a real life project with hundreds of activities will surely exceed the capacity of present computers. For the reasons discussed above, we can say that his analytical solution is impractical for our scheduling problem.

The author concentrated his research on developing a heuristic routine that, following the formulation given above, would originate the best feasible schedule. The author wishes to emphasize that the word "best" implies only that we will try to find a feasible schedule with the minimum peak resource allocations, not just any feasible solution. This best feasible solution cannot be assured to be the optimal feasible solution of the scheduling problem, because optimal solutions are assured only by analytical techniques. However, as we shall see in Chapter III, the heuristic routine developed in this study actually originates schedules that converge to the optimal schedule, and there is a good chance of finding a near optimal schedule if not the real optimal one.

There will be projects for which no feasible solution can be found within the given constraints. In this type of situation, we have to decide whether to increase the level of available resources or to allow some days of project slippage - or perhaps both at the same time - to be able to realize the project. "Project slippage" (S) stands for the number of days of delay in the completion time of a project beyond the corresponding fixed due date. In any case, whichever decision we make will originate in unexpected extra cost for the project which must be reduced to a minimum. We say that this extra cost is unexpected, because an efficient management usually seeks to set a due date that provides enough flexibility to avoid this kind of situation. We are therefore confronted with the problem, of trading the cost of increasing the level of available resources against the penalty costs of delaying the completion of the project. Dollar penalties for delays are common in project contracts, especially in the construction industry.

The heuristic routine, when confronted with this situation of no feasible solution, will originate several schedules with different amounts of project slippage, so that the project manager can decide which one to take. Since the constraint on project duration (e.g. project due date) is broken, we need to set another constraint that replaces it in the formulation: we need to specify a "maximum project slippage" (S_m). This new time constraint may be posed as follows:

Alternative time constraints:

$$t_n \leq t_d + S$$

where:

$S = 0,1,2,\ldots,S_m$ is an integer variable that fixes different amounts of project slippage.

S_m = maximum of project slippage.

II. LITERATURE REVIEW

This chapter reviews the various solution techniques that have been proposed to solve problems of project-network scheduling with limited resources. The review is restricted to the presentation of the basic concepts and approaches involved in each technique (important to our research) as described in the open literature. All of these techniques have a common foundation on standard PERT-CPM procedures, and are in general heuristic solutions of non-generalized problems.

Analytic solution to the generalized problem of project scheduling with resource constraints has not been successful up to the present date because of the following main difficulties:

1. The large amount of alternatives available for scheduling the various activities which leads to a combinatorial problem of formidable magnitude, even for small sized problems.
2. Embedded interdependence of activities as a result of sharing the same resources that are rarely known; and even if they were known, to incorporate them into the formulation would present a major problem.
3. Some activities can also be split in time, crashed, or extended to suit available levels of resources; if we couple them with

the possibilities of overtime work and possible substitution of resources, we might be invalidating the original estimates and complicating the formulation to a point where no meaningful solution is possible.

4. Even with a trimmed-down formulation, a practical size project would probably exceed the capacity of present computers and, in any event, would be an inefficient means of solving the problem.

Analytic formulations of the line balancing problem[42] and of the job shop problem[5] can be transformed to provide analytic formulations for problems of project scheduling with limited resources, but they are interesting mainly from the conceptual standpoint of the problems rather than as practical or efficient means of solving them. We are safe in saying that the heuristic techniques aided by the computational power of digital computers will continue to contribute in an important way to solve some of the complex planning and scheduling problems of project management.

Solution techniques for problems of project-network scheduling with limited resources usually take one of the following two forms:

a. Resource leveling techniques--These techniques attempt to reduce peak resource allocations as much as the time precedence relations of the activities and events in the network will permit within a given project due date. These techniques are usually suitable for use with, or subsequent to, the time/cost tradeoff analysis.[10,26]

b. Resource allocation techniques--These techniques try to allocate all available resources to selected subsets of activities on a day-by-day basis, attempting to find the earliest project completion time consistent with the stated level of available resources.

In order to easily identify the scope of the problem that each technique is aimed at solving, we will further identify them as:

1) single-project single-resource techniques,
2) single-project multi-resource techniques,
3) multi-project single-resource techniques, and
4) multi-project multi-resource techniques,

whenever necessary.

Resource Leveling Techniques

A typical single-project single-resource leveling problem may be posed as follows: the critical path through the network has been determined and all activities have been tentatively scheduled at their earliest start times. When all the activities in the network are scheduled at their earliest start times, we say that we have an "earliest start schedule" for the project. The profile for the daily resource allocations might appear as in Figure 1-A. The problem is to level down the peak resource allocations as much as the network will permit, subject to the constraint on project duration given by the project due date (t_d). The resource profile for a feasible solution schedule might appear as in Figure 1-B.

Burgess and Killebrew[7] suggest a method of comparing alternate schedules obtained by sequentially moving, in time, slack activities and computing the resulting resource profiles. The measure of effectiveness is the sum of squares of the daily resource allocations. This measure has the property of becoming smaller as the variation in resource allocations from day to day becomes smaller.

They present a computer program for the method and give examples of its application for single-projects with one and two resources. They point out that the method does not necessarily produce optimal solutions and may give different solutions for the same problem if different sequences of activities are used as the initial schedule; therefore, a large number of alternate schedules must be computed, using varying activity orderings.

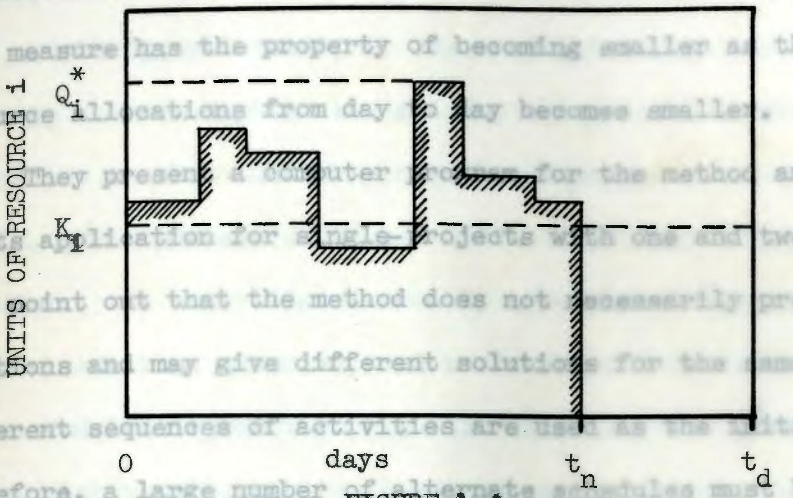


FIGURE 1-A

TYPICAL RESOURCE PROFILE OF AN EARLY START SCHEDULE

Dewitte[12] presents a computerized resource (manpower) leveling procedure developed at Hughes Aircraft Company. Like the Burgess method, it is designed to minimize the variation in resource allocations from day-to-day by adjusting the start times of slack activities. The measure of effectiveness is the absolute deviation of the daily resource allocations from a calculated project mean level of resource allocation. Basically, the method consists of partitioning the resource profile into specially-derived intervals and then sequentially leveling each interval, revising early start times of successor activities where necessary.

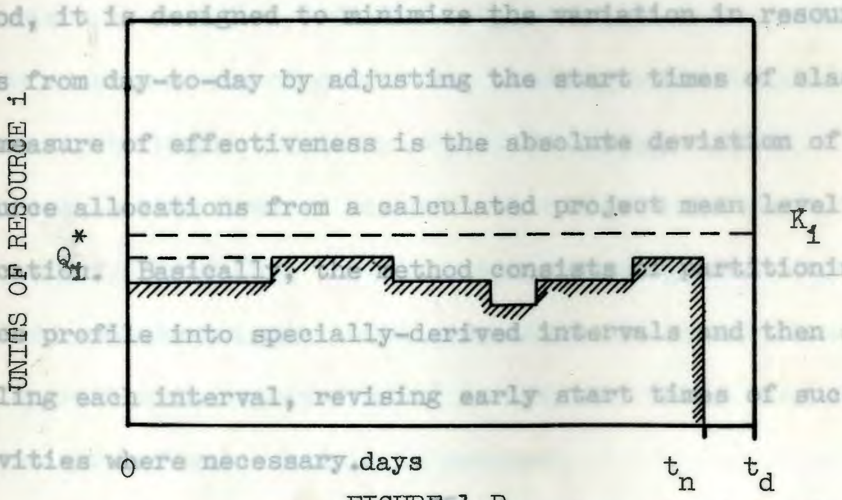


FIGURE 1-B

TYPICAL RESOURCE PROFILE OF A FEASIBLE SOLUTION SCHEDULE

Levy, Thompson, and Wiest[15] present a method for leveling resource (manpower) allocations which is similar in many respects to the two methods just described above. Their problem is essentially one of multi-project multi-resource. First of all an early start schedule, along with total slack values for all activities, is

FIGURE 1.--RESOURCE LEVELING PROBLEM

and computing the resulting resource profiles. The measure of effectiveness is the sum of squares of the daily resource allocations. This measure has the property of becoming smaller as the variation in resource allocations from day to day becomes smaller.

They present a computer program for the method and give examples of its application for single-projects with one and two resources. They point out that the method does not necessarily produce optimal solutions and may give different solutions for the same problem if different sequences of activities are used as the initial schedule; therefore, a large number of alternate schedules must be computed, using varying activity orderings.

Dewitte[12] presents a computerized resource (manpower) leveling procedure developed at Hughes Aircraft Company. Like the Burgess method, it is designed to minimize the variation in resource allocations from day-to-day by adjusting the start times of slack activities. The measure of effectiveness is the absolute deviation of the daily resource allocations from a calculated project mean level of resource allocation. Basically, the method consists of partitioning the resource profile into specially-derived intervals and then sequentially leveling each interval, revising early start times of successor activities where necessary.

Levy, Thompson, and Wiest[25] present a method for leveling resource (manpower) allocations which is similar in many respects to the two methods just described above. Their problem is essentially one of multi-project multi-resource. First of all an early start schedule, along with total slack values for all activities, is

Wilson[42] presents a method designed to produce the minimum amount of daily resource allocations required to achieve a given project due date. Instead of the random choice, he incorporates a dynamic programming scheme at each iteration to determine feasible subsets of activities to be moved. However, he makes the simplifying assumption that each activity requires one unit of the same kind of resource, and that each activity can be interrupted and started again without penalty. This latter assumption is expressed by subdividing the activities into "tasks" that have a duration equal to one unit of time. The method is simple to use for small projects but becomes cumbersome as the number of events and activities in the network increases. Even though his method would be easy to program for machine operation, Wilson does not discuss it.

In addition to this single-project single-resource solution technique, Wilson also presents an interesting comparison of the resource leveling problem in networks with the assembly line balancing problem.

Black[3] presents a technique similar to Wilson's technique just presented above. Black uses the approach of subdividing the activities into unit time portions, the assumption that activities can be split in time, and an adaptation of the line-balancing problem as a base. His methodology is based on the Gutjahr-Nemhauser algorithm for the line-balancing problem. It involves generating of feasible subsets of activities, and then constructing a new network using the generated subsets as activities and stated resource constraints as time-precedence relations. This method will produce all feasible solutions with respect to given resource constraints. It is computationally most efficient when dealing with the single-project multi-resource case, although comparatively speaking it is cumbersome and, in present form, computationally prohibitive for large networks.

Resource Allocation Techniques

A typical single-project single-resource allocation problem may be posed as follows: the critical path through the network has been determined, as well as the values of the different slack times of all the activities in the network. We start by scheduling a selected subset of activities during the first day so as to allocate all available resources; the rest of the activities are temporarily postponed. We continue scheduling selected subsets of activities on a day-by-day basis, always revising the previous scheduled days to avoid violations of the time-precedence relations in the network, and always trying to allocate all available resources. The resource profile resulting from ten days of scheduling on a thirty-two days project may look like that shown in Figure 2-A. We continue with this day-by-day allocation with scheduling process until we reach the solution schedule, whose resource profile may resemble that shown in Figure 2-B.

The essential heuristics of these resource allocation techniques are those that determine which activities shall be scheduled and which shall be postponed in any day of this progressive scheduling process. The approach most frequently used is to use activities' slacks as a basis of priority, scheduling first those activities which are most critical.

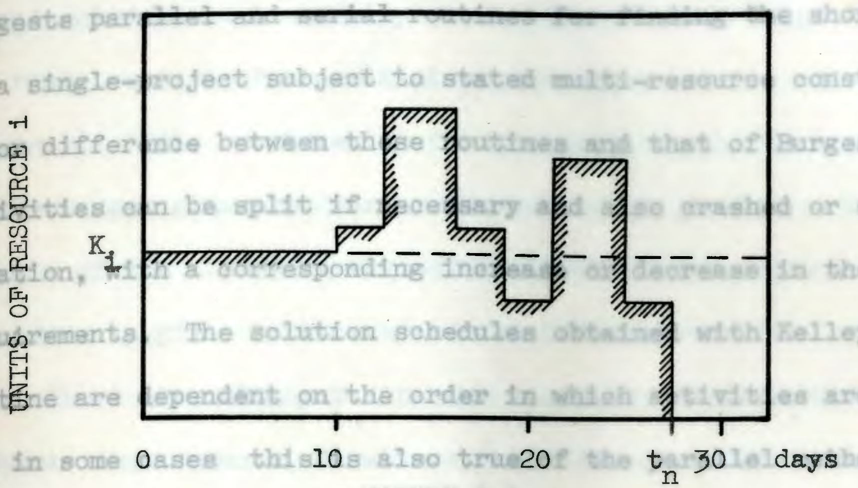


FIGURE 2-A

TYPICAL RESOURCE PROFILE AFTER TEN DAYS OF PROGRESSIVE RESOURCE ALLOCATION WITH SCHEDULING PROCESS

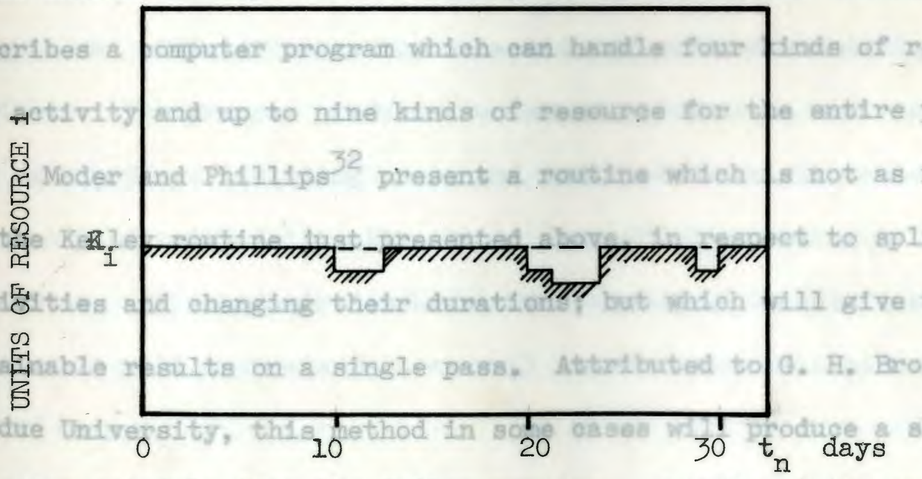


FIGURE 2-B

TYPICAL RESOURCE PROFILE OF THE SOLUTION SCHEDULE

FIGURE 2.--RESOURCE ALLOCATION PROBLEM

Kelley[23] presents a method which is in many respects similar to that presented by Burgess[7] for the resource leveling problem. Kelley suggests parallel and serial routines for finding the shortest schedule of a single-project subject to stated multi-resource constraints. The major difference between these routines and that of Burgess, is that activities can be split if necessary and also crashed or extended in duration, with a corresponding increase or decrease in their resource requirements. The solution schedules obtained with Kelley's serial routine are dependent on the order in which activities are scheduled, and in some cases this is also true of the parallel method. Therefore he suggests repeating the scheduling procedures with various activities orderings. He also discusses additional refinements of the scheduling procedure, such as the use of a "threshold" resource requirement (minimum resource allocation needed to start an activity), which would increase the practical utility of his technique. He also describes a computer program which can handle four kinds of resource per activity and up to nine kinds of resource for the entire project.

Moder and Phillips[32] present a routine which is not as flexible as the Kelley routine just presented above, in respect to splitting activities and changing their durations; but which will give the best obtainable results on a single pass. Attributed to G. H. Brooks of Purdue University, this method in some cases will produce a shorter duration schedule than the Kelley routine. Details of the routine and examples of its application are given in the cited reference.

Lambourn,[24] and Moshman, Johnson, and Larsen,[33] present a multi-project multi-resource allocation technique which is known as RAMPS

("Resource Allocation and Multi-Project Scheduling"). RAMPS is a computerized method designed to handle several projects simultaneously and schedule each activity so that project due dates are achieved and "idle resources" are minimized subject to stated resource constraints. "Idle resources" refers to those available resources which have not been allocated during the scheduling process. Although details of the algorithm are not available, a description of the procedure is given in the referenced publications.

Three sets of input data are required for the activities: resource requirements, durations, and cost of splitting one activity once it has been started. Also, certain project information is required, such as starting date, due date, and dollar-penalty rate for project slippage or (alternatively) a project priority rating. Finally the "scheduling objectives" in terms of six factors, such as work continuity, idle resources, et cetera, must be assigned relative weights and will influence the selection of various scheduling possibilities. The program produces two major outputs: a work schedule for each project including costs and daily resource allocations, and a summary of total daily allocations classified according to kinds of resources. If the desired schedule for a project is not feasible, the output will indicate this as well as the resource constraint that cannot be met.

Since details of the RAMPS computational algorithm are not available in the open literature, we cannot give an analysis of the shortcomings of this technique. However, from the descriptive information provided it is most certainly a heuristic system based on juggling slack activities in a manner somewhat similar to techniques described previously. As such, it probably does not necessarily produce the optimum schedule for a given problem.

McGee and Markarian[28] present a methodology which begins with a time-cost tradeoff formulation of the CPM type. Two sets of time-cost data are required for the activities: a "minimum essential effort" (maximum activity duration with minimum resource requirement) and a "crash effort" (minimum duration with maximum resource requirement). A linear function is assumed to exist between these two points. Kinds of resource and constraints on allocations must be given for each time interval. An initial schedule is obtained using the "minimum essential effort" values for allocating resources. If, for this schedule, one or more of the constrained levels of resources are exceeded, slack activities are rescheduled in an attempt to stay within the constraints. If this action proves unsuccessful, they conclude that the fixed project due date cannot be achieved without additional resources. If the schedule allocates resources without exceeding the constrained levels, a check is made to determine whether the project completion time (given by this schedule) is equal to or less than the fixed project due date. If the project completion time is larger than the project due date, successive crashing of less costly activities on the critical path is made until the desired due date is achieved, always observing that the constrained levels of resource are not exceeded.

To handle several projects simultaneously the "minimum essential" resource allocation schedule for each project is determined first.

Then the values for S_i (completion date minus due date for project i) are computed. The project with the largest S_i value is crashed first until it achieves its project due date ($S_i = 0$). New comparisons of S_i are made to determine which project should be crashed next. Iterations continue in this fashion until all due dates are met (all $S_i \leq 0$), or until no further allocations are possible because of the resource constraints.

No computer program is provided for this technique. However, McGee and Markarian do provide logic flow charts of the algorithms involved in their routine.

Wiest[38-41] has developed a heuristic technique which he calls SPAR-1, and which is designed to allocate available resources on a day-by-day basis to project activities listed according to their early start times. His sequential schedule procedure starts by allocating resources, on the first day, to activities selected from a list of those currently available and sorted in order of their total slack. The most critical activities, those with the smallest amount of slack, have the highest probability of being scheduled first, and as many of these activities are scheduled as available resources permit. Available activities not scheduled on the first day will be tried on the second day and so on, until all activities are finally scheduled, yielding a solution to the problem.

Three sets of data are required for the activities: "normal crew size" or normal amount of resources required by the activity, "maximum crew size" or the maximum amount of resources needed to crash the activity to its minimum duration, and "minimum crew size" or the minimum amount of resources that still permits the successful completion of the activity. The rules for "crew size" selection are:

1) If an activity to be scheduled is "critical" (the degree of criticality is given by an input parameter), it is placed on a priority list and given special treatment.

2) If sufficient resources are available, the activity is scheduled at its maximum crew size.

3) If insufficient resources are available, then an attempt is made to obtain the required resources by means of the "borrow" and "reschedule" routines, which will be described later on.

4) If all efforts fail, however, and the activity cannot be scheduled even at minimum crew size, then its start date is delayed and will be tried for scheduling on the next day.

5) Before any new activity is scheduled on a given day, all activities already scheduled and still active are examined; if any of these activities is critical and has a crew size less than its maximum, and if resources are available, the activity's crew size is increased as much as possible up to its maximum.

6) If an activity requires several kinds of resources, separate activities are created for each kind of resource and these activities are constrained to start on the same day with the same level of resource allocation - that is, normal, minimum, or maximum crew size.

Borrow from active activities.--If available resources are not sufficient for scheduling some critical activity, we will enter into a routine that examines currently active activities to see if we can borrow resources from them. Resources are borrowed from an activity only when the resultant stretching of the activity will not delay the project completion date.

Reschedule of active activities.--Sometimes a critical activity, j, can be scheduled, if other activities previously scheduled which use the same kind of resources had been postponed to a later date. This routine scans the list of currently active activities and picks out those which could be postponed without delaying the project completion date. If sufficient resources can be obtained in this way and/or from the borrow routine described above, then activity j is scheduled and the necessary adjustments are made in previous allocations.

SPAR-1 is able to accomodate single or multiple projects, variable crew sizes, activities that can be split, shift or non shift scheduling, and various criteria functions for evaluating a schedule. Probabilistic elements in the program can lead to different schedules with successive applications of the program, then the best of these solution schedules can be selected.

SPAR-1 is currently written in FORTRAN-IV and can handle a project with up to 1200 activities, 1000 events, and 25 kinds of resources.

T. C. Hu[21] presents an analytic method for minimizing schedule duration of a single-project, given a specified availability of a single-resource, under the simplifying assumption that each activity requires one unit of time and one unit of resource for its completion. The method consists of labeling each node (in an activity-on-node network) with the value $a_i = X_i + 1$, where X_i is the length of the longest path from node N_i to the final node, then grouping M nodes at a time, where M is equal to the number of units of resources available. The number of such groups is then equal to the minimum number of time-units required for project completion. For example, Figure 3 shows an activity-on-node network with the calculated a_i values shown above each node. If there are three units of resource available during any time unit $(M = 3)$, the nodes could be grouped as shown by the broken lines, always giving preference to high values of a_i in first groupings. Since there are six groupings, the project requires six units of time for completion.

Hu proves in his article that this method, which he calls "cutting the longest queue", gives a minimum-time solution schedule for completion of all activities.

Hu's method is very simple to apply for small networks, but as the network increases in size (say for more than twenty activities) we quickly run into computational complexities. This method could be programmed for machine operation, thus increasing the size of the networks that could be handled.

Unfortunately, we cannot say that Hu's method is a practical one, because it is very unlikely to find a real-life project-network for which his model can be successfully applied.

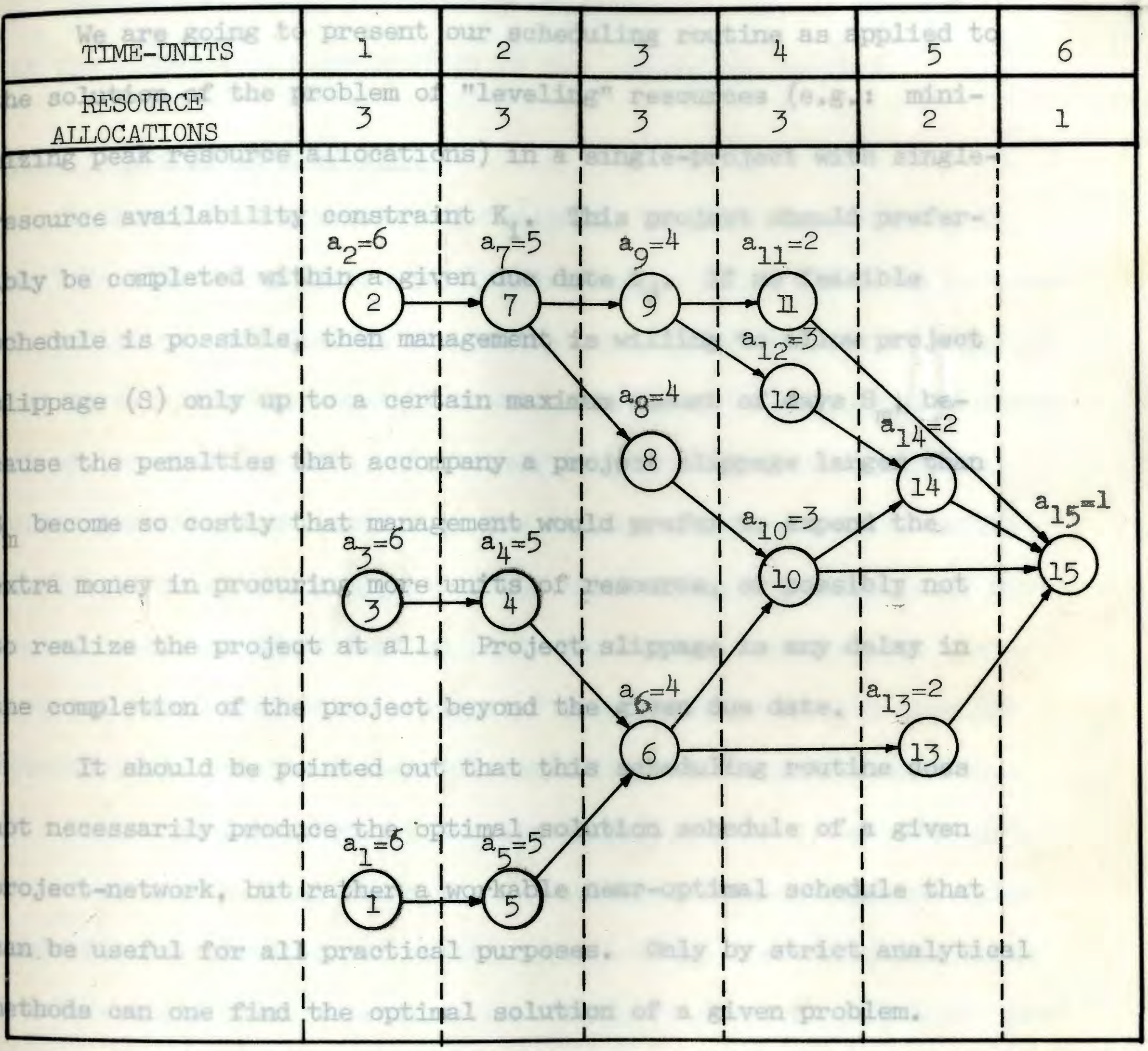


FIGURE 3.--Illustration of Hu's method (when M = 3).

III. SINGLE-PROJECT SINGLE-RESOURCE SCHEDULING

We are going to present our scheduling routine as applied to the solution of the problem of "leveling" resources (e.g.: minimizing peak resource allocations) in a single-project with single-resource availability constraint K_i. This project should preferably be completed within a given due date t_d. If no feasible schedule is possible, then management is willing to allow project slippage (S) only up to a certain maximum amount of days S_m, because the penalties that accompany a project slippage larger than S_m become so costly that management would prefer to expend the extra money in procuring more units of resource, or possibly not to realize the project at all. Project slippage is any delay in the completion of the project beyond the given due date.

It should be pointed out that this scheduling routine does not necessarily produce the optimal solution schedule of a given project-network, but rather a workable near-optimal schedule that can be useful for all practical purposes. Only by strict analytical methods can one find the optimal solution of a given problem. However, because of the extreme computational difficulties encountered when attempting an analytical solution in problems of this type, only heuristic solutions are in common use up to the present date. The solution techniques available in the open literature have been discussed in Chapter II. Heuristic is a synonym for "rule of thumb".

Having done these preliminary observations, let us proceed with the step-by-step explanation of the scheduling routine:

Step A. Planning the project

Planning is the process of analyzing the project, breaking it down into elementary operations (activities) necessary for its successful completion, and finding the technological time-precedence order in which these elementary operations must be done. There must be a statement of the starting date and the due date of the project, whether derived internally or imposed by the customer. One must also define the objectives of the project and its limitations. Each activity must have an estimated "duration" or performance time, coupled with its "resource requirements" or amounts of each kind of resource that the activity needs for its successful completion in the given duration.

The project is then represented graphically by a "network" of arrows and circles that shows the technological time-precedence relations of its activities and events. The longest time-consuming chain of arrows and circles determines the "earliest project completion time" and receives the name of "critical path".

The events are assigned identification numbers in such a way so as to prevent two events from having the same number and also taking care that any activity connecting two nodes always goes from a lower event number to a larger event number. Each activity can now be identified by these two event numbers as: $A_{p,q}$, where p stands for the identification number of its starting event and q stands for the identification number of its ending event, and where p is always less than q.

The planning process ends with the final listing and tabulation of the data. It is recommended that the following method of identifying events and activities be used, because it will help to simplify the retrieval of data for future scheduling computations:

1. The events are assigned the identification numbers (EIN's): 1,2,3,...e; so that the largest identification number e is assigned to the ending event of the project and is also equal to the total number of events in the network.
2. The events' identification numbers are already identifying the activities, in an implicit way, through the $A_{p,q}$ notation explained before. However, the activities are going to be identified in an explicit way, after they are listed as follows:

   first, we group the activities according to their p events;

   second, we list the activities within each group, in ascending order, according to their q events;

   third, we obtain the total list of activities by writing one group after the other, in ascending order, according to the p events;

   fourth, we go through the list of activities assigning the identification numbers (AIN's): 1,2,3,....,a; so that the largest activity identification number a is also equal to the total number of activities in the network.

Step B. Calculation of network characteristics

The data listed and tabulated in the planning step may not be complete for our scheduling purposes. We need to compute or obtain the following information:

For each activity.--Total slack, free slack, independent slack, safety slack, whether it belongs to the critical path or not, whether it is a dummy activity or not, and the amount of "mobility" that each activity has as compared to the mobility of all the other activities in the network (to be measured by the "index of mobility" values); also the earliest start, earliest finish, latest start, and latest finish, if they had not been calculated during the planning step.

For each event.--Earliest time, latest time, slack time, and whether it belongs to the critical path or not.

For the network.--Earliest project completion time (obtained by calculating the critical path), project slack, project due date, maximum project slippage, maximum resource availability per unit of time (K_i), total number of activities, and total number of events.

All these characteristics of the project-network will be used, in one way or another, during the scheduling procedures of our routine; and therefore it is of the utmost importance to obtain their values before we start the scheduling procedures of the following steps.

The definitions and computation algorithms of these network characteristics can be found in almost any textbook dealing with network planning and scheduling, because they form part of the standard terminology and computations of PERT-CPM techniques. The reader interested in obtaining a detailed explanation of these network characteristics is referred to Buffa,[6] Horowitz,[20] Kelley,[23] Levy and Wiest,[26] Meyer,[30] Muth,[34] or Waldron.[37] However, since the characteristic of <u>mobility</u> (or flexibility) of the activities is measured in a rather different way in this thesis, we shall explain it.

Mobility of an activity is the property that measures its ability of being rescheduled at a later date (or at an earlier date) relative to its current position in the schedule and without delaying the current project completion time. The mobility of an activity is a direct function of its slack times, and is constrained by the current positions of its neighboring activities, the current project completion time, and the current level of available resources.

We will measure the property of mobility by assigning to each one of the activities in the network an <u>index of mobility</u>, which will give the order of the activities according to the amount of mobility that each one has, and also according to the varying degrees of difficulty that they have in making use of this property because of their position in the network.

We are presenting next the algorithm that assigns the index of mobility values to each activity in the network:

Index of mobility assignation algorithm.--

1. Set the index of mobility value (IM) equal to the total number of activities in the network (a); that is:

   Set:

   $$IM = a$$

2. Assign the current IM value to one of the non-indexed activ-

ities and then execute rule 3. The rule to decide which one of the still non-indexed activities should be assigned the current IM is as follows:

Assign the current IM value to the non-indexed activity that has the largest amount of total slack. If there is a tie in total slacks, assign this IM to the activity in the tie that has the largest amount of free slack. If there is a tie both in total slacks and free slacks, assign this IM to the activity in the tie that has the largest amount of independent slack. If there is a tie in all three: total slacks, free slacks, and independent slacks, assign this IM to the activity in the tie that has the largest amount of safety slack. If there is still a tie, assign this IM to the activity in the tie that has the largest starting event (event p). If there is still a tie, assign this IM to the activity in the tie that has the largest ending event (event q).

This rule will always assign the current IM value to only one activity because even in the case of ties in all slack times, it is impossible for two activities in a network to have the same starting event and the same ending event.

3. Since the current IM value had been already assigned to one activity (by executing rule 2 above), we should not assign the same IM value to another activity. Therefore, we decrease by one the value of IM before executing rule 4, that is, we set:

$$\text{new IM} = \text{old IM} - 1.$$

4. Here we simply check the current value of IM (set by rule 3). IM will be equal to zero when all the activities in the network had already been indexed. Therefore our rule here is:

Return to execute rule 2, if the current IM value is larger than zero; if IM is equal to zero, stop the algorithm.

It should be emphasized that these index of mobility values will remain the same even if we delay the project completion time given by the critical path - by setting a t_n larger than the t_n' of the early start schedule - because that action only makes the slack times of all activities increase by the same amount of time-units, the amount equal to the difference between the current project completion time (t_n) and the earliest project completion time given by the critical path (t_n').

Step C. Initial schedule

We will use as the initial schedule the one in which all the activities are scheduled to begin at their earliest start dates - e.g. the "early start schedule" - simply because it is very easy to obtain, since the earliest start and earliest finish dates of the activities come straight from the standard PERT-CPM calculations. Actually any other schedule - any one that is obtained by intuition or by scheduling the activities at random - would be as good as the initial schedule that we are proposing, provided of course that the time-precedence relations in the network are honored.

The corresponding resource profile of our initial schedule is then obtained by plotting total daily resource allocations against time.

Step D. Selecting activities for rescheduling

Looking at the resource profile of our initial schedule (a typical resource profile of an early start schedule was shown in Figure 1-A, page 16), it becomes evident that in order to level down the daily resource allocations, we have to cut the peaks and fill in the valleys of the profile. Since it is possible to find several peaks which have the same height and are the tallest of the profile, we have to decide which one of these "highest peaks" will be cut first. As a rule of thumb we will always cut the highest peak which occurs at the farthest day from the project starting date t_o; that is, the farthest to the right on the resource profile.

This peak was selected because it seems to provide a better chance of being cut than the others. Even if the rescheduling procedures explained in the following steps fail to cut this peak, we could still force the cutting of this peak by delaying the current project completion date. Here again, as in any other heuristic decision, we cannot prove this to be the optimal choice, we can simply say that our decision produces good results and seems to make the best of all the available choices. Hereinafter we shall call this peak the "right most largest peak" or "RMLP".

The RMLP results from the addition of the resource requirements of all the activities scheduled during the same day at which the RMLP occurs; therefore it should be evident that, by rescheduling one of these activities to start at a later date, or to end at an earlier date, we will cause the reduction of its height (e.g. we will cut it), provided of course that the resource requirements of the rescheduled activity have a value greater than zero.

Next we have to decide which one of the activities contributing to the height of the RMLP will be rescheduled. It is here that we make use of the index of mobility values (IM's) calculated in Step B. We list these activities in descending order according to their IM values. This listing of the "activities on the RMLP" is needed for the scheduling procedures of our routine, as we shall explain in the following Step E.

Step E. Realizing the reschedule of the activities

When we reschedule one activity while keeping all the others momentarily still, we are actually moving this activity along the time axis and it is obvious that its resource requirements are also moving accordingly, thus changing the resource profile of the project. Furthermore, this activity could be rescheduled at a later date or an earlier date from its present position in the project schedule; that is, this activity could be moved forward or backward along the time axis. The restrictions that we have for rescheduling activities are:

1. The present position of its starting event (event p) and that of its ending event (event q); these events actually behave like barriers between which the activity can float freely.
2. The changes in the resource profile that this rescheduling causes should not originate another peak of the same height or of a

larger height than that of the current RMLP.

3. The activity should be rescheduled as far away as possible from its present position, while filling (resource-wise speaking), the lowest valley that it may reach. This action is convenient for the future scheduling of other activities, since the mobility of the activities is constrained by the height of the current RMLP and by the current positions of its neighbor activities. If the activity cannot be rescheduled at a "better position" - e.g.: all the possible rescheduling positions of this activity only originate peaks of the same height or larger than the current RMLP - we will leave it untouched, and we will say that the activity has been rescheduled in its very same position.

Hereinafter, we shall call "iteration" any successful rescheduling of one activity or group of activities, that cuts the height of the current RMLP by at least one unit of resource, thereby originating a better schedule than the previous one. It should be remembered, however, that the new current RMLP could have the same height as its predecessor, in case we had several peaks of the same height during several different days of our schedule.

There are two types of iteration:

Iteration Type I.--Whether rescheduling forward or backward, this iteration consists of the rescheduling of only one of the activities which are currently scheduled during the same day on which the RMLP occurs. The activities are tried for rescheduling one by one in descending order according to their index of mobility values until an iteration is achieved, or until we exhaust all the "activities on the RMLP". This type of iteration is bound by the three restrictions given above.

Iteration Type II.--When rescheduling forward, this iteration consists of rescheduling forward a group of activities, one by one in descending order according to their identification numbers (AIN's), starting with the activity that has the largest AIN in the network. The largest AIN in our network is "a", which is also equal to the total number of activities in the network, due to the numbering system recommended in Step A. We continue rescheduling forward the activities and updating the position of the events until an iteration is accomplished, or until we have tried to reschedule forward the "activity on the RMLP" that has the lowest AIN number.

When rescheduling backward, this iteration consists of rescheduling backward a group of activities, one by one in ascending order according to their AIN numbers, starting with the activity that has the lowest AIN in the network. The lowest AIN in our network is "1", also due to the numbering system recommended in Step A. We continue rescheduling backward the activities and updating the position of the events until an iteration is accomplished, or until we have tried to reschedule backward the "activity on the RMLP" that has the largest AIN number.

During the research phase of this thesis, it was noticed that the iteration Type II gives better results when it is released from the restriction of "filling the lowest valley that it may reach". It turned out to be better to reschedule the activities as far away as possible, but without originating another peak of the same height as the current RMLP. This type of iteration tends to pack the activities to either side of the schedule - resource wise speaking - leaving an empty space in the center of the resource profile. We will have the opportunity of noticing this action later on, on pages 67 and 69.

The following algorithm for trying to obtain an iteration of either type contains the heuristic rules that turned out to give better results during the research phase of this thesis. We know - from the execution of the previous steps - the height of the current RMLP as well as the day on which it occurs; we also know which activities are currently scheduled during the same day on which this RMLP occurs as well as their identification numbers (AIN's) and their index of mobility values (IM's).

Rescheduling algorithm

1. Set the rescheduling direction as "forward".
2. Try to obtain an iteration Type I.
3. If the iteration was achieved, go back to execute Step D (explained on page 37); if the iteration was not achieved, continue.
4. Try to obtain an iteration Type II.
5. If the iteration was achieved, go back to execute Step D; if the iteration was not achieved, continue.
6. Set the rescheduling direction as "backward".
7. Try to obtain an iteration Type I.
8. If the iteration was achieved, go back to execute Step D; if the iteration was not achieved, continue.

9. Try to obtain an iteration Type II.

10. If the iteration was achieved, go back to execute Step D; if the iteration was not achieved, continue.

11. Set the rescheduling direction as "forward".

12. Try to obtain an iteration Type II.

13. If the iteration was achieved, go back to execute Step D; if the iteration was not achieved, continue.

14. Set the rescheduling direction as "backward".

15. Try to obtain an iteration Type I.

16. If the iteration was achieved, go back to execute Step D; if the iteration was not achieved, continue.

17. Try to obtain an iteration Type II.

18. If the iteration was achieved, go back to execute Step D; if the iteration was not achieved, continue to reschedule backward all the activities of the network - accepting the formation of peaks with a height equal to the height of the current RMLP - and then go to execute Step F (to be explained on page 43). We conclude here that no iteration is possible, because the current schedule is actually the best schedule that our routine can find for the current project completion date (Current t_n). If we wish to obtain a further minimization of the peak resource allocations, we have to delay the project completion date. Whether we can or can not delay the project completion date will be determined by the following steps.

Step F. Utilizing the project slack

Project slack (PS) is the difference - expressed in days - between the given project due date (t_d) and the current project completion date t_n. According to our assumptions, the time constraint for our project can be equally satisfied by any project completion date equal to or larger than the earliest project completion date given by the critical path of the network (t_n') provided that the t_n does not exceed the given t_d. From the definition of PS, it is evident that we might be faced with one of the following three situations:

1. PS is negative.--This is evidently the result of poor management planning, because a rational manager never commits himself to realize a project in a shorter time than the earliest project completion time given by the critical path of the network. The only action that we can take to avoid the corresponding penalties - for this delay of the project completion beyond the agreed t_d - is to start "crashing" selected combinations of activities until the new t_n becomes equal to or less than the given t_d. Crashing activities usually leads to an increase in the resource requirements of the crashed activities, and since this action is beyond the scope of the assumptions given in Chapter I, we will treat any negative PS as if it were a positive project slippage (S). The utilization of project slippage will be explained and discussed later in Step G (page 45).

2. PS is zero.--This situation is not adverse from the point of view of our time constraint, since the t_n can be equal to t_d and still yield a feasible solution to our problem, provided of course that the

height of the RMLP of this solution schedule is equal to or shorter than the height of the maximum resource availability allowed for the project (K_i).

If the height of the RMLP of the solution schedule is larger than the height of the given K_i, then our solution schedule is not feasible - e.g.: the project cannot be realized under the given constraints, t_d and K_i. To be able to realize this project, we have to allow some days of project slippage (S), an increase in the height of the given K_i, or both.

3. PS is positive.--This situation is the most advantageous for our scheduling objective of minimizing peak resource allocations, because it provides more mobility for the activities in the network. It is also the most commonly found in real life, because any rational manager will seek to set a project due date that provides him with the opportunity of trying several completion dates without having to pay penalties for project delays. Contracts usually specify penalties only for delays beyond an agreed due date.

The following algorithm summarizes the computations and decisions to be made in this Step F of our scheduling routine:

1. Determine whether the current PS of the project has a negative, zero, or positive value. The formula to be used is:

$$\text{Current PS} = t_d - \text{current } t_n$$

If PS is negative or zero, go to execute rule 2; if PS is positive, go to execute rule 3.

2. Consider this negative or zero PS as though it were a positive project slippage (S) of the same number of days. That is, set:

current S = - current PS

Then execute Step G (below).

3. The number of slack days this positive PS provides will be used to update the values of the following network characteristics as shown below:

Set:

new current t_n = given t_d

new latest time of event p = current latest time of event p + current PS

for all events: p = 1,2,3,...,e

new latest start of activity A = current latest start of activity A + current PS

new latest finish of activity A = current latest finish of activity A + current PS

for all activities: A = 1,2,3,...,a

Then go back to execute Step D (page 37).

Step G. Utilizing the project slippage and publishing the solution schedules

Project slippage is any delay in the completion date of the project beyond its corresponding due date; e.g., when the current project completion date t_n happens to be larger than the given project due date t_d. The concept of project slippage is closely related to the concept of project slack: it depends entirely on whether the current t_n is smaller or larger than the given t_d to call their difference a project slack (PS) or a project slippage (S). That is why we said in Step F (page 43), that a negative PS will be considered as if it were a positive S of the same number of days.

In project contracting, especially in the construction industry, it is common to specify penalties for delays beyond an agreed due date; e.g., penalties for "project slippage". These penalties are usually in the form of monetary fines which increase at a much faster rate than a simple linear proportionality. For example, a contract may specify that for the first day of slippage the penalty is $1,000.00; for two days, $3,000.00; for three days, $6,000.00, and so on; while in a simple linear proportionality (or one-to-one rate), the penalty for three days of slippage would amount to $3,000.00. All this makes the utilization of project slippage most undesirable to any rational manager. On the other hand, to increase the level of available resources over and above the amount normally available (e.g., to increase the given K_i), is also undesirable because of the cost involved in procuring those extra units of resources. There may even be cases in which those extra units are impossible to procure.

If a manager is confronted with a project-network for which no feasible solution schedule can be found (e.g., he does not find a schedule which completes the project in due time with the available resources), he would like to know how the daily resource allocations vary in relation to different amounts of project slippage, so that he can make a cost evaluation and select the schedule that gives the less costly combination of project slippage and incrementation of available resources.

It is also obvious that our troubled manager would be willing to allow project slippage only up to a certain maximum number of days

S_m (maximum project slippage), before the penalties get so heavy that he has to accept a large incrementation of available resources, or ultimately decide that realization of the project is not worthwhile.

The computations and decisions to be made in this Step G are aimed at providing all the alternative solution schedules necessary to a manager confronted with a situation similar to the one outlined above. Evidently, if the execution of the previous steps had generated a feasible solution schedule, we would be publishing only one, recognized, "heuristic best solution schedule".

The following algorithm summarizes the computations and decisions to be made in this Step G:

1. Compare the height of the current RMLP against the height of the given K_i. If the height of the current RMLP is larger than the height of K_i, conclude that the resource constraint is not satisfied and that the current schedule is not a feasible solution to our problem. Then execute rule 3, further below.

   If the height of the current RMLP is equal to or smaller than the height of K_i, conclude that the resource constraint is being satisfied by the current schedule, and that it might give a feasible solution to our problem. Then execute rule 2.

2. Recall the current value of the project slippage (S). If the current value of S is equal to - or less than - zero, conclude that the time constraint is also satisfied by the current schedule, which is therefore our "heuristic best solution schedule"; then publish it and stop all computations.

   If the current value of S is larger than zero, conclude that the time constraint is not satisfied by the current schedule, which is therefore not a feasible solution to our scheduling problem. Then execute rule 3.

3. Publish the current solution schedule because it gives the minimum daily resource allocations for the current t_n, even though it is not a feasible solution to our scheduling problem; also report whether it was failing to satisfy the resource constraint K_i, the time constraint t_d, or both. Then execute rule 4.

4. Compare the current value of S against the value of S_m (maximum project slippage to be allowed for this project).

If the current S is smaller than the given S_m, update the following information:

new current t_n = current $t_n + 1$

new latest time of event p = current latest time of event p + 1

for all events : p = 1,2,3,...,e

new latest start of activity A = current latest start of activity A + 1

new latest finish of activity A = current latest finish of activity A + 1

for all activities : A = 1,2,3,...,a

Then go back to execute Step D (page 37).

If the current S is equal to - or larger than - the given S_m, stop all computations.

The flow chart given in Figure 4 (page 49) summarizes the

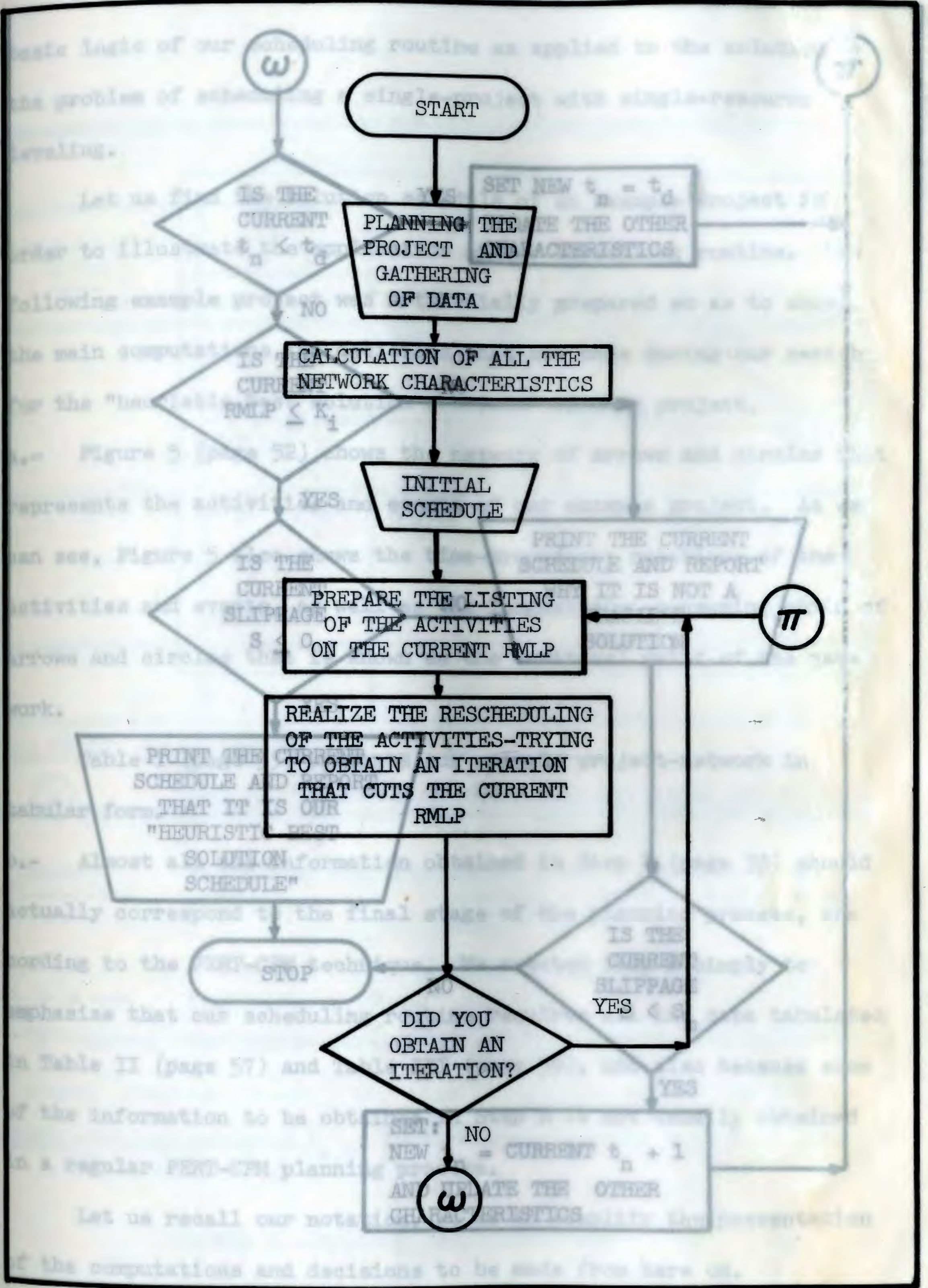

START
PLANNING THE PROJECT AND GATHERING OF DATA
CALCULATION OF ALL THE NETWORK CHARACTERISTICS
INITIAL SCHEDULE
PREPARE THE LISTING OF THE ACTIVITIES ON THE CURRENT RMLP
π
REALIZE THE RESCHEDULING OF THE ACTIVITIES-TRYING TO OBTAIN AN ITERATION THAT CUTS THE CURRENT RMLP
DID YOU OBTAIN AN ITERATION?
YES
NO
ω

FIGURE 4.—FLOW CHART OF THE SCHEDULING ROUTINE

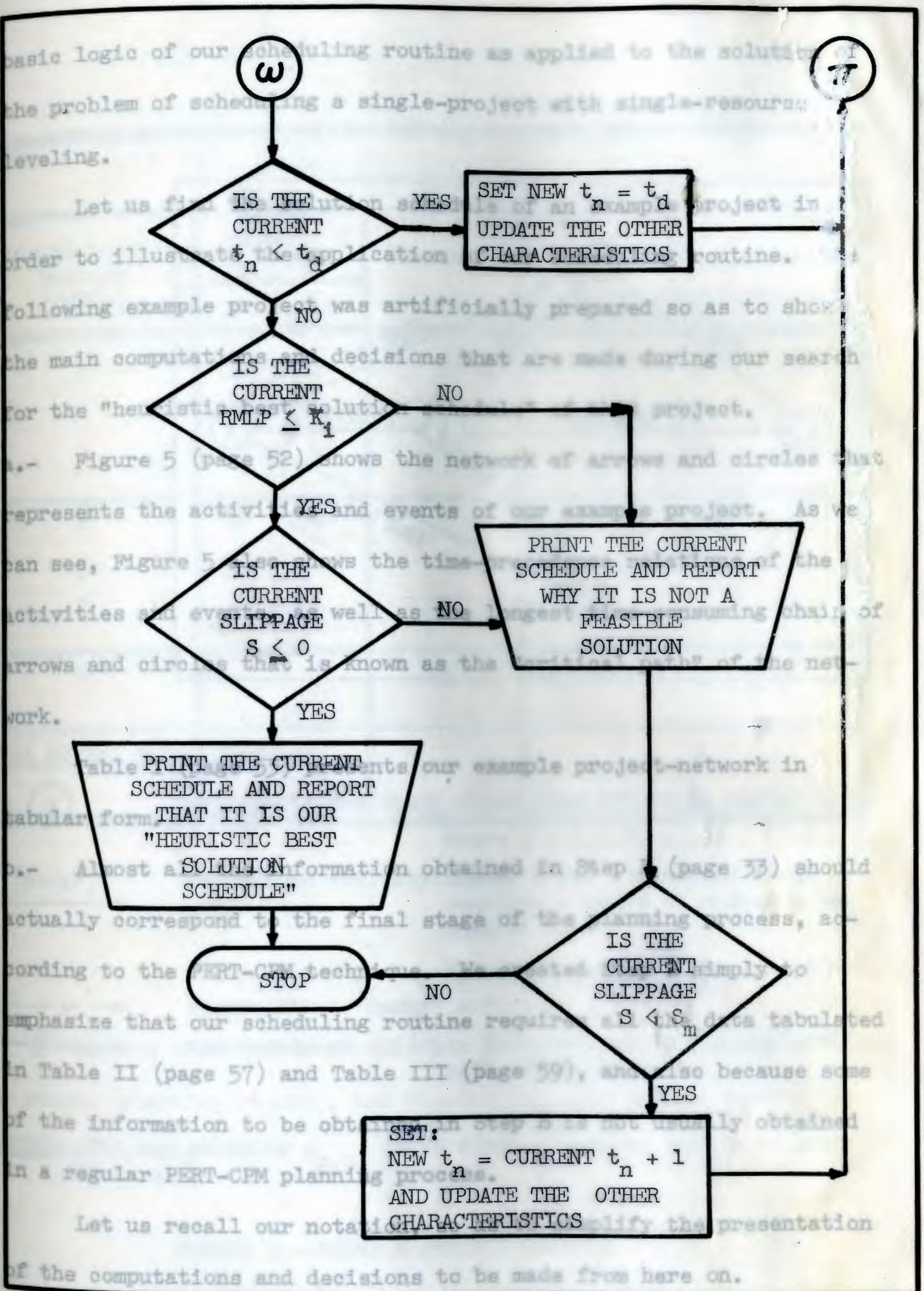

ω
π
IS THE CURRENT $t_n < t_d$
YES
SET NEW $t_n = t_d$ UPDATE THE OTHER CHARACTERISTICS
NO
IS THE CURRENT RMLP $\leq K_i$
NO
YES
PRINT THE CURRENT SCHEDULE AND REPORT WHY IT IS NOT A FEASIBLE SOLUTION
IS THE CURRENT SLIPPAGE $S \leq 0$
NO
YES
PRINT THE CURRENT SCHEDULE AND REPORT THAT IT IS OUR "HEURISTIC BEST SOLUTION SCHEDULE"
STOP
NO
IS THE CURRENT SLIPPAGE $S < S_m$
YES
SET:
NEW t_n = CURRENT $t_n + 1$
AND UPDATE THE OTHER CHARACTERISTICS

FIGURE 4.--FLOW CHART OF THE SCHEDULING ROUTINE (continued)

basic logic of our scheduling routine as applied to the solution of the problem of scheduling a single-project with single-resource leveling.

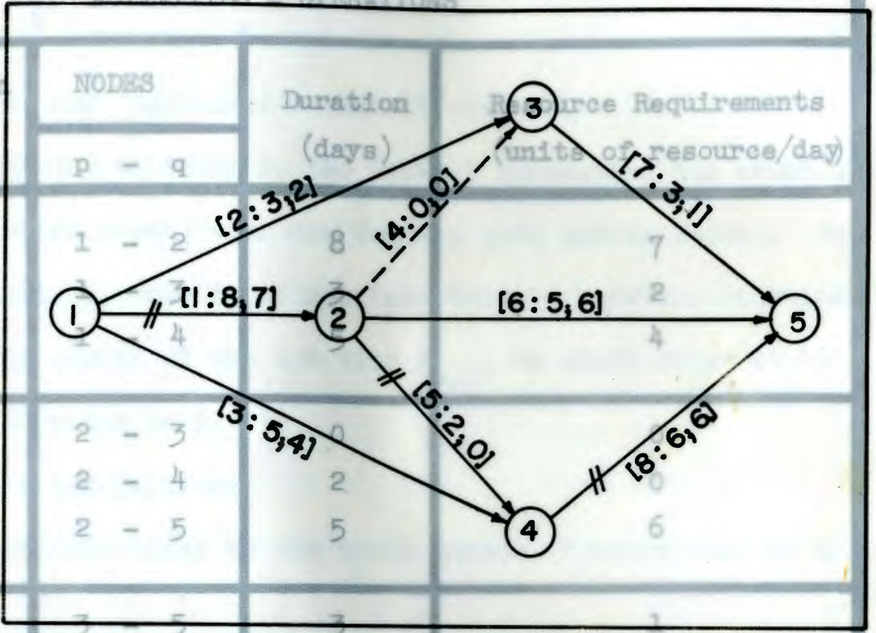
Let us find the solution schedule of an example project in order to illustrate the application of our scheduling routine. The following example project was artificially prepared so as to show the main computations and decisions that are made during our search for the "heuristic best solution schedule" of this project.

a.- Figure 5 (page 52) shows the network of arrows and circles that represents the activities and events of our example project. As we can see, Figure 5 also shows the time-precedence relations of the activities and events, as well as the longest time-consuming chain of arrows and circles that is known as the "critical path" of the network.

Table I (page 53) presents our example project-network in tabular form.

b.- Almost all the information obtained in Step B (page 33) should actually correspond to the final stage of the planning process, according to the PERT-CPM technique. We created Step B simply to emphasize that our scheduling routine requires all the data tabulated in Table II (page 57) and Table III (page 59), and also because some of the information to be obtained in Step B is not usually obtained in a regular PERT-CPM planning process.

Let us recall our notation, so as to simplify the presentation of the computations and decisions to be made from here on.



KEY TO THE NOTATION USED:

○p	represents the event whose identification number is p. Where p = 1,2,3,....,e; e = 5.
[A:d;r] →	means that activity number A has a duration d and requires r units of resource per unit of time. Where A = 1,2,3,....,a; a = 8.
- - - - →	represents a "dummy" activity
—//→	represents an activity that belongs to the critical path of the network.

NOTE: For any activity $A_{p,q}$, it is always required that p be less than q.

FIGURE 5.--EXAMPLE PROJECT-NETWORK

ACTIVITIES = OPERATIONS			
Identification Numbers (AIN's)	NODES	Duration (days)	Resource Requirements (units of resource/day)
	p - q		
1	1 - 2	8	7
2	1 - 3	3	2
3	1 - 4	5	4
4	2 - 3	0	0
5	2 - 4	2	0
6	2 - 5	5	6
7	3 - 5	3	1
8	4 - 5	6	6

$t_d = 24$ days

$S_m = 3$ days

$K_i = 7$ units of resource/day

Observe that the activities are grouped first according to their p events (starting events), and then listed according to their q events (ending events).

TABLE I

DATA OBTAINED IN THE PLANNING STEP

p represents any "event" in the network.

$p = 1,2,3,\ldots,e;$

where e is also equal to the total number of events in the network.

q represents any "successor event" of event p.

$A_{p,q}$ represents any activity in the network connecting the event p (its starting event) with the event q (its ending event). For simplification, when it is not important to know the starting and ending events of the activity $A_{p,q}$, we shall refer to it by the "A" value only.

Then: $A = 1,2,3,\ldots,a;$

where a is also equal to the total number of activities in the network.

$d(A)$ represents the duration of activity A.

$r(A)$ represents the resource requirements of activity A.

$ET(p)$ represents the earliest time that event p can occur.

$LT(p)$ represents the latest time that event p can occur.

$ST(p)$ represents the slack time of event p.

$ES(A)$ represents the earliest start of activity A.

$EF(A)$ represents the earliest finish of activity A.

$LS(A)$ represents the latest start of activity A.

$LF(A)$ represents the latest finish of activity A.

$TS(A)$ represents the total slack of activity A.

$FS(A)$ represents the free slack of activity A.

$IS(A)$ represents the independent slack of activity A.

$SS(A)$ represents the safety slack of activity A.

IM(A) represents the index of mobility of activity A.

t_n' represents the earliest project completion time given by the critical path of the network.

t_d represents the given due date for our project

t_o represents the starting date of our project, or day zero.

PS represents the project slack.

S represents the current project slippage.

S_m represents the maximum project slippage to be allowed for the project.

Then: S is always less than, or equal to, S_m.

K_i represents the maximum level of resources available to any combination of activities during any day.

Let us assume that the values of the following network characteristics had already been obtained in the planning step of our example project, because the setting of these values constitutes the goal of PERT-CPM planning.

Then, the given values are: t_o, t_d, S_m, K_i, t_n', all d(A)'s, all r(A)'s, all ES(A)'s, all EF(A)'s, all LS(A)'s, all LF(A)'s, all ET(p)'s, all LT(p)'s, e, and a.

The slack times of the activities, events, and project can be calculated by the following formulas:

$$ST(p) = LT(p) - ET(p)$$

$$TS(A) = LF(A) - EF(A) = LS(A) - ES(A)$$

$$FS(A_{p,q}) = ET(q) - EF(A_{p,q})$$

$$IS(A_{p,q}) = ET(q) - d(A_{p,q}) - LT(p)$$

$$SS(A_{p,q}) = LS(A_{p,q}) - LT(p)$$

$PS = t_d - t_n'$; if $t_d > t_n'$.

The activity A is a dummy if its d(A) is equal to zero.

The activity A belongs to the critical path of the network, if its TS(A) is zero.

The event p belongs to the critical path of the network, if its ST(p) is zero.

The index of mobility values of the activities were found by following the algorithm given during the presentation of Step B (page 34). However, we will calculate some of these IM values in order to show the application of this algorithm.

1. Set: IM = a = 8

2. The activity that has the largest TS is activity No. 2.

   TS(2) = 10

   Then:

   IM(2) = 8

3. Set: new IM = 8 - 1 = 7

4. Since 7 > 0, return to execute rule 2.

5. The non-indexed activities that have the largest TS values are: No. 3, No. 4, and No. 7.

   TS(3) = 5; TS(4) = 5; TS(7) = 5.

   Of these tied activities, the activities with the largest FS values are: No. 3 and No. 7

   FS(3) = 5; FS(7) = 5; FS(4) = 0.

   Of these tied activities the activity that has the largest IS value is activity No. 3.

   IS(3) = 5; IS(7) = 0.

ACTIVITIES					
Ident. Nos. (AIN's)	Events	Duration	Resource Req's.	Earliest Times	
	p - q			Start	Finish
1	1 - 2	8	7	0	8
2	1 - 3	3	2	0	3
3	1 - 4	5	4	0	5
4	2 - 3	0	0	8	8
5	2 - 4	2	0	8	10
6	2 - 5	5	6	8	13
7	3 - 5	3	1	8	11
8	4 - 5	6	6	10	16

TABLE II

ACTIVITIES

Latest Times		Slack Times				Critical Path	Dummy	Index of Mobility
Start	Finish	Total	Free	Indep.	Safety			
0	8	0	0	0	0	Yes	No	1
10	13	10	5	5	10	No	No	8
5	10	5	5	5	5	No	No	7
13	13	5	0	0	5	No	Yes	5
8	10	0	0	0	0	Yes	No	2
11	16	3	3	3	3	No	No	4
13	16	5	5	0	0	No	No	6
10	16	0	0	0	0	Yes	No	3

NETWORK CHARACTERISTICS (ACTIVITIES)

EVENTS				
Identification Numbers (EIN's)	Times			Critical Path
	Earliest	Latest	Slack	
1	0	0	0	Yes
2	8	8	0	Yes
3	8	13	5	No
4	10	10	0	Yes
5	16	16	0	Yes

$t_n' = 16$ days

$t_d = 24$ days

$PS = 8$ days

$S_m = 3$ days

$K_i = 7$ units of resource/day

$a = 8$ activities

$e = 5$ events

TABLE III

NETWORK CHARACTERISTICS (EVENTS AND PROJECT)

Then:

IM(3) = 7.

6. Set: new IM = 7 - 1 = 6.

7. Since 6 > 0, return to execute rule 2.

8. The non-indexed activities with the largest TS values are: No. 4 and No. 7.

TS(4) = 5; TS(7) = 5.

Of these tied activities, the activity with the largest FS value is activity No. 7.

FS(7) = 5; FS(4) = 0.

Then:

IM(7) = 6.

9. Set: new IM = 6 - 1 = 5

10. Since 5 > 0, return to execute rule 2.

If we continue the assignation of IM values according to the rules of this algorithm, we would find the very same IM values listed in Table II (page 57) under the heading "Index of Mobility". We can see in Table II that the larger IM values correspond to the activities that have positive slack times, while the smaller IM values correspond to the activities on the critical path, which can not be delayed without delaying the current project completion time. The IM values are measuring the mobility of each activity with respect to the mobility of all the others.

c.- The initial schedule for our example project is shown in Figure 6 (page 62), together with its corresponding resource profile. All the activities and events have been located at their earliest times according to the values given in Table II (page 57) and in Table III (page 59); therefore we are using the "early start schedule" of the project. The numbers above the horizontal arrows give the following information for each activity: (1) its identification number A, (2) its duration d, and (3) its resource requirements r; these parameters were put inside the brackets in the following order [A:d;r]. The larger dots represent the location of the events whose identification numbers are written on the lower right hand side of these dots. There should not be any confusion regarding the fact that we have more than one dot with a given number, because an event is a time-point and as such, this point becomes a line perpendicular to the time axis on our two-dimensional graph. We put one dot on each side of the arrows to remember the activities' starting and ending events, and also to represent graphically their slack times. t_o marks the starting date of our project, or the beginning of day one. t_n' marks the "earliest project completion time", which for this schedule is also the "current project completion time" or t_n. t_d marks the given project due date, which is also our time constraint. t_d^* marks the latest due date for our project, in case we are forced to utilize some "project slippage" (S), but only up to the given S_m or "maximum project slippage", in order to satisfy the resource constraint K_i or "maximum availability of resources" marked by a horizontal-dashed-line on the resource profile. The utilization of project slippage was discussed on page 45 of this thesis. PS shows the available project slack, and RMLP points out the "right most largest peak" of our resource profile - the peak we are going to cut by rescheduling one of the activities

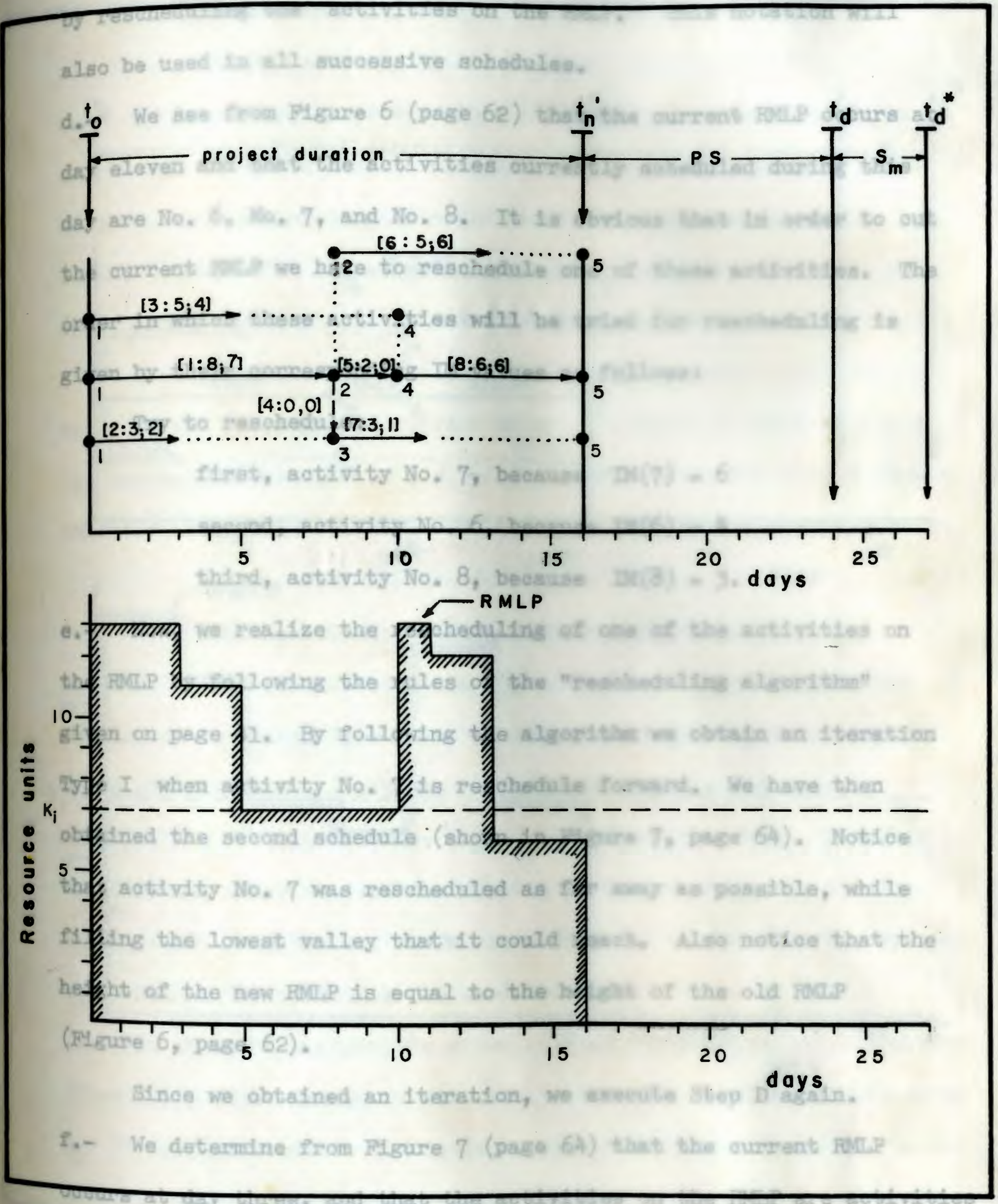


FIGURE 6.--INITIAL SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT

currently scheduled during the same day at which RMLP occurs: e.g., by rescheduling the activities on the RMLP. This notation will also be used in all successive schedules.

d.- We see from Figure 6 (page 62) that the current RMLP occurs at day eleven and that the activities currently scheduled during this day are No. 6, No. 7, and No. 8. It is obvious that in order to cut the current RMLP we have to reschedule one of these activities. The order in which these activities will be tried for rescheduling is given by their corresponding IM values as follows:

Try to reschedule:

first, activity No. 7, because IM(7) = 6

second, activity No. 6, because IM(6) = 4

third, activity No. 8, because IM(8) = 3.

e.- Now, we realize the rescheduling of one of the activities on the RMLP by following the rules of the "rescheduling algorithm" given on page 41. By following the algorithm we obtain an iteration Type I when activity No. 7 is reschedule forward. We have then obtained the second schedule (shown in Figure 7, page 64). Notice that activity No. 7 was rescheduled as far away as possible, while filling the lowest valley that it could reach. Also notice that the height of the new RMLP is equal to the height of the old RMLP (Figure 6, page 62).

Since we obtained an iteration, we execute Step D again.

f.- We determine from Figure 7 (page 64) that the current RMLP occurs at day three, and that the activities on the RMLP are activities No. 1, No. 2, and No. 3. The order for rescheduling is:

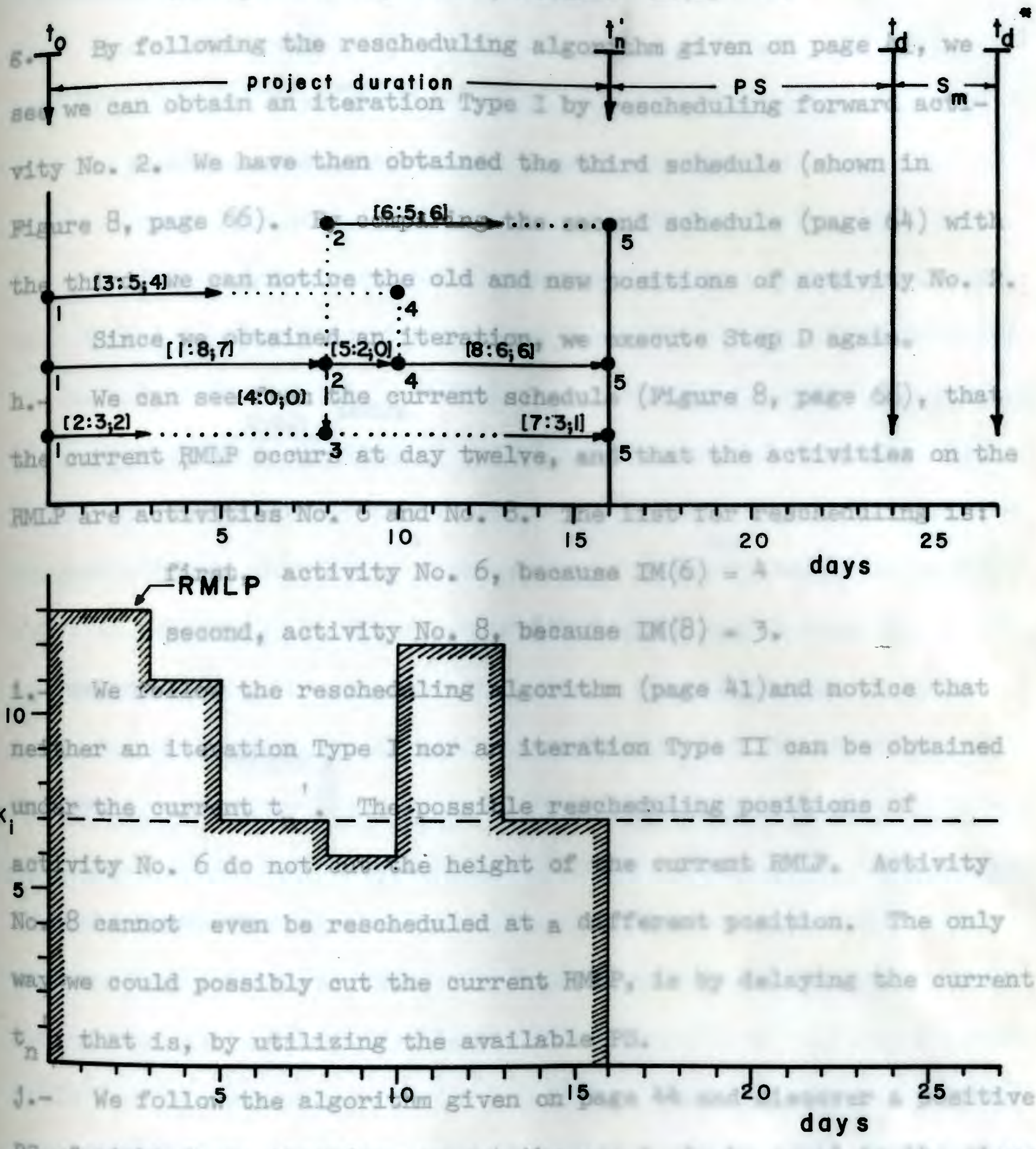


FIGURE 7.--SECOND SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT

first, activity No. 2, because IM(2) = 8

second, activity No. 3, because IM(3) = 7

third, activity No. 1, because IM(1) = 1.

g.- By following the rescheduling algorithm given on page 41, we see we can obtain an iteration Type I by rescheduling forward activity No. 2. We have then obtained the third schedule (shown in Figure 8, page 66). By comparing the second schedule (page 64) with the third, we can notice the old and new positions of activity No. 2.

Since we obtained an iteration, we execute Step D again.

h.- We can see from the current schedule (Figure 8, page 66), that the current RMLP occurs at day twelve, and that the activities on the RMLP are activities No. 6 and No. 8. The list for rescheduling is:

first, activity No. 6, because IM(6) = 4

second, activity No. 8, because IM(8) = 3.

i.- We follow the rescheduling algorithm (page 41)and notice that neither an iteration Type I nor an iteration Type II can be obtained under the current t_n'. The possible rescheduling positions of activity No. 6 do not cut the height of the current RMLP. Activity No. 8 cannot even be rescheduled at a different position. The only way we could possibly cut the current RMLP, is by delaying the current t_n'; that is, by utilizing the available PS.

j.- We follow the algorithm given on page 44 and discover a positive PS of eight days; therefore we set the new t_n to be equal to the given t_d (24 days) and update the latest times of all events, and the latest starting dates and latest finishing dates of all activities, by adding eight days (the value of PS) to them.

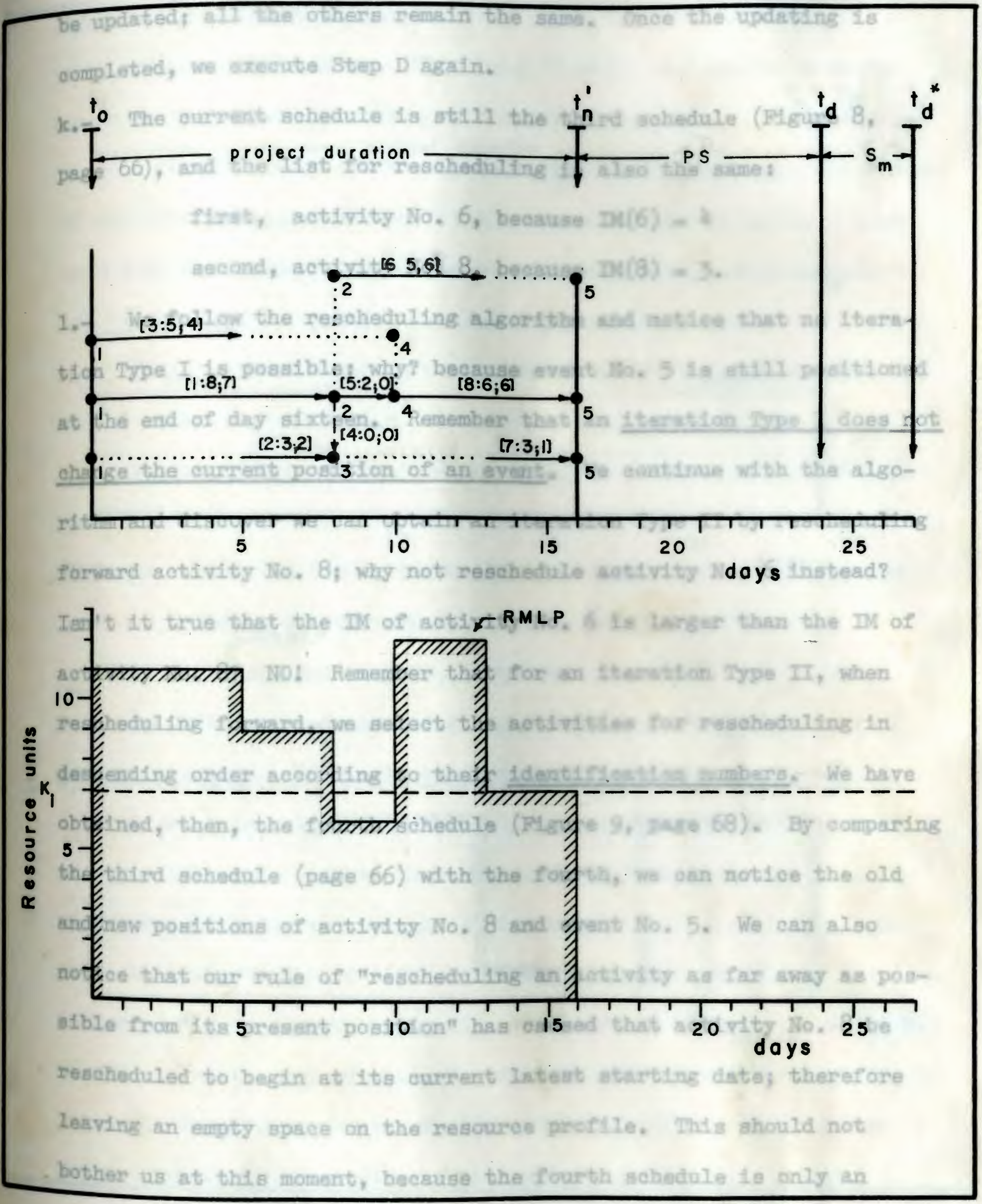


FIGURE 8.--THIRD SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT

These network characteristics are the only ones that need to be updated; all the others remain the same. Once the updating is completed, we execute Step D again.

k.- The current schedule is still the third schedule (Figure 8, page 66), and the list for rescheduling is also the same:

first, activity No. 6, because IM(6) = 4

second, activity No. 8, because IM(8) = 3.

l.- We follow the rescheduling algorithm and notice that no iteration Type I is possible; why? because event No. 5 is still positioned at the end of day sixteen. Remember that an <u>iteration Type I does not change the current position of an event.</u> We continue with the algorithm and discover we can obtain an iteration Type II by rescheduling forward activity No. 8; why not reschedule activity No. 6 instead? Isn't it true that the IM of activity No. 6 is larger than the IM of activity No. 8? NO! Remember that for an iteration Type II, when rescheduling forward, we select the activities for rescheduling in descending order according to their <u>identification numbers.</u> We have obtained, then, the fourth schedule (Figure 9, page 68). By comparing the third schedule (page 66) with the fourth, we can notice the old and new positions of activity No. 8 and event No. 5. We can also notice that our rule of "rescheduling an activity as far away as possible from its present position" has caused that activity No. 8 be rescheduled to begin at its current latest starting date; therefore leaving an empty space on the resource profile. This should not bother us at this moment, because the fourth schedule is only an intermediate one. If we re-read rule 18 of our rescheduling algo-

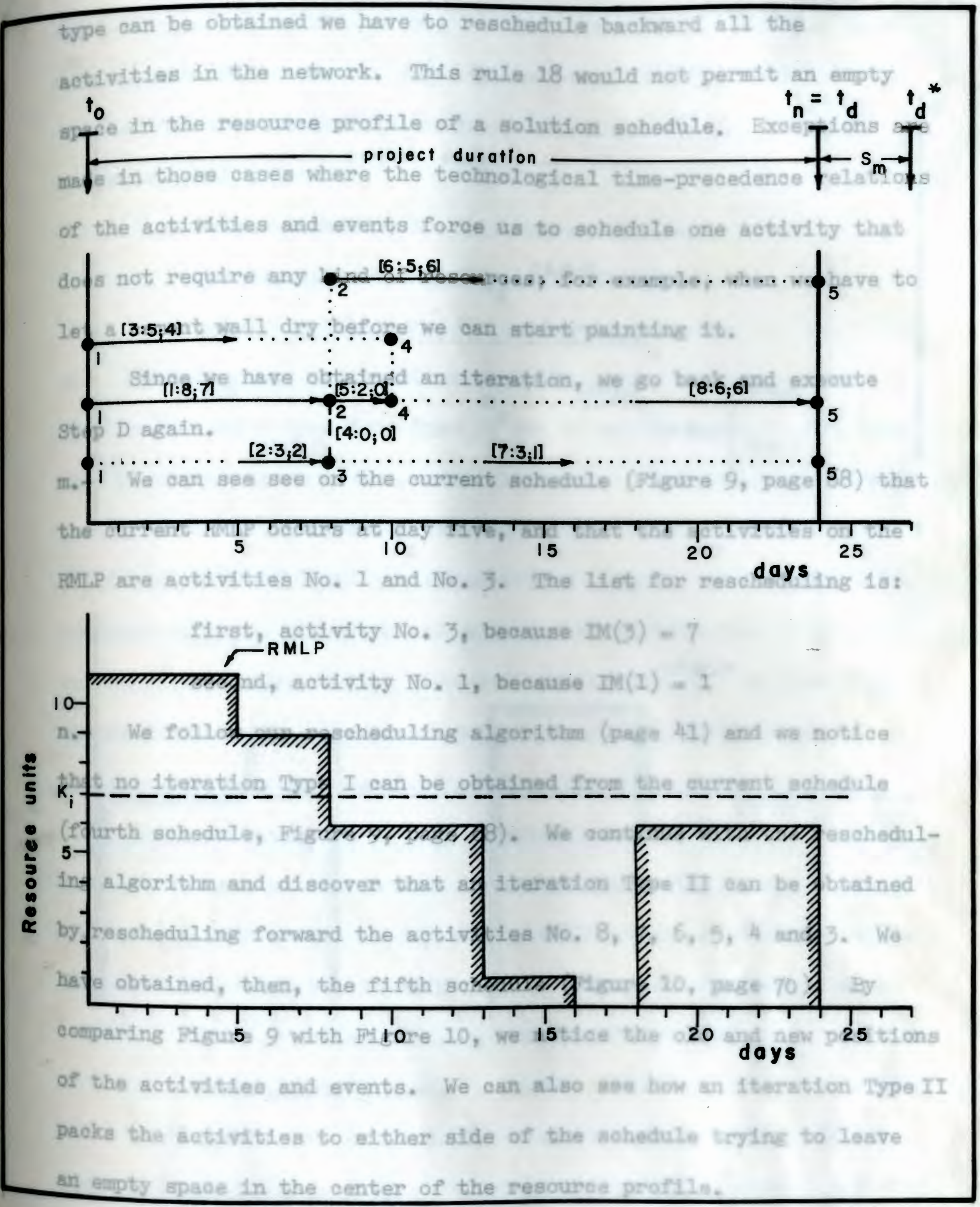


FIGURE 9.--FOURTH SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT

rithm (page 41), we will remember that, when no iteration of either type can be obtained we have to reschedule backward all the activities in the network. This rule 18 would not permit an empty space in the resource profile of a solution schedule. Exceptions are made in those cases where the technological time-precedence relations of the activities and events force us to schedule one activity that does not require any kind of resources; for example, when we have to let a cement wall dry before we can start painting it.

Since we have obtained an iteration, we go back and execute Step D again.

m.- We can see see on the current schedule (Figure 9, page 68) that the current RMLP occurs at day five, and that the activities on the RMLP are activities No. 1 and No. 3. The list for rescheduling is:

first, activity No. 3, because IM(3) = 7

second, activity No. 1, because IM(1) = 1

n.- We follow our rescheduling algorithm (page 41) and we notice that no iteration Type I can be obtained from the current schedule (fourth schedule, Figure 9, page 68). We continue with the rescheduling algorithm and discover that an iteration Type II can be obtained by rescheduling forward the activities No. 8, 7, 6, 5, 4 and 3. We have obtained, then, the fifth schedule (Figure 10, page 70). By comparing Figure 9 with Figure 10, we notice the old and new positions of the activities and events. We can also see how an iteration Type II packs the activities to either side of the schedule trying to leave an empty space in the center of the resource profile.

Since we have obtained an iteration, we return and execute

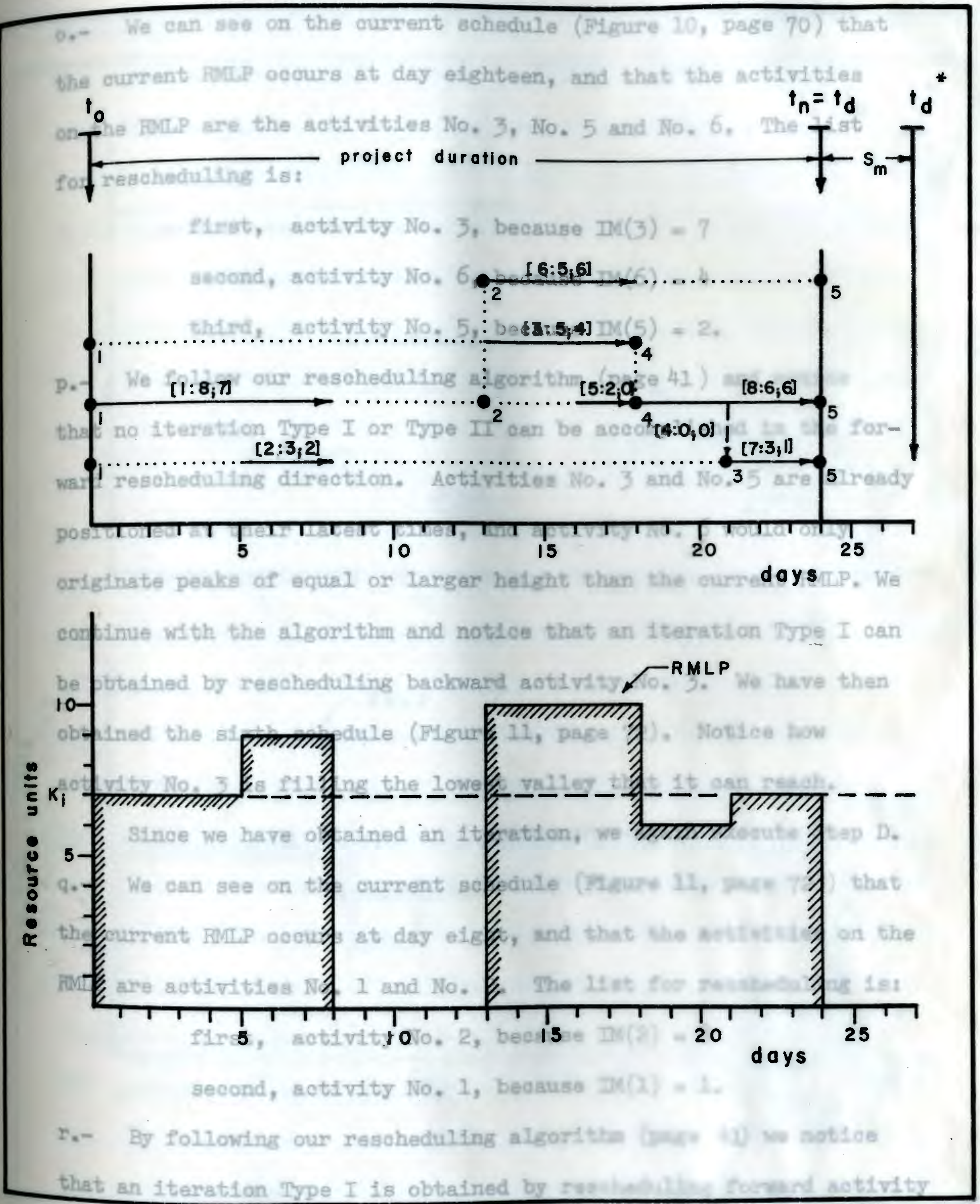


FIGURE 10.--FIFTH SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT
]

Step D once again.

o.- We can see on the current schedule (Figure 10, page 70) that the current RMLP occurs at day eighteen, and that the activities on the RMLP are the activities No. 3, No. 5 and No. 6. The list for rescheduling is:

first, activity No. 3, because IM(3) = 7

second, activity No. 6, because IM(6) = 4

third, activity No. 5, because IM(5) = 2.

p.- We follow our rescheduling algorithm (page 41) and notice that no iteration Type I or Type II can be accomplished in the forward rescheduling direction. Activities No. 3 and No. 5 are already positioned at their latest times, and activity No. 6 would only originate peaks of equal or larger height than the current RMLP. We continue with the algorithm and notice that an iteration Type I can be obtained by rescheduling backward activity No. 3. We have then obtained the sixth schedule (Figure 11, page 72). Notice how activity No. 3 is filling the lowest valley that it can reach.

Since we have obtained an iteration, we again execute Step D.

q.- We can see on the current schedule (Figure 11, page 72) that the current RMLP occurs at day eight, and that the activities on the RMLP are activities No. 1 and No. 2. The list for rescheduling is:

first, activity No. 2, because IM(2) = 8

second, activity No. 1, because IM(1) = 1.

r.- By following our rescheduling algorithm (page 41) we notice that an iteration Type I is obtained by rescheduling forward activity No. 2. We have then obtained the seventh schedule (Figure 12, page 73).

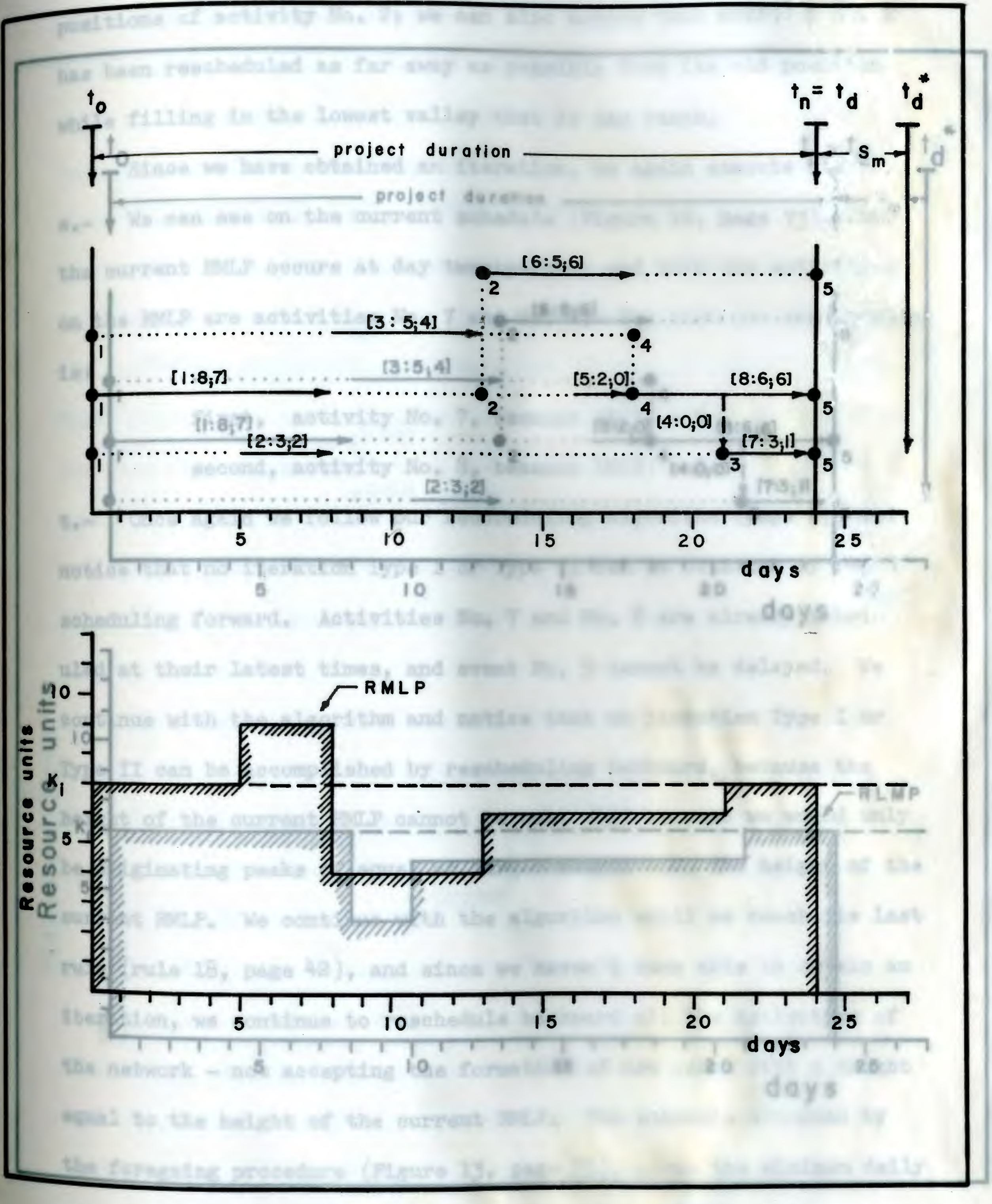


FIGURE 11.--SIXTH SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT

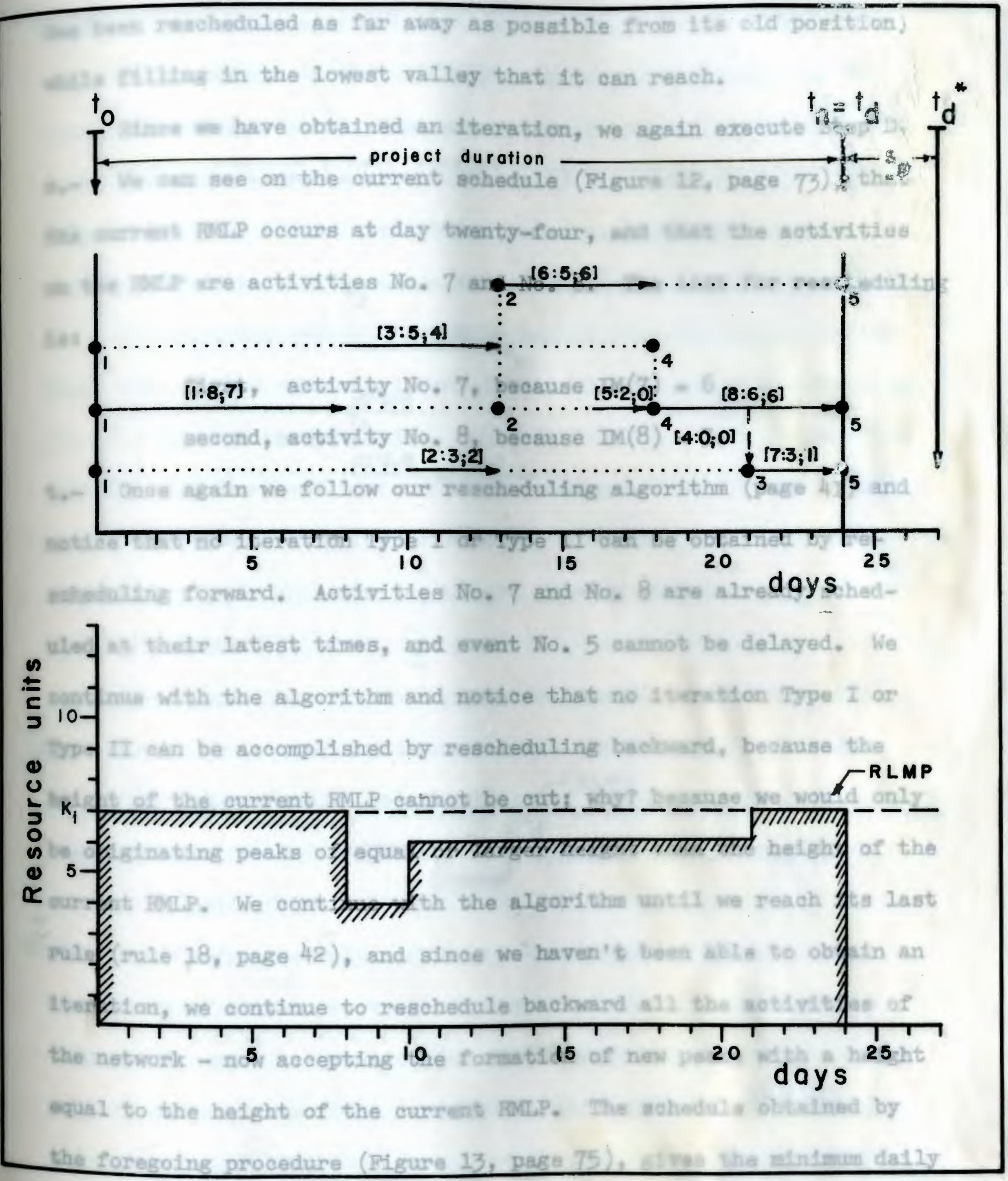


FIGURE 12.--SEVENTH SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT

By comparing Figure 11 with Figure 12 we can notice the old and new positions of activity No. 2; we can also notice that activity No. 2 has been rescheduled as far away as possible from its old position, while filling in the lowest valley that it can reach.

Since we have obtained an iteration, we again execute Step D.

s.- We can see on the current schedule (Figure 12, page 73), that the current RMLP occurs at day twenty-four, and that the activities on the RMLP are activities No. 7 and No. 8. The list for rescheduling is:

first, activity No. 7, because IM(7) = 6

second, activity No. 8, because IM(8) = 3.

t.- Once again we follow our rescheduling algorithm (page 41) and notice that no iteration Type I or Type II can be obtained by rescheduling forward. Activities No. 7 and No. 8 are already scheduled at their latest times, and event No. 5 cannot be delayed. We continue with the algorithm and notice that no iteration Type I or Type II can be accomplished by rescheduling backward, because the height of the current RMLP cannot be cut; why? because we would only be originating peaks of equal or larger height than the height of the current RMLP. We continue with the algorithm until we reach its last rule (rule 18, page 42), and since we haven't been able to obtain an iteration, we continue to reschedule backward all the activities of the network - now accepting the formation of new peaks with a height equal to the height of the current RMLP. The schedule obtained by the foregoing procedure (Figure 13, page 75), gives the minimum daily resource allocations for the current project completion date

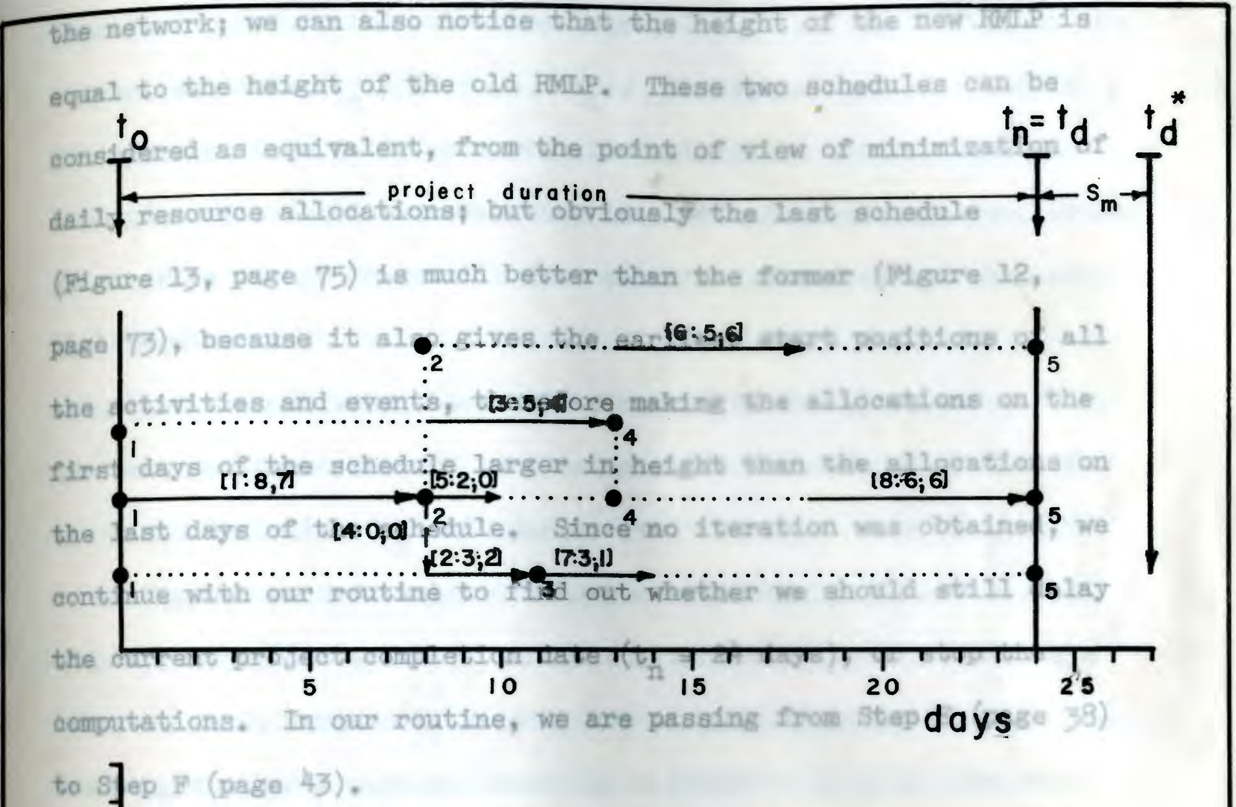


FIGURE 13.--HEURISTIC BEST SOLUTION SCHEDULE AND RESOURCE PROFILE - EXAMPLE PROJECT

(current t_n = 24 days). By comparing Figure 12 with Figure 13, we notice the old and new positions of all the activities and events of the network; we can also notice that the height of the new RMLP is equal to the height of the old RMLP. These two schedules can be considered as equivalent, from the point of view of minimization of daily resource allocations; but obviously the last schedule (Figure 13, page 75) is much better than the former (Figure 12, page 73), because it also gives the earliest start positions of all the activities and events, therefore making the allocations on the first days of the schedule larger in height than the allocations on the last days of the schedule. Since no iteration was obtained, we continue with our routine to find out whether we should still delay the current project completion date (t_n = 24 days), or stop the computations. In our routine, we are passing from Step E (page 38) to Step F (page 43).

u.- We execute the algorithm given on page 44 , and notice that the current PS is equal to zero (because the current t_n = 24 days = given t_d). Therefore we set the value of the current project slippage (S), as follows:

current S = - PS = 0.

Next, we execute the rules of Step G (explained on page 45).

v.- Looking at the resource profile of our current schedule (Figure 13, page 75), we can see that the height of the current RMLP (7 units of resource) is equal to the height of the maximum resource availability (K_i, marked by the horizontal-dashed-line); therefore we conclude that the resource constraint is being satisfied by the current schedule. We recall the value of the current project slippage (S) and notice that it is equal to zero; we therefore conclude that the time constraint is also satisfied by the current schedule (t_n = 24 days = t_d). Since the current schedule gives the minimum daily resource allocations for the project, and also gives the earliest completion date for all activities and events, we can name it our "heuristic best solution schedule" and then stop the computations.

Let us make some observations regarding the given values for the resource constraint (K_i = 7 units of resource per day), and for the time constraint (t_d = 24 days).

Suppose we had assigned a larger value to K_i (K_i = 8,9,...). The heuristic best solution schedule given in Figure 12 would have been the same, because K_i is only one of our two yardsticks for deciding whether a current schedule is feasible or not. The other yardstick is the value given for t_d.

Suppose that we had assigned a smaller value to K_i (K_i = 6,5,4,... units of resource). The schedule given in Figure 13 would not satisfy this smaller K_i and therefore would not give a feasible solution. Then, our routine would tell us to start using project slippage - one day at a time - trying to obtain a further minimization of the height of the current RMLP. For our example project, we would have obtained the very same schedule (Figure 13, page 75), because the height of the current RMLP cannot be reduced; why? because activity No. 1 alone requires seven units of resource per day. If we look at the schedule in Figure 13, we can easily see that the current RMLP could be cut by allowing one day of project slippage (PS) (activities No. 6 and No. 8 would be rescheduled one day later), but we can see that the height of the new RMLP (which would occur at day eight) would be equal to the height of the old RMLP (the one presently occuring at day fourteen). The only way that we could possibly reduce the height of this new RMLP would be by extending (as opposed to "crashing") the duration of activity No. 1 by X days until this activity No. 1 can be accomplished with only six units of resource per day.

Even though the computations made during the entire execution of our scheduling routine are simple and straightforward, we must recognize that they are lengthy and too numerous for hand calculations. A computer program written in FORTRAN IV for the IBM-360 computer was used to make practical the utilization of our scheduling routine in real-life projects. This computer program realizes all the computations and decisions that compose our scheduling routine. It is given in Appendix A of this thesis. The example problem just explained was fed to the computer and the total computer time utilized was 1.26 minutes (1.13 minutes for reading and compiling the program and the data, and .13 minutes for the execution of the calculations). The data-input for this example project, as well as the computer output, is given in Appendix B of this thesis.

A small real-life project (construction of a gas station) composed of 58 activities and 36 events, was also fed to the computer and the total computer time utilized was 12.73 minutes (1.16 minutes for reading and compiling the program and the data, and 11.57 minutes for the execution of the calculations). Its computer solution is given in Appendix C. The data-input is not given because it follows the same format as the data-input of Appendix B, and also because all the data-input is contained in the computer output, and can be reconstructed by the interested reader if he wishes to do so.

The printing format of this computer program was prepared to handle networks with up to 70 activities and 70 events. For larger networks this printing format can easily be transformed by an experienced programmer, in order to obtain the schedule on a tabular list rather than the pictorial tabulation of the schedule which we are providing. The size of the project that this computer program can handle will then be limited only by the memory-storage capacity of the computer.

It is pointed out that our scheduling routine can cope with the so-called "splitting" of activities (activities that may be interrupted and then continued at a later date without extra cost), simply by dividing each "splittable" activity into the required number of sub-activities and sub-events. These sub-activities and sub-events will be incorporated in the data as though they were real activities and events of the original project network. Then, when the "heuristic best solution schedule" is obtained, it will determine whether these splittable activities should be done continuously or with successive interruptions. The original time-precedence relations of the original activities and events remain intact.

It is also pointed out that our scheduling routine can handle those activities that do not have a constant requirement of resources during their entire durations: for example, an activity A may have a duration of 2 days and require 4 units of resource during its first day, and 3 units of resource during its second day. Obviously, to keep track of these "non-constant resource requirements" of the activities represents a larger amount of computations and extra care in the preparation of the resource profiles; but we can see that the scheduling routine is not changed.

IV. GENERALIZED PROJECT SCHEDULING PROBLEM

Let us now contemplate the adaptation procedures for our scheduling routine, so that it can cope with more complicated cases of project scheduling with resource constraints. In the following discussions we will assume that the reader is already familiar with the rules and decisions explained in the steps of the "scheduling routine" just presented in Chapter III.

Single-Project Multi-resource Case

We can consider here, that two different situations could occur:

a. The several kinds of resources required by the activities of the project-network can be weighted and reduced to units of a single common resource. Then, we would be solving again, the single-project single-resource case; and therefore the "heuristic best solution schedule" can also be found by following the scheduling routine given in Chapter III. This situation may arise in projects for which the several kinds of resources can be transformed and expressed in units of money, area, weight, etc., and where the minimization of this single common resource has priority. Projects realized by government agencies usually have the constraint of a fixed-budgeted-periodical availability of dollars, and obviously, even if the availability of men and equipment can be easily satisfied by the market, we have to constrain our expenses to the fixed budget. A repair-project in a long bridge would typify the constraints on total area, or total weight, of the several kinds of resources needed for the repair-project.

Suppose that the given project requires g kinds of resources $(i = 1,2,3,\ldots,g)$. Then, the resource requirements of each activity can be represented by the sum of its weighted individual resource requirements as follows:

$$r(A_j) = \sum_{i=1}^{g} C_i z_{ij}$$

where:

j is the activity identification number.

C_i is the cost-weighting factor for resource i

z_{ij} is the requirement of resource i of activity A_j during any day of its duration.

C_i and $z_{ij} \geq 0$.

Then, the resource constraint can be formulated as follows:

$$S_t = \sum_{j=1}^{a} X_{jt}\, r(A_j) \leq K^*$$

where:

t represents any day of the schedule.

X_{jt} represents a zero-one variable that takes the value of one when the activity A_j is currently scheduled during day t, otherwise it takes the value of zero.

a is equal to the total number of activities in the network.

S_t is the total allocation of resources during day t.

K^* is the maximum availability of money, or area, or weight, etc.

$r(A_j)$ is as previously defined.

b. The second situation is perhaps the most commonly found in real life; in any case, it is more general. It happens when the several kinds of resources required by the activities of the project-network are so different in nature (or maybe management prefers to keep them separate), that they cannot be weighted and reduced to units of a single common resource. In this situation, the mathematical formulation can be transformed as follows:

Let:

The vector $\overline{r(A_j)}$ represents the set of daily resource requirements of activity A_j.

$$\overline{r(A_j)} = (z_{1j}, z_{2j}, \dots, z_{ij}, \dots, z_{gj})$$

where:

A_j, z_{ij}, and g are as previously defined.

The vector $\overline{S}_t$ represents the set of allocations of the g kinds of resources during day t.

$$\overline{S}_t = \left(\sum_{j=1}^{a} X_{jt}\, z_{1j},\ \sum_{j=1}^{a} X_{jt}\, z_{2j}, \dots,\ \sum_{j=1}^{a} X_{jt}\, z_{ij}, \dots,\ \sum_{j=1}^{a} X_{jt}\, z_{gj} \right)$$

where:

a, X_{jt}, z_{ij}, and g are as previously defined.

The vector $\overline{K}$ represents the set of maximum daily availabilities of the g kinds of resources

$$\overline{K} = (K_1, K_2, \ldots, K_i, \ldots, K_g)$$

where:

K_i is the maximum daily availability of resource i.

g is as previously defined.

Then, the resource constraint can be formulated as follows:

$\overline{S}_t \leq \overline{K}$; during any day t.

The adaptation of the scheduling routine given in Chapter III will be accomplished by making the following changes:

1. Provide one resource profile for each of the g kinds of resources.
2. Obtain information from management about the "order of priority" in which the g kinds of resources will be minimized - which kind is the "most important", which one is the "second-most important", and so on. If no "order of priority" is given, assign the "order of priority" at random.
3. Realize the rescheduling of the activities by cutting the RMLP of the resource profile of the "most important" kind of resources in exactly the same way as explained in Chapter III; but do not accept the formation of peaks of larger height than the height of the current largest peak in each one of the resource profiles of the other "g-1" kinds of resources; e.g.: accept only the

formation of peaks of equal or smaller height than the height of the current larger peak in each one of the resource profiles of the other g-1 kinds of resources.

4. All the other heuristic rules and decisions of the scheduling routine given in Chapter III remain the same.

Multi-Project Single-Resource Case

This case happens when several projects require the same kind of resources i which are only available in a fixed amount.

K_i (maximum availability of resource i - during any day of the schedule - to any combination of activities belonging to one or more projects).

For example: a maintenance department, or a shop, that has several projects in its backlog. Each one of these projects requires the same kind of special equipment - or the same kind of highly trained personnel - which is only available in a limited number. The problem is to realize as many projects as possible with the existing resources, and to determine the completion date of each project. This "leveling" problem is very similar to the "allocation" problem. The difference is the objective function of the scheduling procedures. The allocation procedures are aimed at minimizing the project completion dates, while the leveling procedures are aimed at minimizing the daily resource allocations to the given projects; any project completion date could be accepted, as long as it does not go beyond the given due date.

The scheduling routine presented in Chapter III could be used to solve this case by finding first the best heuristic schedule of each individual project by using the parameter K_i (as previously defined) as the resource constraint. The scheduling routine of Chapter III is applied without any modification to each of these single project single resource sub-cases. Then, we would consider each one of these projects as if they were super-activities of a super-project. The resource requirements of these super-activities are given by the resource profile of their corresponding heuristic best solution schedules. All the super-activities would have the day zero as their earliest start times; their corresponding completion times would give their durations; and their corresponding due dates would give their latest finish times. The resource constraint of the super-project is given by K_i (as defined before), and the time constraint is given by the due date of the super-activity that happens to have the largest due date among all these super-activities. The maximum slippage for the super-project is given by the maximum slippage of the super-activity that happens to have the largest due date among all of these super-activities. We would finally obtain the "heuristic best solution schedule" of the super-project by following the heuristic rules and decisions of the scheduling routine given in Chapter III.

Multi-Project Multi-resource Case

This case happens when a certain industrial concern needs to realize several different projects that require several different kinds of resources which are subjected to availability constraints. The problem is to realize as many projects as possible with the existing resources and to determine the completion date of each project. The completion dates should be set, preferably, within the corresponding due dates.

We can also consider here that two situations could occur:

a. A situation where the several kinds of resources required by the activities of the several project-networks can be weighted and reduced to units of a single common resource (money, area, weight, etc.). Then, we would be solving again the multi-project single-resource case, explained immediately preceding.

b. A situation where the several kinds of resources required by the activities of the several project-networks cannot be weighted and reduced to units of a single common resource.

The scheduling routine presented in Chapter III could also be used to solve this case, by first finding the best heuristic schedule of each individual project by using the vector $\overline{K}$ (defined on page 84) as the resource constraint, and then by following the procedures outlined on page 84 for the adaptation of the scheduling routine to solve the single-project multi-resource case in a situation where the several kinds of resources cannot be weighted and reduced to a single common kind of resource such as money, weight, etc. Then, we would consider each one of these projects as if they were super-activities of a super-project as previously explained on page 86, with the difference that the resource constraint is now given by the vector $\overline{K}$ to represent the set of maximum daily availabilities of the several kinds of resources.

$$\overline{K} = (K_1, K_2, \ldots, K_i, \ldots, K_z)$$

where K_i and g are as previously defined.

We would finally obtain the best heuristic schedule of the super-project by following the heuristic rules and decisions of the scheduling routine given in Chapter III as adapted by the instructions given on page 84 for the solution of the single-project multi-resource case in a situation where the several kinds of resources cannot be weighted and reduced to a single common kind of resource.

The preceding adaptations of our scheduling routine do give a solution to each one of these more complicated cases of project scheduling with resource constraints; but, obviously, more research is necessary before we could assume that the heuristic solutions of the multi-project cases are also practical and workable. These heuristic solutions of the multi-project cases should be taken only as valuable suggestions for the final development of multi-project scheduling techniques in some future research.

V. CONCLUSIONS

The scheduling routine presented and discussed in this thesis will find real-life applications in the construction industry and in the maintenance functions of other industrial concerns where the scheduling of the activities of a single-project under constraints on availability of resources and due date is the scheduling problem, and the minimization of the daily allocations of resources is the objective of the scheduling procedures.

The scheduling routine provides practical and workable means of solving the leveling problem of project scheduling with resource constraints for the single-project single-resource case, and for the single-project multi-resource case. It allows for "splitting" and "non-constant resource requirements" of the activities. The practicality of the scheduling routine is enhanced by the fact that it can be programmed for computer operation, therefore releasing the user from the lengthy, repetitive, and tedious calculations associated with problems of this type. A computer program that realizes all the computations and prints out the solution schedules of the single-project single-resource case is given in Appendix A. The Appendices B and C show computer solutions of two examples.

Two extensions of the scheduling routine that could solve the multi-project single-resource case, and the multi-project multi-resource case of the leveling problem of project scheduling with resource constraints are presented as suggestions to a future researcher interested in the solution of these cases. However, more research should be done in order to provide practical and workable solutions for these cases.

Several analytical formulations of varying problems of project scheduling with resource constraints have been published; but, as the authors themselves recognize, they are impractical and usually unworkable means of solving real-life problems. They are important, however, from the conceptual point of view. We present in this thesis an integer linear programming formulation of the leveling problem of project scheduling with resource constraints which could yield the optimal solution, but which is almost impossible to solve. Therefore it does not provide workable means of solving the problem.

The literature review summarizes the variety of heuristic approaches that have been proposed for the solution of different problems of project scheduling with resource constraints. The available publications are mainly concerned with the description of the fundamental heuristics of these scheduling techniques, their goals, and the scope of the problems they are aimed at solving, but very little information, if any, is given about the details of their scheduling procedures. All this precluded a possible comparison of our scheduling routine with the other scheduling techniques. It is emphasized, however, that we tried to incorporate in our scheduling routine as many as possible of those apparently good procedures of the other techniques, and that we supplemented them with new procedures of our own that seemed to yield better results in our specific scheduling problem.

We have shown in the solution of the example project that our scheduling routine originates successive schedules that converge towards the optimal solution schedule: e.g., the feasible schedule that gives the minimum daily allocations of resources. Therefore, we can conclude that it provides a good chance of finding a near-optimal solution schedule, if not the real optimal one. Due to the heuristic nature of the rules and decisions that compose our scheduling routine, we cannot assure the optimality of the solution schedules that it produces. Optimal solution schedules can be assured only by strict analytical techniques. However, since the last word is not yet in on problems of project scheduling with resource constraints, it is believed that new heuristic scheduling techniques based on functional properties of the networks will supplement those already available toward the final development of a practical and workable analytic scheduling technique.

VI. REFERENCES

1. Berman, E. B., "Resource Allocation in a PERT Network under Continuous Time-Cost Functions", Management, July 1964.

2. Bigelow, C. G., "Bibliography on Project Planning and Control by Network Analysis: 1959-1961", Operations Research, Vol. 10, No. 5, 1962.

3. Black, O. J., "An Algorithm for Resource Leveling in Project Networks", unpublished paper, Department of Industrial Administration, Yale University, May 1965.

4. Bower, J. L., "Managing the Resource Allocation Process", Graduate School of Business Administration, Harvard University, Boston, Mass., 1970.

5. Bowman, E. H., "The Schedule-Sequencing Problem", Operations Research, Vol. 7, No. 5, Sept.-Oct. 1959.

6. Buffa, E. S., "Production Inventory Systems - Planning and Control", Part VI, "Planning and Scheduling for Large-Scale Projects", Richard D. Irwin Inc., Homewood, Ill. 1968.

7. Burgess, A. R., and Killebrew, J. B., "Variation in Activity Level on a Cyclic Arrow Diagram", Journal of Industrial Engineering, March-April 1962.

8. Clark, C. E., "The Optimum Allocation of Resources among the Activities of a Network", Journal of Industrial Engineering, Vol. XII, No. 1, Jan.-Feb. 1961.

9. Conway, R. W., "An Experimental Investigation of Priority Assignment in a Job Shop", Memorandum R.M.-3789-P.R., the Rand Corporation, February 1964.

10. Davis, E. W., "Resource Allocation in Project Network Models - A Survey", Journal of Industrial Engineering, Vol. XVII, No. 4, April 1966.

11. ________., and Heidorn, G. E., "An Algorithm for Optimal Project Scheduling under Multiple Resource Constraints", Management Science, Vol. 17, No. 12, August 1971.

12. De Witte, L., "Manpower Leveling of PERT Networks", Data Processing for Science/Engineering, March-April 1964.

13. Eisner, H., "A Generalized Network Approach to the Planning and Scheduling of a Research Program", Operations Research, Vol. 9, No. 3, 1961.

14. Fendley, L. G., "The Development of a Complete Multiproject Scheduling System Using a Forecasting and Sequencing Technique", Ph.D. Dissertation, Arizona State University, Tempe, Arizona, September 1966.

15. ________., "Toward the Development of a Complete Multiproject Scheduling System", Journal of Industrial Engineering, Vol. XIX, No. 10, October 1968.

16. Fulkerson, D. R., "Scheduling in Project Networks", Memorandum R.M.-4137-P.R., the Rand Corporation, June, 1964.

17. Ghare, P. M., "Optimal Resource Allocation in Activity Networks", Paper, Operations Research Society of America Meeting, Houston, Texas, November 1965.

18. Heuser, W. A., Jr., and Wynne, B. E., Jr., "An Application of the Critical Path Method to Job-Shop Scheduling - A Case Study", Management Technology, Vol. 3, No. 2, December 1963.

19. Hooper, P. C., "Resource Allocation and Levelling", Proceedings of the Third CPA Symposium, Operations Research Society, London, 1965.

20. Horowitz, J., "Critical Path Scheduling - Management Control through CPM and PERT", the Ronald Press Company, New York, 1967.

21. Hu, T. C., "Parallel Sequencing and Assembly Line Problems", Operations Research, Vol. 9, No. 6, Nov.-Dec., 1961.

22. Johnson, T. J. R., "An Algorithm for the Resource-Constrained Project Scheduling Problem", unpublished Ph.D. Thesis, School of Management, M.I.T., August 1967.

23. Kelley, J. E., Jr., "The Critical-Path Method: Resources Planning and Scheduling",Chapter 21 in "Industrial Scheduling" ed. by Math J.F. and Thompson, G.L., Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1963.

24. Lambourn, S., "Resource Allocation and Multiproject Scheduling (RAMPS), A New Tool in Planning and Control", Computer Journal, Vol. 5, No. 4, January 1963.

25. Levy, F. K., Thompson, G. S., and Wiest, J. D., "Multi-Ship Multi-Shop Workload Smoothing Program", Naval Research Logistics Quarterly, March 1963.

26. ________., and Wiest, J. D., "A Management Guide to PERT/CPM" Prentice Hall Inc., Englewood Cliffs, New Jersey, 1969.

27. Martino, R. L., "Resource Allocation and Scheduling", Project Management and Control Series, Vol. III, New York: American Management Association, 1965.

28. McGee, A. A., and Markarian, M. D., "Optimum Allocation of Research/Engineering Manpower within a Multi-Project Organizational Structure", IEEE Transactions on Engineering Management, September 1962.

29. Meyer, W. L., and Shaffer, L. R., "Extensions of the Critical Path Method through the Application of Integer Programming" Department of Civil Engineering, University of Illinois, July 1963.

30. ________., Ritter, J. B., and Shaffer, L. R., "The Critical Path Method", McGraw-Hill Book Company, New York, 1965.

31. Mize, J. H., "A Heuristic Scheduling Model for Multiproject Organizations", Ph.D. Thesis, Purdue University, Lafayette, Indiana, August 1964.

32. Moder, J. J., and Phillips, C. R., "Project Management with CPM and PERT", Reinhold Corporation, New York, 1964.

33. Moshman, J., Johnson, J., and Larsen, M., "RAMPS: A Technique for Resource Allocation and Multiproject Scheduling", Proceedings, Spring Joint Computer Conference, 1963.

34. Muth, J. F., Thompson, G. L., and Winters, P. R., Editors, "Industrial Scheduling", Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1963.

35. Pascoe, T. L., "An Experimental Comparison of Heuristic Methods for Allocating Resources", Ph.D. Thesis, Cambridge University, Engineering Department, 1965.

36. Phillips, C. R., "Fifteen Key Features of Computer Programs for CPM and PERT", Journal of Industrial Engineering, Jan.-Feb. 1964.

37. Waldron, A. J., "Applied Principles of Project Planning and Control", published by A. James Waldron, 371 Kings Highway West, Haddonfield, N. J., April 1966.

38. Wiest, J. D., "The Scheduling of Large Projects with Limited Resources", Ph.D. Thesis, Carnegie Institute of Technology, Pittsburgh, Pennsylvania, 1963.

39. ________., "Some Properties of Schedules for Large Projects with Limited Resources", Operations Research, Vol. 12, No. 3, May-June 1964.

40. ________., "Heuristic Programs for Decision Making", Harvard Business Review, Sept.-Oct. 1966.

41. ________., "A Heuristic Model for Scheduling Large Projects with Limited Resources", Management Science, Vol. 13, No. 6, February 1967.

42. Wilson, R. C., "Assembly Line Balancing and Resource Scheduling", University of Michigan Summer Conference on Production and Inventory Control, 1964.

APPENDIX A

COMPUTER PROGRAM OF THE SCHEDULING ROUTINE

```
C
C     HEURISTIC SCHEDULING ROUTINE
C     PROJECT SCHEDULING WITH RESOURCE CONSTRAINTS
C     SINGLE PROJECT SINGLE RESOURCE CASE OF THE LEVELING PROBLEM
C     MASTER OF SCIENCE THESIS ---  HECTOR J ALVAREZ
C     UNIVERSITY OF RHODE ISLAND --- 1971
C
      COMMON KINODE(70),KJNODE(70),KTIME(70),KRSRCE(70),NET(70)
      COMMON KEF(70),NLT(70),KLS(70),KLF(70),KFLOAT(70),KFREE(70)
      COMMON KINDEP(70),KSAFE(70),NSLACK(70),KRANK(70),LEVEL(80)
      COMMON KCHOIS(30),KLU,LMAX,MAXTU,NOTA,NAINLM,MMOVER,MAXRES
      COMMON YES,SIGNO,YNO,BLANCO,PALO,MSCHED(70,80),SINOP
      COMMON NA,NN,NLAST,NSLIP,NTIME,LSLIP,LTIME,NOSCH,MSALE,INCRE
      COMMON KES(70),KPOS(30),KPOSRK(30)
      DIMENSION KKESTA(70),KKEFIN(70),NNESTA(70),NSCHED(70,80)
      DIMENSION KESTA(70),KEFIN(70),NESTA(70)
      EQUIVALENCE (KFREE(1),KESTA(1)),(KINDEP(1),KEFIN(1))
      EQUIVALENCE (KSAFE(1),NESTA(1))
      NOTA=0
      LLMAX=10000
      CALL CALNET
   49 CALL REPROF
   50 MSALE=0
      CALL SCHEDU
      CALL PEAK
      CALL TORITE
      SINOP=0
      IF(MSALE) 60,60,49
   60 CALL MOVER
      IF(MMOVER) 70,70,50
   70 CALL TOLEFT
      IF(MSALE) 80,80,49
   80 CALL MOVEL
      IF(MMOVER) 90,90,50
   90 SINOP=SINOP+1
      IF(SINOP-2) 60,100,100
  100 NOTA=0
      NOSCH=NOSCH-1
      CALL SCHEDU
      IF(LMAX-LLMAX) 110,120,120
  110 DO 111 KO=1,NA
      KKESTA(KO)=KESTA(KO)
      KKEFIN(KO)=KEFIN(KO)
      NNESTA(KO)=NESTA(KO)
      DO 112 NON=1,NTIME
      NSCHED(KO,NON)=MSCHED(KO,NON)
  112 CONTINUE
  111 CONTINUE
```

```
      NNTIME=NTIME
      NNOSCH=NOSCH-1
      MMAXTU=MAXTU
      LLMAX=LMAX
  120 IF(NTIME-LTIME) 124,121,121
  121 IF(LMAX-MAXRES) 126,126,122
  122 IF(NSLIP-LSLIP) 123,125,125
  123 NSLIP=NSLIP+1
  124 NTIME=LTIME+NSLIP
      DO 160 NO=1,NN
      NLT(NO)=NLT(NO)+INCRE
  160 CONTINUE
      INCRE=1
      GO TO 49
  125 WRITE(6,900) MAXRES,LTIME,LSLIP
  900 FORMAT('1'//T10,'THE ROUTINE CANNOT FIND AN SCHEDULE WHICH SATISFI
     +ES THE RESOURCE CONSTRAINT OF LMAX=',I3,' . '//T10,'THE BEST HEURI
     1STIC SCHEDULE FOR A GIVEN PROJECT DUE TIME OF ',I3,' UNITS OF TIME
     2'//T10,'AND WITH A MAXIMUM PROJECT SLIPPAGE OF ',I3,' UNITS OF TIM
     3E'//T10,'IS THE FOLLOWING : '/)
      GO TO 127
  126 WRITE(6,910)
  910 FORMAT('1'//////////T10,'THE HEURISTIC BEST SOLUTION SCHEDULE FOR
     +THE GIVEN PROJECT IS :')
  127 DO 141 KO=1,NA
      KESTA(KO)=KKESTA(KO)
      KEFIN(KO)=KKEFIN(KO)
      NESTA(KO)=NNESTA(KO)
      DO 142 NON=1,NTIME
      IF(NON-NNTIME) 146,146,145
  145 MSCHED(KO,NON)=0
      GO TO 142
  146 MSCHED(KO,NON)=NSCHED(KO,NON)
  142 CONTINUE
  141 CONTINUE
      NTIME=NNTIME
      NOSCH=NNOSCH
      MAXTU=MMAXTU
      LMAX=LLMAX
      NOTA=0
      CALL REPROF
      CALL SCHEDU
      STOP
      END
      SUBROUTINE CALNET
C
C
C   CALCULATES NETWORK CHARACTERISTICS
C
C
      COMMON KINODE(70),KJNODE(70),KTIME(70),KRSRCE(70),NET(70)
      COMMON KEF(70),NLT(70),KLS(70),KLF(70),KFLOAT(70),KFREE(70)
      COMMON KINDEP(70),KSAFE(70),NSLACK(70),KRANK(70),LEVEL(80)
      COMMON KCHOIS(30),KLU,LMAX,MAXTU,NOTA,NAINLM,MMOVER,MAXRES
      COMMON YES,SIGNO,YNO,BLANCO,PALO,MSCHED(70,80),SINOP
      COMMON NA,NN,NLAST,NSLIP,NTIME,LSLIP,LTIME,NCSCH,MSALE,INCRE
      COMMON KES(70),KPOS(30),KPOSRK(30)
      READ(5,8500) YES,SIGNO,YNO,BLANCO,PALO
C
C   READS DATA INPUT
C
      READ(5,9000) NA,LTIME,LSLIP,MAXRES
      DO 1000 NO=1,NA
 1000 READ(5,9000) KINODE(NO),KJNODE(NO),KTIME(NO),KRSRCE(NO)
C
C   CALCULATES EARLY START AND EARLY FINISH FOR ALL ACTIVITIES
C   ALSO CALCULATES EARLY TIMES FOR ALL NODES
C
      NN=KJNODE(NA)
      NLAST=0
      DO 1001 NNO=1,NN
 1001 NET(NNO)=0
      DO 1002 NO=1,NA
      I=KINODE(NO)
      J=KJNODE(NO)
      KES(NO)=NET(I)
      KEF(NO)=KES(NO)+KTIME(NO)
      IF(KEF(NO)-NET(J)) 2,2,1
    1 NET(J)=KEF(NO)
    2 IF(KEF(NO)-NLAST) 1002,1002,3
    3 NLAST=KEF(NO)
 1002 CONTINUE
C
C   CALCULATES LATEST START AND LATEST FINISH FOR ALL ACTIVITIES
C   ALSO CALCULATES LATE TIMES FOR ALL NODES
C
      DO 1003 K=1,NN
 1003 NLT(K)=NLAST
      DO 1004 NO=1,NA
      L=NA+1-NO
      I=KINODE(L)
      J=KJNODE(L)
      KLF(L)=NLT(J)
      KLS(L)=KLF(L)-KTIME(L)
      IF(KLS(L)-NLT(I)) 4,5,5
    4 NLT(I)=KLS(L)
    5 KFLOAT(L)=KLS(L)-KES(L)
 1004 CONTINUE
C
C  PRINTS NODES TABLE
C  CALCULATES SLACK TIMES FOR ALL NODES
C
      WRITE(6,9001)
      DO 1005 K=1,NN
      NSLACK(K)=NLT(K)-NET(K)
      IF(NSLACK(K)) 6,6,7
    6 WRITE(6,9002) K,NET(K),NLT(K),NSLACK(K),YES
      GO TO 1005
    7 WRITE(6,9002) K,NET(K),NLT(K),NSLACK(K),YNO
 1005 CONTINUE
C
C  PRINTS ACTIVITIES TABLE
C  CALCULATES FLOAT TIMES FOR ALL ACTIVITIES
C
      WRITE(6,9003)
      DO 1006 K=1,NA
      KFREE(K)=0
      KINDEP(K)=0
      KSAFE(K)=0
      IF(KFLOAT(K)) 1006,1006,8
    8 I=KINODE(K)
      J=KJNODE(K)
      KFREE(K)=NET(J)-KEF(K)
      KINDEP(K)=NET(J)-KTIME(K)-NLT(I)
      IF(KINDEP(K)) 9,10,10
    9 KINDEP(K)=0
   10 KSAFE(K)=KLS(K)-NLT(I)
 1006 CONTINUE
C
C ASSIGNS IM VALUES TO ALL ACTIVITIES
C
      KR=1
      DO 1007 K=1,NA
      KRANK(K)=0
      IF(KFLOAT(K)) 11,11,1007
   11 KRANK(K)=KR
      KR=KR+1
 1007 CONTINUE
      DO 1008 K=1,NA
   12 KAC=K
```

```
      IF(KRANK(K)) 14,14,1008
   14 K1=K+1
      DO 1009 KJ=K1,NA
      IF(KRANK(KJ)) 1009,15,1009
   15 IF(KFLOAT(KAC)-KFLOAT(KJ)) 1009,16,19
   16 IF(KFREE(KAC)-KFREE(KJ)) 1009,17,19
   17 IF(KINDEP(KAC)-KINDEP(KJ)) 1009,18,19
   18 IF(KSAFE(KAC)-KSAFE(KJ)) 1009,1009,19
   19 KAC=KJ
 1009 CONTINUE
      KRANK(KAC)=KR
      KR=KR+1
      IF(KRANK(K)) 12,12,1008
 1008 CONTINUE
C
C FINDS OUT WHICH ACTIVITIES LIE ON THE CRITICAL PATH
C ALSO FINDS OUT WHICH ACTIVITIES ARE DUMMIES
C
      DO 1010 K=1,NA
      IF(KTIME(K)) 20,20,23
   20 IF(KFLOAT(K)) 21,21,22
   21 WRITE(6,9004) K,KINODE(K),KJNODE(K),KRSRCE(K),KTIME(K),KES(K),
     +KEF(K),KLS(K),KLF(K),KFLOAT(K),KFREE(K),KINDEP(K),KSAFE(K),
     1YES,YES,KRANK(K)
      GO TO 1010
   22 WRITE(6,9004) K,KINODE(K),KJNODE(K),KRSRCE(K),KTIME(K),KES(K),
     +KEF(K),KLS(K),KLF(K),KFLOAT(K),KFREE(K),KINDEP(K),KSAFE(K),
     1YNO,YES,KRANK(K)
      GO TO 1010
   23 IF(KFLOAT(K)) 24,24,25
   24 WRITE(6,9004) K,KINODE(K),KJNODE(K),KRSRCE(K),KTIME(K),KES(K),
     +KEF(K),KLS(K),KLF(K),KFLOAT(K),KFREE(K),KINDEP(K),KSAFE(K),
     1YES,YNO,KRANK(K)
      GO TO 1010
   25 WRITE(6,9004) K,KINODE(K),KJNODE(K),KRSRCE(K),KTIME(K),KES(K),
     +KEF(K),KLS(K),KLF(K),KFLOAT(K),KFREE(K),KINDEP(K),KSAFE(K),
     1YNO,YNO,KRANK(K)
 1010 CONTINUE
C
C  CALCULATES GENERAL INFORMATION FOR THE PROJECT
C
      NOSCH=0
      NSLIP=0
      NTIME=NLAST
      DO 1011 K=1,NA
      KFREE(K)=KES(K)
      KINDEP(K)=KEF(K)
 1011 CONTINUE
```

```
      DO 1012 N=1,NN
      KSAFE(N)=NET(N)
 1012 CONTINUE
      IF(NTIME-LTIME) 1016,1016,1015
 1015 LTIME=NTIME
 1016 INCRE=LTIME-NTIME
      WRITE(6,9005) NTIME,LTIME,INCRE,MAXRES,LSLIP
C
C     FORMATS FOR INPUT-OUTPUT
C
 8500 FORMAT(5A4)
 9000 FORMAT(4I3)
 9001 FORMAT(1H1/T24,' N O D E S = E V E N T S'///T10,'NUMBER',T20,
     +'EARLIEST TIME',T38,'LATEST TIME',T52,'SLACK',T62,'CRITICAL PATH'
     1//)
 9002 FORMAT(T11,I3,T25,I3,T42,I3,T53,I3,T67,1A4/)
 9003 FORMAT(1H1/T50,'A C T I V I T I E S'///T8,'NUMBER',T15,'INODE',
     +T21,'JNODE',T28,'RSRCE',T35,'DURA',T41,'E-START',T48,' E-FIN',
     1T56,'L-START',T64,' L-FIN',T72,'TOTAL-F',T80,' FREE-F',T88,
     2' IND-F',T96,'SAFE-F',T103,'C-PATH',T110,'DUMMY',T116,'INDEX'/)
 9004 FORMAT(T6,3I6,3I7,T45,7I8,T104,1A4,T110,1A4,T117,I3/)
 9005 FORMAT(///T12,'THE EARLIEST PROJECT COMPLETION TIME IS ',I5,' UNIT
     +S OF TIME'/T12,'THE GIVEN PROJECT DUE DATE IS',I5,' UNITS OF TIME'
     1/T12,'THE CURRENT PROJECT SLACK IS',I5,' UNITS OF TIME'/T12,'THE D
     2AILY RESOURCE AVAILABILITY IS',I5,' UNITS OF RESOURCE'/T12,'THE MA
     3XIMUM PROJECT SLIPPAGE IS',I5,' UNITS OF TIME'/)
      RETURN
      END
      SUBROUTINE REPROF
C
C
C     CALCULATION OF THE RESOURCE PROFILE
C
C
      COMMON KINODE(70),KJNODE(70),KTIME(70),KRSRCE(70),NET(70)
      COMMON KEF(70),NLT(70),KLS(70),KLF(70),KFLOAT(70),KFREE(70)
      COMMON KINDEP(70),KSAFE(70),NSLACK(70),KRANK(70),LEVEL(80)
      COMMON KCHOIS(30),KLU,LMAX,MAXTU,NOTA,NAINLM,MMOVER,MAXRES
      COMMON YES,SIGNO,YNO,BLANCO,PALO,MSCHED(70,80),SINOP
      COMMON NA,NN,NLAST,NSLIP,NTIME,LSLIP,LTIME,NCSCH,MSALE,INCRE
      COMMON KES(70),KPOS(30),KPOSRK(30)
      DIMENSION KESTA(70),KEFIN(70),NESTA(70)
      EQUIVALENCE (KFREE(1),KESTA(1)),(KINDEP(1),KEFIN(1))
      EQUIVALENCE (KSAFE(1),NESTA(1))
      DO 1100 M=1,NTIME
      LEVEL(M)=0
 1100 CONTINUE
      DO 1101 MACT=1,NA
      MT1=KESTA(MACT)+1
      MT2=KESTA(MACT)+KTIME(MACT)
      DO 1101 MTIME=1,NTIME
      IF(KTIME(MACT)) 105,105,102
  102 IF(MTIME-MT1) 105,104,103
  103 IF(MTIME-MT2) 104,104,105
  104 MSCHED(MACT,MTIME)=KRSRCE(MACT)
      GO TO 106
  105 MSCHED(MACT,MTIME)=0
  106 LEVEL(MTIME)=LEVEL(MTIME)+MSCHED(MACT,MTIME)
 1101 CONTINUE
      RETURN
      END
      SUBROUTINE SCHEDU
C
C
C CALCULATION AND PRINT OUT OF CURRENT SCHEDULE
C THE CORRESPONDING RESOURCE PROFILE IS ALSO PRINTED
C
C
      COMMON KINODE(70),KJNODE(70),KTIME(70),KRSRCE(70),NET(70)
      COMMON KEF(70),NLT(70),KLS(70),KLF(70),KFLOAT(70),KFREE(70)
      COMMON KINDEP(70),KSAFE(70),NSLACK(70),KRANK(70),LEVEL(80)
      COMMON KCHOIS(30),KLU,LMAX,MAXTU,NOTA,NAINLM,MMOVER,MAXRES
      COMMON YES,SIGNO,YNO,BLANCO,PALO,MSCHED(70,80),SINOP
      COMMON NA,NN,NLAST,NSLIP,NTIME,LSLIP,LTIME,NOSCH,MSALE,INCRE
      COMMON KES(70),KPOS(30),KPOSRK(30)
      DIMENSION KESTA(70),KEFIN(70),NESTA(70),SON(70)
      EQUIVALENCE (KFREE(1),KESTA(1)),(KINDEP(1),KEFIN(1))
      EQUIVALENCE (KSAFE(1),NESTA(1))
      NOSCH=NOSCH+1
      IF(NOTA) 150,150,160
  150 WRITE(6,9101) NOSCH,NA
      NT1=NTIME+1
      DO 1106 MTIME=1,NT1
      M=MTIME-1
C
C MARKS THE ENDING TIMES OF THE ACTIVITIES
C
      DO 1102 K=1,NA
      IF(KEFIN(K)-M) 108,107,108
  107 SON(K)=PALO
      GO TO 1102
  108 SON(K)=BLANCO
 1102 CONTINUE
      WRITE(6,9103) (SON(K),K=1,NA)
      NO=0
C
C  MARKS THE CURRENT LOCATIONS OF THE NODES
C
      DO 1103 N=1,NN
      IF(NESTA(N)-M) 1103,109,1103
  109 NO=NO+1
      SON(NO)=N*1.
 1103 CONTINUE
      IF(NO) 120,120,119
  119 WRITE(6,9102) (SON(N),N=1,NO)
C
C  MARKS THE START TIMES OF THE ACTIVITIES
C
  120 DO 1104 K=1,NA
```

```
      IF(KESTA(K)-M) 111,110,111
  110 SON(K)=PALO
      GO TO 1104
  111 SON(K)=BLANCO
 1104 CONTINUE
      WRITE(6,9103) (SON(K),K=1,NA)
      IF(NT1-MTIME) 1106,1106,112
C
C   PRINTS THE CURRENT SCHEDULE AND IT'S CORRESPONDING
C   RESOURCE PROFILE
C
  112 WRITE(6,9104) MTIME,(MSCHED(K,MTIME),K=1,NA)
      WRITE(6,9105) LEVEL(MTIME)
 1106 CONTINUE
      NOTA=50
C
C   FORMATS
C
 9101 FORMAT(1H1/T32,'S C H E D U L E   N U M B E R',I4///
     +T20,'ACTIVITIES : 1,2,3,........,',I2/T8,'TIME-UNIT',T114,'LEVEL'/)
 9102 FORMAT(T12,'NODES LOCATED HERE ARE :',20F3.0)
 9103 FORMAT(T16,80A1)
 9104 FORMAT(T9,I4,T16,80I1)
 9105 FORMAT('+',T114,I4)
  160 RETURN
      END
      SUBROUTINE PEAK
C
C
C    FINDS OUT THE LOCATION OF THE RIGHT-MOST-LARGEST-PEAK (MAXTU)
C    AND SAVES IT'S VALUE (LMAX)
C
C
      COMMON KINODE(70),KJNODE(70),KTIME(70),KRSRCE(70),NET(70)
      COMMON KEF(70),NLT(70),KLS(70),KLF(70),KFLOAT(70),KFREE(70)
      COMMON KINDEP(70),KSAFE(70),NSLACK(70),KRANK(70),LEVEL(80)
      COMMON KCHOIS(30),KLU,LMAX,MAXTU,NOTA,NAINLM,MMOVER,MAXRES
      COMMON YES,SIGNO,YNO,BLANCO,PALO,MSCHED(70,80),SINOP
      COMMON NA,NN,NLAST,NSLIP,NTIME,LSLIP,LTIME,NCSCH,MSALE,INCRE
      COMMON KES(70),KPOS(30),KPOSRK(30)
      DIMENSION KESTA(70),KEFIN(70),NESTA(70)
      EQUIVALENCE (KFREE(1),KESTA(1)),(KINDEP(1),KEFIN(1))
      EQUIVALENCE (KSAFE(1),NESTA(1))
      LMAX=0
      DO 1200 M=1,NTIME
      L=LEVEL(M)
      IF(L-LMAX) 1200,200,200
  200 LMAX=L
      MAXTU=M
 1200 CONTINUE
C
C    DETERMINES WHICH ACTIVITIES CONTRIBUTE TO BUILD UP LMAX
C    AND SAVES THEIR IDENTIFICATION NUMBERS (KPOS(K)) ,
C    AND THEIR INDEX OF MOBILITY NUMBERS (KPOSRK(K)) .
C
      N=0
      DO 1201 K=1,NA
      MT1=KESTA(K)+1
      MT2=KEFIN(K)
      IF(MT2-MT1) 1201,201,201
  201 IF(MT1-MAXTU) 202,202,1201
  202 IF(MT2-MAXTU) 1201,203,203
  203 N=N+1
      KPOS(N)=K
      KPOSRK(N)=KRANK(K)
 1201 CONTINUE
      KLU=0
  205 K1=KPOSRK(1)
      NCHOI=1
C
C    DETERMINES THE ORDER OF RESCHEDULING
C
      DO 1202 NO=1,N
      KR=KPOSRK(NO)
      IF(KR-K1) 1202,1202,206
 206  K1=KPOSRK(NO)
      NCHOI=NO
1202  CONTINUE
      KLU=KLU+1
      KCHOIS(KLU)=KPOS(NCHOI)
      KPOSRK(NCHOI)=0
      KPOS(NCHOI)=0
      IF(N-KLU) 207,207,205
 207  NAINLM=N
      RETURN
      END
      SUBROUTINE TORITE
C
C
C  ITERATION TYPE I --- FORWARDS
C
C
      COMMON KINODE(70),KJNODE(70),KTIME(70),KRSRCE(70),NET(70)
      COMMON KEF(70),NLT(70),KLS(70),KLF(70),KFLOAT(70),KFREE(70)
      COMMON KINDEP(70),KSAFE(70),NSLACK(70),KRANK(70),LEVEL(80)
      COMMON KCHOIS(30),KLU,LMAX,MAXTU,NOTA,NAINLM,MMOVER,MAXRES
      COMMON YES,SIGNO,YNO,BLANCO,PALO,MSCHED(70,80),SINOP
      COMMON NA,NN,NLAST,NSLIP,NTIME,LSLIP,LTIME,NOSCH,MSALE,INCRE
      COMMON KES(70),KPOS(30),KPOSRK(30)
      DIMENSION KESTA(70),KEFIN(70),NESTA(70)
      EQUIVALENCE (KFREE(1),KESTA(1)),(KINDEP(1),KEFIN(1))
      EQUIVALENCE (KSAFE(1),NESTA(1))
      N=NAINLM
C
C  ATTEMPTS TO RESCHEDULE THE ACTIVITIES AND REPORTS
C  WHETHER THE ATTEMPT WAS SUCCESSFUL OR NOT
C
      DO 1203 NO=1,N
      K=KCHOIS(NO)
      J=KJNODE(K)
      NJT=NESTA(J)
      IF(NJT-KEFIN(K)) 1203,1203,208
  208 MDIF=NJT-KEFIN(K)
      KMDIF=MAXTU-KESTA(K)
      IF(MDIF-KMDIF) 1203,209,209
  209 NSTEP=MDIF-KMDIF+1
      LL3=LMAX
      DO 1204 MSTEP=1,NSTEP
      MXT1=MAXTU+MSTEP
      MXT2=MXT1-1+KTIME(K)
      LL2=0
      DO 1205 M9=MXT1,MXT2
      LEV=LEVEL(M9)+KRSRCE(K)
      IF(LEV-LMAX) 210,1204,1204
  210 IF(LEV-LL2) 1205,1205,211
  211 LL2=LEV
 1205 CONTINUE
      IF(LL2-LL3) 212,212,1204
  212 LL3=LL2
      MHAPN=MSTEP
 1204 CONTINUE
      IF(LL3-LMAX) 214,1203,1203
  214 KESTA(K)=MAXTU+MHAPN-1
      KEFIN(K)=KESTA(K)+KTIME(K)
      MSALE=100
      NAINLM=0
      GO TO 215
 1203 CONTINUE
  215 RETURN
      END
      SUBROUTINE TOLEFT
C
C
C   ITERATION TYPE I --- BACKWARDS
C
C
      COMMON KINODE(70),KJNODE(70),KTIME(70),KRSRCE(70),NET(70)
      COMMON KEF(70),NLT(70),KLS(70),KLF(70),KFLOAT(70),KFREE(70)
      COMMON KINDEP(70),KSAFE(70),NSLACK(70),KRANK(70),LEVEL(80)
      COMMON KCHOIS(30),KLU,LMAX,MAXTU,NOTA,NAINLM,MMOVER,MAXRES
      COMMON YES,SIGNO,YNO,BLANCO,PALO,MSCHED(70,80),SINOP
      COMMON NA,NN,NLAST,NSLIP,NTIME,LSLIP,LTIME,NOSCH,MSALE,INCRE
      COMMON KES(70),KPOS(30),KPOSRK(30)
      DIMENSION KESTA(70),KEFIN(70),NESTA(70)
      EQUIVALENCE (KFREE(1),KESTA(1)),(KINDEP(1),KEFIN(1))
      EQUIVALENCE (KSAFE(1),NESTA(1))
C
C   ATTEMPTS TO RESCHEDULE THE ACTIVITIES AND REPORTS
C   WHETHER THE ATTEMPT WAS SUCCESSFUL OR NOT
C
      DO 1501 NO=1,NAINLM
      K=KCHOIS(NO)
      I=KINODE(K)
      NIT=NESTA(I)
      KST=KESTA(K)
      IF(NIT-KST) 501,1501,1501
  501 MDIF=KST-NIT
      KMDIF=KEFIN(K)+1-MAXTU
      IF(MDIF-KMDIF) 1501,502,502
  502 NSTEP=MDIF-KMDIF+1
      LL1=LMAX
      DO 1502 MSTEP=1,NSTEP
      MXT2=MAXTU-MSTEP
      MXT1=MXT2-KTIME(K)+1
      LL2=0
      DO 1503 M8=MXT1,MXT2
      M9=MXT2-M8+MXT1
      LEV1=LEVEL(M9)
      IF(LEV1-LMAX) 503,1502,1502
  503 IF(M9-KESTA(K)) 505,505,504
  504 LEV1=LEV1-KRSRCE(K)
  505 LEV=LEV1+KRSRCE(K)
      IF(LEV-LMAX) 506,1502,1502
  506 IF(LEV-LL2) 1503,1503,507
  507 LL2=LEV
 1503 CONTINUE
      IF(LL2-LL1) 508,508,1502
  508 LL1=LL2
```

```
      MHAPN=MSTEP
 1502 CONTINUE
      IF(LL1-LMAX) 509,1501,1501
  509 KEFIN(K)=MAXTU-MHAPN
      KESTA(K)=KEFIN(K)-KTIME(K)
      MSALE=100
      NAINLM=0
      GO TO 510
 1501 CONTINUE
  510 RETURN
      END
      SUBROUTINE MOVER
C
C
C   ITERATION TYPE II --- FORWARDS
C
C
      COMMON KINODE(70),KJNODE(70),KTIME(70),KRSRCE(70),NET(70)
      COMMON KEF(70),NLT(70),KLS(70),KLF(70),KFLOAT(70),KFREE(70)
      COMMON KINDEP(70),KSAFE(70),NSLACK(70),KRANK(70),LEVEL(80)
      COMMON KCHOIS(30),KLU,LMAX,MAXTU,NOTA,NAINLM,MMOVER,MAXRES
      COMMON YES,SIGNO,YNO,BLANCO,PALO,MSCHED(70,80),SINOP
      COMMON NA,NN,NLAST,NSLIP,NTIME,LSLIP,LTIME,NOSCH,MSALE,INCRE
      COMMON KES(70),KPOS(30),KPOSRK(30)
      DIMENSION KONTRA(70)
      DIMENSION KESTA(70),KEFIN(70),NESTA(70),KONTRN(70)
      EQUIVALENCE (KFREE(1),KESTA(1)),(KINDEP(1),KEFIN(1))
      EQUIVALENCE (KSAFE(1),NESTA(1)),(KFLOAT(1),KONTRN(1))
C
C   SETS ALL CONTROLS TO ZERO
C
      DO 1399 K=1,NA
      KONTRN(K)=0
      KONTRA(K)=0
 1399 CONTINUE
C
C   DETERMINES WHICH ACTIVITY - OF THOSE THAT CONTRIBUTE TO
C   BUILD UP LMAX - HAS THE SMALLEST IDENTIFICATION NUMBER.
C   THIS ACTIVITY (K9) WOULD BE THE LAST ACTIVITY TO BE
C   RESCHEDULED, IF LMAX CAN NOT BE CUT.
C
      K9=NA
      DO 1400 K=1,NAINLM
      IF(KCHOIS(K)-K9) 399,399,1400
  399 K9=KCHOIS(K)
 1400 CONTINUE
C
C  RESCHEDULES THE ACTIVITIES IN DESCENDING ORDER ACCORDING
C   TO THEIR IDENTIFICATION NUMBERS, STARTS WITH ACTIVITY NA
C
      DO 1401 K=1,NA
      K1=NA-K+1
      J=KJNODE(K1)
      IF(K1-K9) 421,460,460
  460 IF(J-NN) 401,400,400
  400 KONTRN(J)=NLT(NN)
  401 K2=KEFIN(K1)
      N5=NESTA(J)
      IF(K2-NLT(J)) 402,407,407
  402 K3=N5
  403 K4=K3-KTIME(K1)+1
      IF(K2-K3) 404,407,407
  404 DO 1402 N6=K4,K3
      LEV=LEVEL(N6)+KRSRCE(K1)
      IF(LMAX-LEV) 405,405,1402
  405 IF(K2-K3) 406,407,407
  406 K3=K3-1
      GO TO 403
 1402 CONTINUE
      K2=K3
  407 IF(N5-NLT(J)) 408,416,416
  408 IF(KONTRN(J)) 409,409,410
  409 KONTRN(J)=NLT(NN)
      DO 1403 K6=1,NA
      ICHE=KINODE(K6)
      IF(ICHE-J) 1403,470,1403
  470 K5=KESTA(K6)
      IF(K5-KONTRN(J)) 414,1403,1403
  414 KONTRN(J)=K5
 1403 CONTINUE
  410 IF(N5-KONTRN(J)) 411,416,416
  411 N5=N5+1
      GO TO 402
C
C   REPORTS THE NEW POSITION OF THE ACTIVITY K1
C
  416 KEFIN(K1)=K2
      KESTA(K1)=K2-KTIME(K1)
      K4=KESTA(K1)+1
C
C   ADJUSTS CHANGES IN THE RESOURCE PROFILE
C
      CALL REPROF
      IF(MAXTU-K4) 417,415,415
  415 IF(MAXTU-KEFIN(K1)) 418,418,417
  417 KONTRA(K1)=100
C
C   REPORTS WHETHER THE ITERATION WAS ACCOMPLISHED OR NOT
C
  418 DO 1404 K7=1,NAINLM
      IF(K1-KCHOIS(K7)) 1404,419,1404
  419 IF(KONTRA(K1)) 1404,1404,420
 1404 CONTINUE
 1401 CONTINUE
  421 MMOVER=0
      GO TO 422
  420 MMOVER=100
```

```
C
C      REPORTS THE NEW POSITIONS OF THE NODES
C
  422 DO 1405 N=1,NN
      IF(KONTRN(N)) 1405,1405,423
  423 NESTA(N)=KONTRN(N)
 1405 CONTINUE
      RETURN
      END
      SUBROUTINE MOVEL
C
C
C ITERATION TYPE II --- BACKWARDS
C
C
      COMMON KINODE(70),KJNODE(70),KTIME(70),KRSRCE(70),NET(70)
      COMMON KEF(70),NLT(70),KLS(70),KLF(70),KFLOAT(70),KFREE(70)
      COMMON KINDEP(70),KSAFE(70),NSLACK(70),KRANK(70),LEVEL(80)
      COMMON KCHOIS(30),KLU,LMAX,MAXTU,NOTA,NAINLM,MMOVER,MAXRES
      COMMON YES,SIGNO,YNO,BLANCO,PALO,MSCHED(70,80),SINOP
      COMMON NA,NN,NLAST,NSLIP,NTIME,LSLIP,LTIME,NOSCH,MSALE,INCRE
      COMMON KES(70),KPOS(30),KPOSRK(30)
      DIMENSION KONTRA(70)
      DIMENSION KESTA(70),KEFIN(70),NESTA(70),KONTRN(70)
      EQUIVALENCE (KFREE(1),KESTA(1)),(KINDEP(1),KEFIN(1))
      EQUIVALENCE (KSAFE(1),NESTA(1)),(KFLOAT(1),KONTRN(1))
      TINOCO=0
      DO 1600 K=1,NA
      KONTRN(K)=0
      KONTRA(K)=0
 1600 CONTINUE
C
C DETERMINES WHICH ACTIVITY - OF THOSE THAT CONTRIBUTE TO
C  BUILD UP LMAX - HAS THE LARGEST IDENTIFICATION NUMBER.
C THIS ACTIVITY (K9) WOULD BE THE LAST ACTIVITY TO BE
C RESCHEDULED, IF LMAX CAN NOT BE CUT.
C
      K9=1
      DO 1601 K=1,NAINLM
      IF(KCHOIS(K)-K9) 1601,1601,600
  600 K9=KCHOIS(K)
 1601 CONTINUE
C
C  RESCHEDULES THE ACTIVITIES IN ASCENDING ORDER ACCORDING
C  TO THEIR IDENTIFICATION NUMBERS, STARTS WITH ACTIVITY 1.
C
  680 IF(SINOP-1) 1660,1659,1659
 1659 TINOCO=50
      K9=NA
 1660 DO 1602 K=1,NA
      I=KINODE(K)
      IF(K-K9) 601,601,624
  601 IF(I-1) 602,602,603
  602 KONTRN(I)=NET(1)
  603 K2=KESTA(K)+1
      N5=NESTA(I)+1
      IF(K2-NET(I)-1) 612,612,604
  604 K3=N5
  605 K4=K3+KTIME(K)-1
      IF(K2-K3) 612,612,606
  606 DO 1603 N6=K3,K4
      LEV1=LEVEL(N6)
      IF(LEV1-LMAX) 607,610,610
  607 IF(N6-KESTA(K)-1) 609,608,608
  608 LEV1=LEV1-KRSRCE(K)
  609 LEV=LEV1+KRSRCE(K)
      IF(TINOCO) 630,630,629
  629 IF(LMAX-LEV) 610,1603,1603
  630 IF(LMAX-LEV) 610,610,1603
  610 IF(K2-K3) 612,612,611
  611 K3=K3+1
      GO TO 605
 1603 CONTINUE
      K2=K3
  612 IF(N5-1-NET(I)) 619,619,670
  670 IF(KONTRN(I)) 614,614,617
  614 KONTRN(I)=NET(1)
      DO 1604 K6=1,NA
      JCHE=KJNODE(K6)
      IF(JCHE-I) 1604,615,1604
  615 K5=KEFIN(K6)
      IF(K5-KONTRN(I)) 1604,1604,616
  616 KONTRN(I)=K5
 1604 CONTINUE
  617 IF(N5-KONTRN(I)-1) 619,619,618
  618 N5=N5-1
      GO TO 604
C
C   REPORTS THE NEW POSITION OF THE ACTIVITY K.
C
  619 KESTA(K)=K2-1
      KEFIN(K)=KESTA(K)+KTIME(K)
C
C   ADJUSTS CHANGES IN THE RESOURCE PROFILE
C
      CALL REPROF
      IF(MAXTU-K2) 621,620,620
  620 IF(MAXTU-KEFIN(K)) 622,622,621
  621 KONTRA(K)=100
C
C   REPORTS WHETHER THE ITERATION WAS ACCOMPLISHED OR NOT.
C
  622 IF(TINOCO) 681,681,1602
  681 DO 1605 K7=1,NAINLM
      IF(K-KCHOIS(K7)) 1605,623,1605
 623 IF(KONTRA(K)) 1605,1605,625
1605 CONTINUE
1602 CONTINUE
 624 MMOVER=0
     GO TO 626
 625 MMOVER=100

   REPORTS THE NEW POSITIONS OF THE NODES.

 626 DO 1606 N=1,NN
     IF(KONTRN(N)) 1606,1606,627
 627 NESTA(N)=KONTRN(N)
1606 CONTINUE
     IF(TINOCO) 643,643,640
 640 K9=0
     DO 1607 K8=1,NA
     IF(KJNODE(K8)-NN) 1607,641,641
 641 IF(K9-KEFIN(K8)) 642,1607,1607
 642 K9=KEFIN(K8)
1607 CONTINUE
     NESTA(NN)=K9
 643 RETURN
     END
```

APPENDIX B

COMPUTER SOLUTION OF THE EXAMPLE PROJECT

DATA - INPUT

```
/*
//GO.SYSIN DD *
 YES*    NO
008024003007
001002008007
001003003002
001004005004
002003000000
002004002000
002005005006
003005003001
004005006006
/*
```

```
                    N O D E S = E V E N T S

NUMBER     EARLIEST TIME        LATEST TIME     SLACK      CRITICAL PATH

   1                 0                  0          0                YES
   2                 8                  8          0                YES
   3                 8                 13          5                 NO
   4                10                 10          0                YES
   5                16                 16          0                YES
```

```
                                             A C T I V I T I E S

NUMBER INODE JNODE  RSRCE   DURA   E-START E-FIN   L-START   L-FIN  TOTAL-F   FREE-F  IND-F  SAFE-F C-PATH DUMMY INDEX

  1      1     2      7       8       0      8        0        8       0         0       0       0    YES     NO     1

  2      1     3      2       3       0      3       10       13      10         5       5      10     NO     NO     8

  3      1     4      4       5       0      5        5       10       5         5       5       5     NO     NO     7

  4      2     3      0       0       8      8       13       13       5         0       0       5     NO    YES     5

  5      2     4      0       2       8     10        8       10       0         0       0       0    YES     NO     2

  6      2     5      6       5       8     13       11       16       3         3       3       3     NO     NO     4

  7      3     5      1       3       8     11       13       16       5         5       0       0     NO     NO     6

  8      4     5      6       6      10     16       10       16       0         0       0       0    YES     NO     3


   THE EARLIEST PROJECT COMPLETION TIME IS    16 UNITS OF TIME
   THE GIVEN PROJECT DUE DATE IS   24 UNITS OF TIME
   THE CURRENT PROJECT SLACK IS    8 UNITS OF TIME
   THE DAILY RESOURCE AVAILABILITY IS    7 UNITS OF RESOURCE
   THE MAXIMUM PROJECT SLIPPAGE IS    3 UNITS OF TIME
```

```
                     S C H E D U L E   N U M B E R   1

            ACTIVITIES : 1,2,3,........, 8
TIME-UNIT                                                        LEVEL

   NODES LOCATED HERE ARE : 1.
       ¬¬¬
   1   72400000                                                    13

   2   72400000                                                    13

   3   72400000                                                    13
        ¬
   4   70400000                                                    11

   5   70400000                                                    11
         ¬
   6   70000000                                                     7

   7   70000000                                                     7

   8   70000000                                                     7
       ¬  ¬
   NODES LOCATED HERE ARE : 2. 3.
          ¬¬¬¬
   9   00000610                                                     7

  10   00000610                                                     7
           ¬
   NODES LOCATED HERE ARE : 4.
              ¬
  11   00000616                                                    13
             ¬
  12   00000606                                                    12

  13   00000606                                                    12
            ¬
  14   00000006                                                     6

  15   00000006                                                     6

  16   00000006                                                     6
              ¬
   NODES LOCATED HERE ARE : 5.
```

```
                    S C H E D U L E   N U M B E R   3

          ACTIVITIES : 1,2,3,........, 8
TIME-UNIT                                                                LEVEL

    NODES LOCATED HERE ARE : 1.
           ¬ ¬
    1    70400000                                                          11

    2    70400000                                                          11

    3    70400000                                                          11

    4    70400000                                                          11

    5    70400000                                                          11
           ¬
          ¬
    6    72000000                                                           9

    7    72000000                                                           9

    8    72000000                                                           9
         ¬¬ ¬
    NODES LOCATED HERE ARE : 2. 3.
             ¬¬¬
    9    00000600                                                           6

   10    00000600                                                           6
              ¬
    NODES LOCATED HERE ARE : 4.
                ¬
   11    00000606                                                          12

   12    00000606                                                          12

   13    00000606                                                          12
              ¬
               ¬
   14    00000016                                                           7

   15    00000016                                                           7

   16    00000016                                                           7
               ¬¬
    NODES LOCATED HERE ARE : 5.
```

THE HEURISTIC BEST SOLUTION SCHEDULE FOR THE GIVEN PROJECT IS :

```
                    S C H E D U L E   N U M B E R   8

          ACTIVITIES : 1,2,3,........, 8
TIME-UNIT                                                          LEVEL

    NODES LOCATED HERE ARE : 1.
        ¬
    1   70000000                                                     7

    2   70000000                                                     7

    3   70000000                                                     7

    4   70000000                                                     7

    5   70000000                                                     7

    6   70000000                                                     7

    7   70000000                                                     7

    8   70000000                                                     7
        ¬  ¬
    NODES LOCATED HERE ARE : 2.
         ¬¬¬¬
    9   02400000                                                     6

   10   02400000,                                                    6
            ¬
   11   02400000                                                     6
         ¬
    NODES LOCATED HERE ARE : 3.
              ¬
   12   00400010                                                     5

   13   00400010                                                     5
          ¬
    NODES LOCATED HERE ARE : 4.
              ¬
   14   00000610                                                     7
              ¬
   15   00000600                                                     6

   16   00000600                                                     6
```

```
17   00000600                                                        6

18   00000600                                                        6

19   00000006                                                        6

20   00000006                                                        6

21   00000006                                                        6

22   00000006                                                        6

23   00000006                                                        6

24   00000006                                                        6
 NODES LOCATED HERE ARE : 5.
```

APPENDIX C

COMPUTER SOLUTION OF THE CONSTRUCTION OF A GAS STATION

```
                 N O D E S = E V E N T S

NUMBER     EARLIEST TIME        LATEST TIME    SLACK        CRITICAL PATH

   1             0                   0            0               YES
   2             5                   5            0               YES
   3             8                  17            9                NO
   4            10                  19            9                NO
   5            14                  23            9                NO
   6            14                  23            9                NO
   7            15                  31           16                NO
   8            18                  34           16                NO
   9            10                  10            0               YES
  10             8                  39           31                NO
  11            15                  41           26                NO
  12            25                  25            0               YES
  13            20                  36           16                NO
  14            29                  29            0               YES
  15            31                  31            0               YES
  16            37                  40            3                NO
  17            35                  35            0               YES
  18            23                  39           16                NO
  19            26                  42           16                NO
  20            37                  42            5                NO
  21            26                  49           23                NO
  22            35                  46           11                NO
  23            37                  46            9                NO
  24            37                  40            3                NO
  25            40                  40            0               YES
  26            44                  44            0               YES
  27            19                  45           26                NO
```

ACTIVITIES

NUMBER	INODE	JNODE	RSRCE	DURA	E-START	E-FIN	L-START	L-FIN	TOTAL-F	FREE-F	IND-F	SAFE-F	C-PATH	DUMMY	INDEX
1	1	2	3	5	0	5	0	5	0	0	0	0	YES	NO	1
2	2	3	1	3	5	8	14	17	9	0	0	9	NO	NO	27
3	2	7	6	10	5	15	21	31	16	0	0	16	NO	NO	40
4	2	9	2	5	5	10	5	10	0	0	0	0	YES	NO	2
5	2	11	1	10	5	15	31	41	26	0	0	26	NO	NO	48
6	2	20	12	10	5	15	32	42	27	22	22	27	NO	NO	51
7	2	21	5	10	5	15	39	49	34	11	11	34	NO	NO	55
8	2	22	8	15	5	20	31	46	26	15	15	26	NO	NO	49
9	2	23	4	20	5	25	26	46	21	12	12	21	NO	NO	43
10	2	28	1	5	5	10	40	45	35	0	0	35	NO	NO	56
11	3	4	0	2	8	10	17	19	9	0	0	0	NO	NO	23
12	3	6	3	1	8	9	22	23	14	5	0	5	NO	NO	35
13	3	7	0	0	8	8	31	31	23	7	0	14	NO	YES	45
14	3	10	0	0	8	8	39	39	31	0	0	22	NO	YES	53
15	4	5	10	4	10	14	19	23	9	0	0	0	NO	NO	24
16	5	6	0	0	14	14	23	23	9	0	0	0	NO	YES	25
17	5	8	0	0	14	14	34	34	20	4	0	11	NO	YES	42
18	6	12	5	2	14	16	23	25	9	9	0	0	NO	NO	29
19	7	8	3	3	15	18	31	34	16	0	0	0	NO	NO	36
20	8	13	1	2	18	20	34	36	16	0	0	0	NO	NO	37
21	9	12	6	15	10	25	10	25	0	0	0	0	YES	NO	3
22	10	11	7	2	8	10	39	41	31	5	0	0	NO	NO	54
23	11	27	3	4	15	19	41	45	26	0	0	0	NO	NO	47
24	12	14	4	4	25	29	25	29	0	0	0	0	YES	NO	4
25	13	18	1	3	20	23	36	39	16	0	0	0	NO	NO	38
26	14	15	2	2	29	31	29	31	0	0	0	0	YES	NO	5
27	15	16	1	6	31	37	34	40	3	0	0	3	NO	NO	16

28	15	17	2	4	31	35	31	35	0	0	0	0	YES	NO	6
29	15	31	5	5	31	36	41	46	10	8	8	10	NO	NO	31
30	16	20	0	0	37	37	42	42	5	0	0	2	NO	YES	21
31	16	23	0	0	37	37	46	46	9	0	0	6	NO	YES	26
32	16	25	0	0	37	37	40	40	3	3	0	0	NO	YES	18
33	17	22	0	0	35	35	46	46	11	0	0	11	NO	YES	33
34	17	24	9	2	35	37	38	40	3	0	0	3	NO	NO	17
35	17	25	3	5	35	40	35	40	0	0	0	0	YES	NO	7
36	17	30	0	0	35	35	45	45	10	0	0	10	NO	YES	30
37	18	19	15	3	23	26	39	42	16	0	0	0	NO	NO	39
38	19	20	0	0	26	26	42	42	16	11	0	0	NO	YES	41
39	19	21	0	0	26	26	49	49	23	0	0	7	NO	YES	44
40	20	29	3	6	37	43	42	48	5	0	0	0	NO	NO	20
41	21	35	1	2	26	28	49	51	23	23	0	0	NO	NO	46
42	22	32	2	2	35	37	46	48	11	9	0	0	NO	NO	34
43	23	32	4	2	37	39	46	48	9	7	0	0	NO	NO	28
44	24	25	0	0	37	37	40	40	3	3	0	0	NO	YES	19
45	25	26	5	4	40	44	40	44	0	0	0	0	YES	NO	8
46	26	31	0	0	44	44	46	46	2	0	0	2	NO	YES	14
47	26	33	3	2	44	46	44	46	0	0	0	0	YES	NO	9
48	27	30	0	0	19	19	45	45	26	16	0	0	NO	YES	50
49	27	32	0	0	19	19	48	48	29	27	1	3	NO	YES	52
50	28	29	0	0	10	10	48	48	38	33	0	3	NO	YES	58
51	28	30	0	0	10	10	45	45	35	25	0	0	NO	YES	57
52	29	35	7	3	43	46	48	51	5	5	0	0	NO	NO	22
53	30	35	1	6	35	41	45	51	10	10	0	0	NO	NO	32
54	31	32	8	2	44	46	46	48	2	0	0	0	NO	NO	13
55	32	35	2	3	46	49	48	51	2	2	0	0	NO	NO	15
56	33	34	1	3	46	49	46	49	0	0	0	0	YES	NO	10
57	34	35	3	2	49	51	49	51	0	0	0	0	YES	NO	11

```
58    35    36     4     1    51     52     51     52     0      0      0      0   YES    NO    12
```

```
THE EARLIEST PROJECT COMPLETION TIME IS     52 UNITS OF TIME
THE GIVEN PROJECT DUE DATE IS   60 UNITS OF TIME
THE CURRENT PROJECT SLACK IS    8 UNITS OF TIME
THE DAILY RESOURCE AVAILABILITY IS   21 UNITS OF RESOURCE
THE MAXIMUM PROJECT SLIPPAGE IS    3 UNITS OF TIME
```

SCHEDULE NUMBER 1

```
              ACTIVITIES : 1,2,3,........,58
TIME-UNIT                                                                   LEVEL

    NODES LOCATED HERE ARE : 1.
        ¬
    1   3000000000000000000000000000000000000000000000000000000000          3

    2   3000000000000000000000000000000000000000000000000000000000          3

    3   3000000000000000000000000000000000000000000000000000000000          3

    4   3000000000000000000000000000000000000000000000000000000000          3

    5   3000000000000000000000000000000000000000000000000000000000          3
        ¬
    NODES LOCATED HERE ARE : 2.
        ¬¬¬¬¬¬¬¬¬¬
    6   01621*58410000000000000000000000000000000000000000000000000         40

    7   01621*58410000000000000000000000000000000000000000000000000         40

    8   01621*58410000000000000000000000000000000000000000000000000         40
          ¬         ¬¬
    NODES LOCATED HERE ARE : 3.10.
                   ¬¬¬¬      ¬
    9   0062 1*5841030000000007000000000000000000000000000000000000         49
                    ¬
    10  00621*58410000000000070000000000000000000000000000000000000         46
            ¬     ¬¬         ¬                            ¬¬
    NODES LOCATED HERE ARE : 4. 9.28.
                      ¬     ¬                             ¬¬
    11  00601*58400000*000006000000000000000000000000000000000000          52

    12  00601*58400000*000006000000000000000000000000000000000000          52

    13  00601*58400000*000006000000000000000000000000000000000000          52

    14  00601*58400000*000006000000000000000000000000000000000000          52
                       ¬¬¬
    NODES LOCATED HERE ARE : 5. 6.
                        ¬¬¬
    15  00601*5840000000050060000000000000000000000000000000000000          47
           ¬  ¬¬¬
    NODES LOCATED HERE ARE : 7.11.
                           ¬   ¬
```

```
16    0000000840000000053060300000000000000000000000000000000000          29

17    0000000840000000003060300000000000000000000000000000000000          24

18    0000000840000000003060300000000000000000000000000000000000          24
 NODES LOCATED HERE ARE : 8.
19    0000000840000000000160300000000000000000000000000000000000          22
 NODES LOCATED HERE ARE :27.
20    0000000840000000000160000000000000000000000000000000000000          19
 NODES LOCATED HERE ARE :13.
21    0000000040000000000060001000000000000000000000000000000000          11

22    0000000040000000000060001000000000000000000000000000000000          11

23    0000000040000000000060001000000000000000000000000000000000          11
 NODES LOCATED HERE ARE :18.
24    000000004000000000006000000000000000*000000000000000000000          25

25    000000004000000000006000000000000000*000000000000000000000          25
 NODES LOCATED HERE ARE :12.
26    00000000000000000000000400000000000*000000000000000000000          19
 NODES LOCATED HERE ARE :19.21.
27    0000000000000000000000040000000000000000100000000000000000           5

28    0000000000000000000000040000000000000000100000000000000000           5

29    0000000000000000000000040000000000000000000000000000000000           4
 NODES LOCATED HERE ARE :14.
30    0000000000000000000000000200000000000000000000000000000000           2

31    0000000000000000000000000200000000000000000000000000000000           2
 NODES LOCATED HERE ARE :15.
32    0000000000000000000000000012500000000000000000000000000000           8

33    0000000000000000000000000012500000000000000000000000000000           8
```

```
34   00000000000000000000000001250000000000000000000000000000          8

35   00000000000000000000000001250000000000000000000000000000          8
 NODES LOCATED HERE ARE :17.22.30.

36   00000000000000000000000001050000930000002000000000100000         21

37   00000000000000000000000001000000930000002000000000100000         16
 NODES LOCATED HERE ARE :16.20.23.24.

38   00000000000000000000000000000000030000300400000000100000         11

39   00000000000000000000000000000000030000300400000000100000         11

40   00000000000000000000000000000000030000300000000000100000          7
 NODES LOCATED HERE ARE :25.

41   00000000000000000000000000000000000000300005000000100000          9

42   00000000000000000000000000000000000000300005000000000000          8

43   00000000000000000000000000000000000000300005000000000000          8
 NODES LOCATED HERE ARE :29.

44   00000000000000000000000000000000000000000050000007000000         12
 NODES LOCATED HERE ARE :26.31.

45   00000000000000000000000000000000000000000000300007080000         18

46   00000000000000000000000000000000000000000000300007080000         18
 NODES LOCATED HERE ARE :32.33.

47   00000000000000000000000000000000000000000000000000002100          3

48   00000000000000000000000000000000000000000000000000002100          3

49   00000000000000000000000000000000000000000000000000002100          3
 NODES LOCATED HERE ARE :34.

50   00000000000000000000000000000000000000000000000000000030          3
```

```
51    000000000000000000000000000000000000000000000000000000000000000030                    3
                                                                       ¬
  NODES LOCATED HERE ARE :35.
                                                                       ¬
52    000000000000000000000000000000000000000000000000000000000000000004                    4
                                                                       ¬
  NODES LOCATED HERE ARE :36.
```

```
                     S C H E D U L E   N U M B E R  19

              ACTIVITIES : 1,2,3,........,58
TIME-UNIT                                                                        LEVEL

  NODES LOCATED HERE ARE : 1.
      ¬
  1   3000000000000000000000000000000000000000000000000000000000                    3

  2   3000000000000000000000000000000000000000000000000000000000                    3

  3   3000000000000000000000000000000000000000000000000000000000                    3

  4   3000000000000000000000000000000000000000000000000000000000                    3

  5   3000000000000000000000000000000000000000000000000000000000                    3
      ¬
  NODES LOCATED HERE ARE : 2.
       ¬¬¬ ¬   ¬
  6   01620*000100000000000000000000000000000000000000000000000                    22

  7   01620*000100000000000000000000000000000000000000000000000                    22

  8   01620*000100000000000000000000000000000000000000000000000                    22
       ¬          ¬¬
  NODES LOCATED HERE ARE : 3.10.
                ¬¬¬¬
  9   00620*0001030000000000000000000000000000000000000000000000                   24
                 ¬

 10   00620*0001000000000000000000000000000000000000000000000000                   21
         ¬     ¬¬
  NODES LOCATED HERE ARE : 4. 9.28.
                          ¬
 11   00600*00000000000000600000000000000000000000000000000000000                  24

 12   00600*00000000000000600000000000000000000000000000000000000                  24

 13   00600*00000000000000600000000000000000000000000000000000000                  24

 14   00600*00000000000000600000000000000000000000000000000000000                  24

 15   00600*00000000000000600000000000000000000000000000000000000                  24
        ¬  ¬
  NODES LOCATED HERE ARE : 7.
          ¬   ¬     ¬   ¬
 16   00001000400000*000306000000000000000000000000000000000000                    24
```

```
17   00001000400000*000306000000000000000000000000000000000000                                            24

18   00001000400000*000306000000000000000000000000000000000000                                            24
                       ¬                              ¬¬
                                                      ¬¬
19   00001000400000*000006000000000000000000000000000000000000                                            21
                   ¬¬¬
 NODES LOCATED HERE ARE : 5. 6. 8.
                    ¬¬¬ ¬ ¬
20   000010004000000005016700000000000000000000000000000000000                                            24

21   000010004000000005016700000000000000000000000000000000000                                            24
                      ¬ ¬ ¬
 NODES LOCATED HERE ARE :13.
            ¬                ¬
22   000010084000000000006000100000000000000000000000000000000                                            20

23   000010084000000000006000100000000000000000000000000000000                                            20

24   000010084000000000006000100000000000000000000000000000000                                            20
                             ¬
 NODES LOCATED HERE ARE :18.

25   000010084000000000006000000000000000000000000000000000000                                            19
         ¬               ¬
 NODES LOCATED HERE ARE :11.12.
           ¬                ¬
26   000000584000000000000004000000000000000000000000000000000                                            21

27   000000584000000000000004000000000000000000000000000000000                                            21
                           ¬
28   000000584000000000000034000000000000000000000000000000000                                            24

29   000000584000000000000034000000000000000000000000000000000                                            24
                            ¬
 NODES LOCATED HERE ARE :14.
                              ¬
30   000000584000000000000030020000000000000000000000000000000                                            22

31   000000584000000000000030020000000000000000000000000000000                                            22
                           ¬  ¬                     ¬¬
 NODES LOCATED HERE ARE :15.27.
                               ¬¬                   ¬¬
32   000000584000000000000000001200000000000000000000000000000                                            20

33   000000584000000000000000001200000000000000000000000000000                                            20

34   000000584000000000000000001200000000000000000000000000000                                            20
```

```
35   000000584000000000000000001200000000000000000000000000000                                          20
           ¬ ¬                  ¬    ¬  ¬
 NODES LOCATED HERE ARE :17.30.
                                 ¬   ¬ ¬¬
36   000000080000000000000000001050000030000000000000000000000                                          17
            ¬
 NODES LOCATED HERE ARE :22.
                                      ¬       ¬
37   000000000000000000000000001050000930000002000000000000000                                          20
                               ¬  ¬¬¬
 NODES LOCATED HERE ARE :16.23.
                                  ¬¬¬
38   000000000000000000000000000050000930000002000000000000000                                          19
                                      ¬       ¬ ¬
 NODES LOCATED HERE ARE :24.
                                         ¬      ¬
39   000000000000000000000000000050000030*00000000000000000000                                          23

40   000000000000000000000000000050000030*00000000000000000000                                          23
                                 ¬     ¬
 NODES LOCATED HERE ARE :25.
                                              ¬ ¬
41   000000000000000000000000000000000000*00004050000000000000                                          24
                                         ¬¬¬
 NODES LOCATED HERE ARE :19.20.21.
                                         ¬¬¬¬           ¬
42   000000000000000000000000000000000000003104050000000100000                                          14
                                              ¬

43   000000000000000000000000000000000000003100050000000100000                                          10
                                            ¬

44   000000000000000000000000000000000000003000050000000100000                                           9
                                                ¬¬
 NODES LOCATED HERE ARE :26.31.
                                                 ¬¬      ¬
45   000000000000000000000000000000000000003000000300000180000                                          15

46   000000000000000000000000000000000000003000000300000180000                                          15
                                                  ¬      ¬
 NODES LOCATED HERE ARE :32.33.
                                                          ¬¬
47   000000000000000000000000000000000000003000000000000102100                                           7
                                           ¬            ¬
 NODES LOCATED HERE ARE :29.
                                                       ¬
48   000000000000000000000000000000000000000000000000007002100                                          10

49   000000000000000000000000000000000000000000000000007002100                                          10
                                                          ¬¬
 NODES LOCATED HERE ARE :34.
                                                            ¬
50   000000000000000000000000000000000000000000000000007000030                                          10
                                                       ¬
```

```
51   00000000000000000000000000000000000000000000000000000000030                                             3
                                                              ¬
 NODES LOCATED HERE ARE :35.
                                                               ¬
52   0000000000000000000000000000000000000000000000000000000004                                              4
                                                               ¬
 NODES LOCATED HERE ARE :36.
```

```
                          SCHEDULE NUMBER 27

              ACTIVITIES : 1,2,3,.......,58
TIME-UNIT                                                          LEVEL


  NODES LOCATED HERE ARE : 1.
      ¬
  1   3000000000000000000000000000000000000000000000000000000000       3

  2   3000000000000000000000000000000000000000000000000000000000       3

  3   3000000000000000000000000000000000000000000000000000000000       3

  4   3000000000000000000000000000000000000000000000000000000000       3

  5   3000000000000000000000000000000000000000000000000000000000       3
      ¬
  NODES LOCATED HERE ARE : 2.
       ¬ ¬¬¬   ¬
  6   0102 1*00010000000000000000000000000000000000000000000000       17

  7   01021*0001000000000000000000000000000000000000000000000000      17

  8   01021*0001000000000000000000000000000000000000000000000000      17
       ¬          ¬¬
  NODES LOCATED HERE ARE : 3.10.
                ¬¬¬¬
  9   00021*0001030000000000000000000000000000000000000000000000      19
                 ¬

 10   00021*0001000000000000000000000000000000000000000000000000      16
         ¬     ¬¬                                      ¬¬
  NODES LOCATED HERE ARE : 4. 9.28.
                          ¬                            ¬¬
 11   00001*00000000000000600000000000000000000000000000000000000     19

 12   00001*0000000000000060000000000000000000000000000000000000      19

 13   00001*0000000000000060000000000000000000000000000000000000      19

 14   00001*0000000000000060000000000000000000000000000000000000      19

 15   00001*0000000000000060000000000000000000000000000000000000      19
          ¬¬
              ¬     ¬
 16   0000000040000*000006000000000000000000000000000000000000000     20
```

```
17   0000000040000000*000006000000000000000000000000000000000000                                           20

18   00000000400000*0000060000000000000000000000000000000000000                                           20

19   00000000400000*0000060000000000000000000000000000000000000                                           20
                   ¬¬¬
 NODES LOCATED HERE ARE : 5. 6.
            ¬       ¬¬
20   0000000840000000000060000000000000000000000000000000000000                                           18

21   0000000840000000000060000000000000000000000000000000000000                                           18

22   0000000840000000000060000000000000000000000000000000000000                                           18

23   0000000840000000000060000000000000000000000000000000000000                                           18

24   0000000840000000000060000000000000000000000000000000000000                                           18

25   0000000840000000000060000000000000000000000000000000000000                                           18
                         ¬
                      ¬
26   0000000840000000050000000000000000000000000000000000000000                                           17

27   0000000840000000050000000000000000000000000000000000000000                                           17
                      ¬
 NODES LOCATED HERE ARE :12.
                          ¬
28   0000000840000000000007000000000000000000000000000000000000                                           19

29   0000000840000000000007000000000000000000000000000000000000                                           19
                          ¬
 NODES LOCATED HERE ARE :11.
       ¬                   ¬
30   0060000840000000000000300000000000000000000000000000000000                                           21

31   0060000840000000000000300000000000000000000000000000000000                                           21

32   0060000840000000000000300000000000000000000000000000000000                                           21

33   0060000840000000000000300000000000000000000000000000000000                                           21
                           ¬
 NODES LOCATED HERE ARE :27.
                            ¬
34   0060000840000000000000040000000000000000000000000000000000                                           22
            ¬                                        ¬¬
           ¬                                         ¬¬
35   0060005040000000000000040000000000000000000000000000000000                                           19
```

```
36   0060005000000000000000040000000000000000000000000000000000          15

37   0060005000000000000000040000000000000000000000000000000000          15

 NODES LOCATED HERE ARE :14.

38   0060005000000000000000000200000000000000000000000000000000          13

39   0060005000000000000000000200000000000000000000000000000000          13

 NODES LOCATED HERE ARE : 7.15.

40   0000005000000000003000000012500000000000000000000000000000          16

41   0000005000000000003000000012500000000000000000000000000000          16

42   0000005000000000003000000012500000000000000000000000000000          16

 NODES LOCATED HERE ARE : 8.

43   0000005000000000000100000012500000000000000000000000000000          14

 NODES LOCATED HERE ARE :17.22.30.

44   0000005000000000000100000010500000300000020000000000100000          18

 NODES LOCATED HERE ARE :13.

45   0000000000000000000000001010000009300000020000000000100000          17

 NODES LOCATED HERE ARE :16.23.

46   0000000000000000000000001000000009300000004000000000100000          18

 NODES LOCATED HERE ARE :24.

47   0000000000000000000000001000000000300000004000000000100000           9

 NODES LOCATED HERE ARE :18.

48   000000000000000000000000000000000030*000000000000000100000          19

 NODES LOCATED HERE ARE :25.

49   000000000000000000000000000000000000*000000050000000100000          21

50   000000000000000000000000000000000000*000000050000000000000          20

 NODES LOCATED HERE ARE :19.20.21.

51   0000000000000000000000000000000000000003100050000000000000           9
```

```
52    0000000000000000000000000000000000000003100050000000000000          9
                                              ¬   ¬¬
 NODES LOCATED HERE ARE :26.31.
                                                   ¬¬      ¬
53    0000000000000000000000000000000000000003000000300000080000         14

54    0000000000000000000000000000000000000003000000300000080000         14
                                                    ¬      ¬
 NODES LOCATED HERE ARE :32.33.
                                                            ¬¬
55    0000000000000000000000000000000000000003000000000000002100          6

56    0000000000000000000000000000000000000003000000000000002100          6
                                             ¬
 NODES LOCATED HERE ARE :29.
                                                         ¬
57    0000000000000000000000000000000000000000000000000007002100         10
                                                            ¬¬
 NODES LOCATED HERE ARE :34.
                                                              ¬
58    0000000000000000000000000000000000000000000000000007000030         10

59    0000000000000000000000000000000000000000000000000007000030         10
                                                         ¬    ¬
 NODES LOCATED HERE ARE :35.
                                                               ¬
60    0000000000000000000000000000000000000000000000000000000004          4
                                                               ¬
 NODES LOCATED HERE ARE :36.
```

```
                S C H E D U L E   N U M B E R  28

            ACTIVITIES : 1,2,3,........,58
TIME-UNIT                                                                 LEVEL

    NODES LOCATED HERE ARE : 1.
        ¬
    1   3000000000000000000000000000000000000000000000000000000000        3

    2   3000000000000000000000000000000000000000000000000000000000        3

    3   3000000000000000000000000000000000000000000000000000000000        3

    4   3000000000000000000000000000000000000000000000000000000000        3

    5   3000000000000000000000000000000000000000000000000000000000        3
        ¬
    NODES LOCATED HERE ARE : 2.
         ¬ ¬¬¬   ¬
    6   01021*000100000000000000000000000000000000000000000000000000      17

    7   01021*000100000000000000000000000000000000000000000000000000      17

    8   01021*000100000000000000000000000000000000000000000000000000      17
         ¬          ¬¬
    NODES LOCATED HERE ARE : 3.10.
                  ¬¬¬¬
    9   00021*0001030000000000000000000000000000000000000000000000        19
                   ¬
   10   00021*000100000000000000000000000000000000000000000000000000      16
           ¬     ¬¬                                      ¬¬
    NODES LOCATED HERE ARE : 4. 9.28.
                           ¬                             ¬¬
   11   00001*00000000000006000000000000000000000000000000000000000       19

   12   00001*00000000000006000000000000000000000000000000000000000       19

   13   00001*00000000000006000000000000000000000000000000000000000       19

   14   00001*00000000000006000000000000000000000000000000000000000       19

   15   00001*00000000000006000000000000000000000000000000000000000       19
            ¬¬
                ¬     ¬
   16   000000004000000*00006000000000000000000000000000000000000000       20
```

```
17   0000000040000*000006000000000000000000000000000000000000000000                                        20

18   0000000040000*000006000000000000000000000000000000000000000000                                        20

19   0000000040000*000006000000000000000000000000000000000000000000                                        20
                   ¬¬¬
 NODES LOCATED HERE ARE : 5. 6.
            ¬       ¬¬
20   0000000840000000000060000000000000000000000000000000000000                                        18

21   0000000840000000000060000000000000000000000000000000000000                                        18

22   0000000840000000000060000000000000000000000000000000000000                                        18

23   0000000840000000000060000000000000000000000000000000000000                                        18

24   0000000840000000000060000000000000000000000000000000000000                                        18

25   0000000840000000000060000000000000000000000000000000000000                                        18
                         ¬
                      ¬
26   0000000840000000050000000000000000000000000000000000000000                                        17

27   0000000840000000050000000000000000000000000000000000000000                                        17
                      ¬
 NODES LOCATED HERE ARE :12.
                          ¬
28   0000000840000000000007000000000000000000000000000000000000                                        19

29   0000000840000000000007000000000000000000000000000000000000                                        19
                          ¬
 NODES LOCATED HERE ARE :11.
       ¬                   ¬
30   0060000840000000000000300000000000000000000000000000000000                                        21

31   0060000840000000000000300000000000000000000000000000000000                                        21

32   0060000840000000000000300000000000000000000000000000000000                                        21

33   0060000840000000000000300000000000000000000000000000000000                                        21
                           ¬
 NODES LOCATED HERE ARE :27.
                            ¬
34   0060000840000000000000040000000000000000000000000000000000                                        22
            ¬                                       ¬¬
           ¬                                        ¬¬
35   0060005040000000000000040000000000000000000000000000000000                                        19
```

```
36   0060005000000000000000040000000000000000000000000000000000        15

37   0060005000000000000000040000000000000000000000000000000000        15
 NODES LOCATED HERE ARE :14.
38   0060005000000000000000000200000000000000000000000000000000        13

39   0060005000000000000000000200000000000000000000000000000000        13
 NODES LOCATED HERE ARE : 7.15.
40   0000005000000000003000000012500000000000000000000000000000        16

41   0000005000000000003000000012500000000000000000000000000000        16

42   0000005000000000003000000012500000000000000000000000000000        16
 NODES LOCATED HERE ARE : 8.
43   0000005000000000000100000012500000000000000000000000000000        14
 NODES LOCATED HERE ARE :17.22.30.
44   0000005000000000000100000010500000300000020000000000100000        18
 NODES LOCATED HERE ARE :13.
45   0000000000000000000000001010000009300000020000000000100000        17
 NODES LOCATED HERE ARE :16.23.
46   0000000000000000000000001000000009300000004000000000100000        18
 NODES LOCATED HERE ARE :24.
47   0000000000000000000000001000000000300000004000000000100000         9
 NODES LOCATED HERE ARE :18.
48   000000000000000000000000000000000030*000000000000000100000        19
 NODES LOCATED HERE ARE :25.
49   000000000000000000000000000000000000*000000050000000100000        21

50   000000000000000000000000000000000000*000000050000000000000        20
 NODES LOCATED HERE ARE :19.20.21.
51   0000000000000000000000000000000000000003100050000000000000         9
```

```
52   0000000000000000000000000000000000000003100050000000000000                                           9
                                             ¬   ¬¬
 NODES LOCATED HERE ARE :26.31.
                                                  ¬¬      ¬
53   0000000000000000000000000000000000000003000000300000080000                                          14

54   0000000000000000000000000000000000000003000000300000080000                                          14
                                                   ¬      ¬
 NODES LOCATED HERE ARE :32.33.
                                                           ¬¬
55   0000000000000000000000000000000000000003000000000000002100                                           6

56   0000000000000000000000000000000000000003000000000000002100                                           6
                                            ¬
 NODES LOCATED HERE ARE :29.
                                                        ¬
57   0000000000000000000000000000000000000000000000000007002100                                          10
                                                           ¬¬
 NODES LOCATED HERE ARE :34.
                                                             ¬
58   0000000000000000000000000000000000000000000000000007000030                                          10

59   0000000000000000000000000000000000000000000000000007000030                                          10
                                                        ¬    ¬
 NODES LOCATED HERE ARE :35.
                                                              ¬
60   0000000000000000000000000000000000000000000000000000000004                                           4
                                                              ¬
 NODES LOCATED HERE ARE :36.
61   0000000000000000000000000000000000000000000000000000000000                                           0
```

```
                         SCHEDULE NUMBER 29

              ACTIVITIES : 1,2,3,........,58
TIME-UNIT                                                                          LEVEL

  NODES LOCATED HERE ARE : 1.
      ¬
  1   3000000000000000000000000000000000000000000000000000000000              3

  2   3000000000000000000000000000000000000000000000000000000000              3

  3   3000000000000000000000000000000000000000000000000000000000              3

  4   3000000000000000000000000000000000000000000000000000000000              3

  5   3000000000000000000000000000000000000000000000000000000000              3
      ¬
  NODES LOCATED HERE ARE : 2.
       ¬ ¬¬¬   ¬
  6   01021*000100000000000000000000000000000000000000000000000000             17

  7   01021*000100000000000000000000000000000000000000000000000000             17

  8   01021*000100000000000000000000000000000000000000000000000000             17
       ¬          ¬¬
  NODES LOCATED HERE ARE : 3.10.
                ¬¬¬¬
  9   00021*00010300000000000000000000000000000000000000000000000             19
                 ¬

 10   00021*000100000000000000000000000000000000000000000000000000             16
         ¬     ¬¬                                       ¬¬
  NODES LOCATED HERE ARE : 4. 9.28.
                          ¬                             ¬¬
 11   0000 1*00000000000000600000000000000000000000000000000000000             19

 12   00001*000000000000006000000000000000000000000000000000000000             19

 13   00001*000000000000006000000000000000000000000000000000000000             19

 14   00001*000000000000006000000000000000000000000000000000000000             19

 15   00001*000000000000006000000000000000000000000000000000000000             19
          ¬¬
              ¬     ¬
 16   0000000040000 0*000006000000000000000000000000000000000000             20
```

```
17   00000000400000*000006000000000000000000000000000000000000          20

18   00000000400000*000006000000000000000000000000000000000000          20

19   00000000400000*000006000000000000000000000000000000000000          20
                   ¬¬¬
 NODES LOCATED HERE ARE : 5. 6.
            ¬       ¬¬
20   000000084000000000006000000000000000000000000000000000000          18

21   000000084000000000006000000000000000000000000000000000000          18

22   000000084000000000006000000000000000000000000000000000000          18

23   000000084000000000006000000000000000000000000000000000000          18

24   000000084000000000006000000000000000000000000000000000000          18

25   000000084000000000006000000000000000000000000000000000000          18
                         ¬
                      ¬
26   000000084000000005000000000000000000000000000000000000000          17

27   000000084000000005000000000000000000000000000000000000000          17
                      ¬
 NODES LOCATED HERE ARE :12.
                          ¬
28   000000084000000000000700000000000000000000000000000000000          19

29   000000084000000000000700000000000000000000000000000000000          19
                          ¬
 NODES LOCATED HERE ARE :11.
       ¬                   ¬
30   006000084000000000000030000000000000000000000000000000000          21

31   006000084000000000000030000000000000000000000000000000000          21

32   006000084000000000000030000000000000000000000000000000000          21

33   006000084000000000000030000000000000000000000000000000000          21
                           ¬
 NODES LOCATED HERE ARE :27.
                            ¬
34   006000084000000000000004000000000000000000000000000000000          22
            ¬                                      ¬¬
           ¬                                       ¬¬
35   006000504000000000000000400000000000000000000000000000000          19
```

```
             ¬
36   0060005000000000000000040000000000000000000000000000000000          15

37   0060005000000000000000040000000000000000000000000000000000          15
                            ¬
 NODES LOCATED HERE ARE :14.
                              ¬
38   0060005000000000000000000200000000000000000000000000000000          13

39   0060005000000000000000000200000000000000000000000000000000          13
       ¬     .                ¬
 NODES LOCATED HERE ARE : 7.15.
                       ¬       ¬¬¬
40   0000005000000000003000000012500000000000000000000000000000          16

41   0000005000000000003000000012500000000000000000000000000000          16

42   0000005000000000003000000012500000000000000000000000000000          16
                       ¬
 NODES LOCATED HERE ARE : 8.
                        ¬
43   0000005000000000000100000012500000000000000000000000000000          14
                                ¬    ¬  ¬
 NODES LOCATED HERE ARE :17.22.30.
                                     ¬¬ ¬                ¬
44   0000005000000000000100000010500090000000000000000000100000          22
           ¬            ¬        ¬
 NODES LOCATED HERE ARE :13.
                             ¬         ¬      ¬
45   0000000000000000000000001010000009300000020000000000100000          17
                               ¬  ¬¬¬ ¬         ¬
 NODES LOCATED HERE ARE :16.23.24.
                                  ¬¬¬          ¬¬
46   0000000000000000000000001000000000300000024000000000100000          11
                                              ¬

47   0000000000000000000000001000000000300000004000000000100000           9
                             ¬                 ¬
 NODES LOCATED HERE ARE :18.
                                         ¬
48   000000000000000000000000000000000030*000000000000000100000          19

49   000000000000000000000000000000000030*000000000000000100000          19
                                       ¬                 ¬
 NODES LOCATED HERE ARE :25.
                                                 ¬
50   000000000000000000000000000000000000*000000050000000000000          20
                                         ¬¬¬
 NODES LOCATED HERE ARE :19.20.21.
                                          ¬¬¬¬
51   0000000000000000000000000000000000000003100050000000000000           9

52   0000000000000000000000000000000000000003100050000000000000           9
```

```
                                                ¬
53    000000000000000000000000000000000000000003000050000000000000                                              8
                                                    ¬¬
 NODES LOCATED HERE ARE :26.31.
                                                     ¬¬      ¬
54    000000000000000000000000000000000000000003000000300000080000                                             14

55    000000000000000000000000000000000000000003000000300000080000                                             14
                                                      ¬      ¬
 NODES LOCATED HERE ARE :32.33.
                                                              ¬¬
56    000000000000000000000000000000000000000003000000000000002100                                              6
                                               ¬
 NODES LOCATED HERE ARE :29.
                                                           ¬
57    000000000000000000000000000000000000000000000000000007002100                                             10

58    000000000000000000000000000000000000000000000000000007002100                                             10
                                                              ¬¬
 NODES LOCATED HERE ARE :34.
                                                                ¬
59    000000000000000000000000000000000000000000000000000007000030                                             10
                                                           ¬
60    000000000000000000000000000000000000000000000000000000000030                                              3
                                                                ¬
 NODES LOCATED HERE ARE :35.
                                                                 ¬
61    000000000000000000000000000000000000000000000000000000000004                                              4
                                                                 ¬
 NODES LOCATED HERE ARE :36.

62    000000000000000000000000000000000000000000000000000000000000                                              0
```

THE HEURISTIC BEST SOLUTIC

```
                    S C H E D U L E   N U M B E R  34

               ACTIVITIES : 1,2,3,........,58
TIME-UNIT                                                                          LEVEL

    NODES LOCATED HERE ARE : 1.
        ¬
   1    3000000000000000000000000000000000000000000000000000000000                   3

   2    3000000000000000000000000000000000000000000000000000000000                   3

   3    3000000000000000000000000000000000000000000000000000000000                   3

   4    3000000000000000000000000000000000000000000000000000000000                   3

   5    3000000000000000000000000000000000000000000000000000000000                   3
        ¬
    NODES LOCATED HERE ARE : 2.
         ¬ ¬¬¬   ¬
   6    01021*0001000000000000000000000000000000000000000000000000                  17

   7    01021*0001000000000000000000000000000000000000000000000000                  17

   8    01021*0001000000000000000000000000000000000000000000000000                  17
         ¬          ¬¬
    NODES LOCATED HERE ARE : 3.10.
                  ¬¬¬¬
   9    00021*0001030000000000000000000000000000000000000000000000                  19
                   ¬

  10    00021*0001000000000000000000000000000000000000000000000000                  16
           ¬     ¬¬                                        ¬¬
    NODES LOCATED HERE ARE : 4. 9.28.
                             ¬                             ¬¬
  11    00001*000000000000006000000000000000000000000000000000000                   19

  12    00001*000000000000006000000000000000000000000000000000000                   19

  13    00001*000000000000006000000000000000000000000000000000000                   19

  14    00001*000000000000006000000000000000000000000000000000000                   19

  15    00001*000000000000006000000000000000000000000000000000000                   19
            ¬¬
                ¬     ¬
  16    0000000040000*000006000000000000000000000000000000000000                    20
```

```
17   000000004000000*000060000000000000000000000000000000000         20

18   000000004000000*000060000000000000000000000000000000000         20

19   000000004000000*000060000000000000000000000000000000000         20
                    ¬¬¬
 NODES LOCATED HERE ARE : 5. 6.
            ¬      ¬¬
20   000000084000000000060000000000000000000000000000000000         18

21   000000084000000000060000000000000000000000000000000000         18

22   000000084000000000060000000000000000000000000000000000         18

23   000000084000000000060000000000000000000000000000000000         18

24   000000084000000000060000000000000000000000000000000000         18

25   000000084000000000060000000000000000000000000000000000         18
                        ¬
                     ¬
26   000000084000000050000000000000000000000000000000000000         17

27   000000084000000050000000000000000000000000000000000000         17
                     ¬
 NODES LOCATED HERE ARE :12.
                         ¬
28   000000084000000000007000000000000000000000000000000000         19

29   000000084000000000007000000000000000000000000000000000         19
                         ¬
 NODES LOCATED HERE ARE :11.
       ¬
30   006000084000000000000000000000000000000000000000000000         18

31   006000084000000000000000000000000000000000000000000000         18

32   006000084000000000000000000000000000000000000000000000         18

33   006000084000000000000000000000000000000000000000000000         18

34   006000084000000000000000000000000000000000000000000000         18
            ¬
           ¬              ¬
35   006000504000000000000300000000000000000000000000000000         18
            ¬
```

```
                            ¬
36   0060005000000000000000340000000000000000000000000000000000                                        18

37   0060005000000000000000340000000000000000000000000000000000                                        18

38   0060005000000000000000340000000000000000000000000000000000                                        18
                           ¬                        ¬¬
 NODES LOCATED HERE ARE :27.
                                                    ¬¬
39   0060005000000000000000040000000000000000000000000000000000                                        15
       ¬                    ¬
 NODES LOCATED HERE ARE : 7.14.
                       ¬      ¬
40   0000005000000000003000000200000000000000000000000000000000                                        10

41   0000005000000000003000000200000000000000000000000000000000                                        10
                              ¬
 NODES LOCATED HERE ARE :15.
                               ¬¬¬
42   0000005000000000003000000012500000000000000000000000000000                                        16
                        ¬
 NODES LOCATED HERE ARE : 8.
                        ¬
43   0000005000000000000100000012500000000000000000000000000000                                        14

44   0000005000000000000100000012500000000000000000000000000000                                        14
           ¬            ¬
 NODES LOCATED HERE ARE :13.
                             ¬
45   0000000000000000000000001012500000000000000000000000000000                                         9
                                ¬    ¬  ¬
 NODES LOCATED HERE ARE :17.22.30.
                                     ¬¬¬¬
46   0000000000000000000000001010500009300000000000000000000000                                        19
                                 ¬
                                              ¬
47   0000000000000000000000001010000009300000020000000000000000                                        16
                             ¬ ¬  ¬¬¬ ¬
 NODES LOCATED HERE ARE :16.18.23.24.
                                  ¬¬¬    ¬
48   0000000000000000000000000000000000300000020000000000000000                                        20
                                              ¬ ¬
                                                ¬        ¬
49   0000000000000000000000000000000000300000000000000000100000                                        19

50   0000000000000000000000000000000000300000000000000000100000                                        19
                                       ¬ ¬¬¬
 NODES LOCATED HERE ARE :19.20.21.25.
                                           ¬¬¬¬ ¬ ¬
51   0000000000000000000000000000000000000000310405000000100000                                        14

52   0000000000000000000000000000000000000000310405000000100000                                        14
                                              ¬ ¬
```

```
53   0000000000000000000000000000000000000003000050000000100000                                           9

54   0000000000000000000000000000000000000003000050000000100000                                           9
                                                 ¬¬      ¬
 NODES LOCATED HERE ARE :26.31.
                                                  ¬¬      ¬
55   0000000000000000000000000000000000000003000000300000080000                                          14

56   0000000000000000000000000000000000000003000000300000080000                                          14
                                            ¬      ¬      ¬
 NODES LOCATED HERE ARE :29.32.33.
                                                        ¬  ¬¬
57   0000000000000000000000000000000000000000000000000007002100                                          10

58   0000000000000000000000000000000000000000000000000007002100                                          10

59   0000000000000000000000000000000000000000000000000007002100                                          10
                                                        ¬  ¬¬
 NODES LOCATED HERE ARE :34.
                                                             ¬
60   0000000000000000000000000000000000000000000000000000000030                                           3

61   0000000000000000000000000000000000000000000000000000000030                                           3
                                                             ¬
 NODES LOCATED HERE ARE :35.
                                                              ¬
62   0000000000000000000000000000000000000000000000000000000004                                           4
                                                              ¬
 NODES LOCATED HERE ARE :36.

63   0000000000000000000000000000000000000000000000000000000000                                           0
```