# AN ENUMERATION METHOD FOR PROJECT SCHEDULING

by

FAHRETTİN TUĞHAN ULUDAĞ ALPAT

B.S. in C.E., İSTANBUL TEKNİK ÜNİVERSİTESİ, 1982

Submitted to the Institute for Graduate Studies in Social Sciences in Partial fulfillment of the requirements for the degree of

Master of Art

in

Business Administration



Boğaziçi University

1985

AN ENUMERATION METHOD
FOR PROJECT SCHEDULING

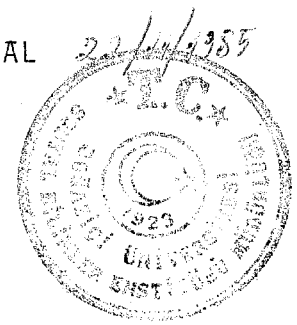
APPROVED BY

Doç. Dr. Seyhan Tuğcu
(Thesis Supervisor)

Prof. Dr. Mustafa Dilber

Doç. Dr. Ceyhan Uyar

DATE OF APPROVAL 22/06/1955



*ACKNOWLEDGEMENTS*

*I wish to express my sincere gratitude to my thesis supervisor, Doç.Dr. Seyhan Tuğcu, for the invaluable guidance she has offered me during this study.*

*My deep gratitude also goes*

*to Prof. Dr. Mustafa Dilber and Doç. Dr. Ceyhan Uyar for their criticism and constructive suggestions;*

*to Sabiha Çelik for her devotion and skill in writing the manuscript;*

*to all the personnel of the Computer Center for their help during my programming.*

*ABSTRACT*

*This thesis considers a scheduling problem that has precedence constraints of a general form as in CPM/Pert problems. It consists of two stages:*

*First, the schedule of activities which minimizes total resource usage (optimum schedule) is generated. For this purpose, an exhaustive enumeration procedure is developed.*

*Second, for this schedule, optimum resource requirements and overtime schedule that minimizes the total of regular and overtime work along with hiring and firing costs are determined. To accomplish this, another exhaustive enumeration technique is used to find the set of jobs that uses overtime.*

*Following are obtained:*

* *Total resources used (with and without overtime) for each unit of time in the project,*
* *Set of activities using overtime,*
* *Total cost in case of regular work,*
* *Minimum cost in case of overtime work.*

## ÖZET

*Bu tez birbirini izleyen işlerden oluşan genel CPM/Pert türü projelerde faaliyetlerin zamanlanmasını incelemektedir.*

*İlk olarak toplam kaynak kullanımını en aza indirecek zamanlama bulunur. Bu amaçla bir teker teker değerlendirme yöntemi (enumeration) geliştirilmiştir.*

*Daha sonra ilk aşamada bulunan zamanlamaya ek olarak normal ücret, fazla mesai (overtime) maliyeti, işe alış ve işten çıkartma maliyetleri toplamını en aza indirecek şekilde bir mesai planlaması yapılmaktadır. Burada da yine bir başka teker teker değerlendirme metodu kullanılarak fazla mesai ile yapılan işler grubu belirlenmektedir.*

*Sonuçta şunlar elde edilmektedir:*

* *Projenin birim zaman dilimlerine ait toplam kaynak kullanımları,*
* *Fazla mesai ile yapılan işler grubu,*
* *Normal çalışmanın toplam maliyeti,*
* *Fazla mesaili çalışmada minimum maliyet.*

CONTENTS


| | |
|---|---|
| Acknowledgements | iii |
| Abstract | iv |
| Özet | v |
| List of figures | viii |
| Introduction | 1 |
| Ch. I | |
| Network scheduling with limited resources | 2 |
| A- Heuristic approaches to scheduling | 2 |
| B- Optimal Techniques | 4 |
| B1- General problem formulation | 4 |
| B2- Optimal techniques-a review | 6 |
| Ch. II | |
| A- Algorithm of the procedure | 14 |
| B- The comparison | 18 |
| Ch. III | |
| Data preparation | 21 |
| A- Network parameters | 21 |
| B- Activity parameters | |
| Ch. IV | |
| Details of computations | 27 |
| A-CPM computations | 27 |
| B- Exhaustive enumeration | 27 |
| B1- Scheduling (the first stage) | 27 |
| An Example | 29 |
| B2- Enumeration for overtime (second stage) | 33 |

Ch. V

A- Test problems 37

B- The computer code 52

C- The program and computational experience 61

Ch. VI

A- Conslusions 62

B- Recommendations for further work 63

References 65

LIST OF FIGURES


| Figure | | Page |
|---|---|---|
| 1 | Example of network cuts | 13 |
| 2 | Flowchart of the algorithm | 16 |
| 3 | The numbering | 21 |
| 4 | Example project | 30 |
| 5 | Computation steps | 32 |
| 6 | Project 1 | 38 |
| 7 | Project 2 | 39 |
| 8-19 | Bar charts and histograms | 40-51 |

# *INTRODUCTION*

*The growing complexites of today's projects have demanded more systematic and more effective planning techniques with the objective of optimizing the efficiency of executing the project. Efficiency here means accomplishing the utmost reduction in the time required to complete the project while accounting for the economic feasibility of using avaliable resources.*

*The primary focus of this research has been the development of an iterative procedure which provides an optimum solution to the 1 scheduling problems with technological (precedence) and resource constraints. The existing procedures require large memory computers and excessive computer time to solve prablems of realistic size (10)-pp.1170-1171 . Therefore the objective has been to attack this problem by developing an iterative procedure that could be used on small core computers while at the same time requiring at least a reasonable amount of execution time.*

# CHAPTER I

## NETWORK SCHEDULING WITH LIMITED RESOURCES

The problem generally encountered in the literature is scheduling the activities of a project network to minimize project duration under conditions of precedence constraints and resource requirements.

There are mainly two types of solution techniques mentioned in the literature related to this topic: (1) Heuristic approaches and (2) optimal techniques.

### A) HEURISTIC APPROACHES TO SCHEDULING

A most standard type of constrained-resource project scheduling problem is project duration minimization under fixed apriori resource requirements. This is generally encountered when the performance of project activities depend on resources that are limited in nature. Because of these situations, the activities must be sequenced and as a result, it becomes no more possible to finish the project within the time that is determined from standard critical path analysis with no resource constraints.

The problem stated above is a combinatorial one and therefore optimal techniques of mathematical programming have been satisfactory on at most only for smaller sized project networks.

"The problem of how to allocate limited resources among

the activities of a project is yet largely unsolved in a manner conducive to rapid computation. While such allocation problems can be represented as integer and 0-1 programming problems, the computation times involved often render the methods impractical or infeasible. Huber (11) concludes that 0-1 programming is an ineffective means of solving this problem, and others (12), (13) dismiss integer programming as an alternative because of excessive computation times.

Recent research efforts have concentrated on enumerative methods for solving this problem. Johnson (5) presents a branch and bound approach for solving the single-constrained resource, project scheduling problem. He indicates that his algorithm should be practical for projects of up to 50 activities". (9)

The difficulties of attacking this problem have led to development of heuristic procedures that produce "good" feasible solutions instead of optimal ones. These rules are nothing more than assigning activity pirorities on making the activity scheduling decisions which are necessary for solving resource conflicts. One heuristic approach example is looking first at the activities that have the most slack and attempting to delay them as long as possible without delaying the completion forthe entire project.

From a large variety of heuristic procedures some are listed in (2), and most of the complicated heuristics used in commercial network analysis programs are not published.

Other procedures are published and descriptions of common rules are given (3). This kind of heuristic based scheduling procedures for constrained-resource scheduling are today the most practical ways of obtaining workable solutions to large complex problems that are encountered. When they are used in connection with certain optimization routines (4), (5), (6),

the resulting preliminary starting solution can be used for further requirements with less effort.

Because of their nature, heuristic sequencing rule produce solutions with varying degress of "goodness", depending on the problem. Here, by " goodness " the ability to produce "reasonable" schedules for actual projects is meant, and one means of checking this is by the measure of the project duration.

There have also been reported comparisons of heuristic based solutions relative to the optimum solution. In one such reported comparison, authors Pritsker, Watters and Wolfe (7) compared two heuristic based schedules with the optimum based schedule for a small multiproject example containing projects of about three jobs each. In this case, a minimum-slack-first rule gave the optimum schedule.

B) OPTIMAL TECHNIQUES

B.1.) GENERAL PROBLEM FORMULATION:

In summary, a general formulation which is used by a number of authors (6), (7), (9) and (10) is as follows:

$$\text{Minimize } f_N \tag{1}$$

$$\text{subject to } \max_{nEP_j} \{fn\} + dj \leq fj \qquad j=1,\ldots,N \tag{2}$$

$$\sum_{jES_t} r_{jk} \leq R_{kt} \qquad \begin{matrix} k=1,\ldots,K \\ t=1,\ldots,PC \end{matrix} \tag{3}$$

$f_j$ : An integer variable representing the current finish time for job j.

N : Total number of jobs in the network. It is also the

number of the unique ending job.

$P_j$ : The set of all immediate predecessors of job j.

$d_j$ : The time duration of job j.

k : A specific resource. k=1,....., K.

K : Number of resource types.

$r_{jk}$: Amount of resource k required by job j each period job j is active.

$R_{kt}$: Total amount of resource k available in time period t.

PC : The critical path completion time for the project.

$S_t$ : The set of all jobs active in time period t.

The objective of this formulation is to minimize the terminal job's (N) completion time which also is equivalent to minimizing project duration (1). The first constraint (2) is the one involved with the precedence relationships which means that a job is considered for assignment only if all of its predecessors have been assigned and the latest finish time of all these predecessors is less or equal to the start time of that particular job. This is an obvious result of the construction logic of the project networks that is if a job is a successor of a group of activities, its duration may not overlap with any of the predecessor job's durations when they are performed. Finally the rule (3) states that resource usage must not exceed resource availability in any given time period. The set of all jobs active concurrently must use a total amount of resurce that is less or equal to the allowed or available quantity of that particular resource type in any given time period. In the formulation of the constraint (3) it is also implicitly assumed that a resource can be assigned to only one activity at a time. Furthermore, for all the jobs in the network once a resource $r_{jk}$ has been assigned to activity j, that resource $r_{jk}$ must remain assigned to job j as long as job j is active.

When the scheduling problem is formulated in this way, it is not detailed enough to be solved by integer programming

techniques because the elements of set $S_t$ are not included in the formulation. One way of overcoming this is the use of 0-1 integer variables such as:

$x_{jt}$ : a variable which is 1 if job project is completed in period t; 0 otherwise. $x_{jt}$ is not a variable in all periods, since it is 0 for $t < ES_j$ and $LF_j < t$.

$x_t$ : a variable which is 1 in period t if all jobs of project have been completed by period t; 0 otherwise.

Although the above mentioned supplementary variables make the problem yielding to solution technique of integer programming, they at the same time increase the number of variables and constraints required in the formulation. But this inconvenience must be regarded as one of the burdens of optimum solution techniques. Once these techniques are implemented, they at the same time bring their inconveniencies in some way.

B.2) OPTIMAL TECHNIQUES - A REVIEW

Owing to the mathematical complexities of the constrained resource scheduling problem's combinatorial nature, the methods developed up to date, as stated above are limited only to relatively small projects. But with the condensation of the big networks they become a very effective tool for managers. Condensation means representing a group of related activities by a single activity.

Here in chronological order some of the optimal procedures will be summarized and necessary details will be given:

a) In his research A.M. Geoffrion (8) gives a hybrid technique of Balasian implicit enumeration with integer linear

programming and its applications to some network problems.

b) Linus Schrage (6) considers a scheduling problem that has both precedence and resource constraints and gives an enumerative technique for generating all active schedules for this problem. Based on this technique, he describes a branch and bound method for implicitly enumerating all schedules and determining the optimum.

c) Pritsker, Lawrance, and Wolfe (7) in their joint effort have developed a general technique to solve multiproject and job shop scheduling problems with 0-1 programming. Their approach considers three different objectives:

1- Minimize total throughput time for all projects,
2- Minimize the time by which all projects are completed,
3- Minimize total lateness or total lateness penalty for projects.

They developed the formulation to meet the following constraints:

1- Limited resources,
2- Precedence relations between jobs,
3- Job splitting possibilities,
4- Project and job due dates,
5- Substitution of resources to perform the jobs,
6- Concurrent and nonconcurrent job performance requirements

d) Patterson and Huber (9) give a method to produce minimum duration schedules for the resource constrained project scheduling problems that incorporates a bounding technique similar to those used with branch and bound procedures in conjuction with 0-1 programming. The algorithms developed consist of examining the feasibility of a series of 0-1 integer programming problems rather than solving one 0-1 problem optimally.

formulation of the problem is:

(4) Maximize :

$$\sum_{t=1}^{T} X_t ,$$

Subject to

(5) $\sum_{t=e_j}^{l_j} X_{jt} = 1$ , $j=1,2,\ldots,N$ job completion constraints,

(6) $X_t \leq (1/N) \sum_{j=1}^{N} \sum_{q=e_j}^{t-1} X_{jq}$, $t=e, e+1, \ldots, T$ project completion constraints,

(7) $\sum_{t=e_m}^{l_m} tX_{mt} + dn \leq \sum_{t=e_n}^{l_n} tX_{nt}$ , $m \leq n^*$, precedence constraints,

(8) $\sum_{j=1}^{N} \sum_{q=t}^{t\ d_j-1} r_{jk} X_{jq} \leq R_k$ , $t=1,\ldots,T$ $k=1,2,\ldots,K$ resource constraints.

(* denotes" immediately precedes" )

The definitions are such that:

Subscripts;

j : job number, $j = 1,2,\ldots,N$,

k : resource number, $k = 1,2,\ldots,K$,

t: time period, $t = 1,2,\ldots,T$

(T is the last period in scheduling horizon)

Constants:

CP : the oritical path length of the project (in time periods t),

$d_j$ : duration of job j (in integer time periods t),

$e$ : earliest possible period by which the project could be completed,

$e_j$ : the earliest possible period in which job j could be completed,

$l_j$ : the latest possible period in which job j could be completed,

$r_{jk}$ : amount of resource k required during each time period of job j's duration,

$R_k$ : amount of resource k available during each scheduling period.

Variables:

$X_{jt}$ = 1 if job j is completed in period t,
= 0 otherwise, ($X_{jt}$ 0 for t $e_j$ and t $l_j$)

$X_t$ = 1 in period t if all jobs have been completed by period t,
= 0 otherwise, ($X_t$ 0 for t e)

The minimum bounding algorithm:

The minimum bounding algorithm initially establishes a lower bound T on the length of a schedule and examines the resulting 0-1 programming problem (based on a maximum schedule span of T periods) to see if it is feasible. If the 0-1 problem is feasible, the algorithm terminates. If the 0-1 program is not feasible, T is increased by one time unit and another 0-1 program is set up using a maximum schedule span of T+1 periods. The resulting 0-1 program is then examined for feasibility. This process is continued until feasibility in a 0-1 program is attained.

An outline of the algorithm follows:

1. Determine T to be used in (4)-(8) as follows,

$$W \quad \max \left\{ CP, \left[ \max_k \frac{1}{R_k} \sum_{j-1}^{N} r_{jk} d_j \right] \right\},$$

$T = \lceil W \rceil +$ where $\lceil W \rceil +$ denotes the smallest integer $\leq W$,

T is the lower bound on the length of a schedule.

2. Set up a 0-1 integer programming problem using (4)-(8) with T as the maximum finish period. Call this problem P.

3. Is P feasible ?

Yes. Terminate 0-1 program and return the feasible solution. Since a schedule of length less than T time periods is not feasible, the shortest duration schedule has been found.

No. Replace T by T+1 and return to step g.

e) Finally authors Patterson and Talbot (10) describe an integer programming algorithm for allocating limited resources to competing activities of a project such that the completion time of the project is minimized among all possible alternatives. The procedure consists of a systematic evaluation of all possible job finish times. To limit the number of task assignments, which have to be explicitly exaluated, a device called a "network cut" is used which removes from consideration the evaluation of job finish times which can not lead to reduced project completion time.

In the method, precedence and resource restrictions are met by using an immediate predecessor array and two resource related arrays. In this manner they avoid explicitly formulating the constraints. The resulting formulation has a low computer core requirement.

Integer Formulation:

The model assumes that resource availabilities, requirements, activity durations ($d_j$), the set of all immediate predecessors, number of resource types and number of activities (N) are all integer valved and known with certainty.

To begin, jobs in the project are numbered ( hence

considered for augmentation ) such that if n is a member of the predecessors' set of job m, then n<m. If a heuristic solution HP is not known for the completion time of the project, they set $HP = \sum_{j=1}^{N} d_j$. The latest possible finish time of job j is:

$$u_j = LF_j - 1 + (HP - LF_N)$$

$LF_j$: latest finish time of job j.

The algorithm begins by assigning job 1 to its earliest completion time and subtracting its resource requirements from the resource available for the duration it is active. Next, job 2 is assigned to its earliest feasible completion time. In general, precedence relationships are maintained by selecting for assignment the lowest numbered job that has not been assigned a feasible completion time. The described job numbering rule insures that a job is considered for assignment only if all of its precedessors have been assigned.

The resource remaining array is searched for the period job 2 is active and its requirements are subtracted from the array. This assignment process is continued for jobs 3,4,....,N until either job N is assigned a completion time or some job (j*) N cannot be assigned due to resource infeasibility.

If job N is assigned a completion time $f_N$, an improved solution to the problem has been found. It is $HP-f_N$ periods shorter than the best solution found before. The improved solution is stored and upper bounds ($u_j$) are all reduced by $HP-f_N$ units.

If in the augmentation phase of the algorithm, a job j*can not be assigned a resource feasible completion time less than or equal to $u_j$, then backtracking occurs. An attempt is made to reassign job j*-1 to the earliest feasible completion

time greater than $f_{j^*-1}$. If this is possible, then the assignment process continues with job j*. If it is not possible to reassign j*-1 either because of resource infeasibility or because $f_{j^*-1} = u_{j^*-1}$, then backtracking proceeds to job j*-2.

The use of Network Cuts in Fathoming Partial Solutions:

In order to improve the basic enumeration procedure described above a fathoming technique is employed that reduces solution times by identifying, early in the enumeration procedure, partial schedules that cannot possibly lead to improved solutions.

A network cut is an integer time period C which is $1 \leq c \leq HP$ and serves two purposes: (1) It is used to identify when schedule elimination rules can be applied, (2) It is a parameter used in the formulation of the elimination rules. A job is being cut by c if c is between its early start and latest possible finish time ($u_j$). An integer time period qualifies as a cut if the two conditions hold.

1. There exists at least one job with their early start times equal to that period,
2. There is no preceeding job that has an early start time smaller than that period.

The figure below represents some possible cuts for a 21 job project. The cuts are at times 1,10,11,14,20,21,27 and 30.

The method expedites backtracking by exploiting the precedence and numbering relationships between jobs. The procedure eliminates the need to augment certain variables during the solution procedure, which is equivalent to eliminating from explicit considerations many partial schedules. Partial schedules identified by network cuts are exaluated for their

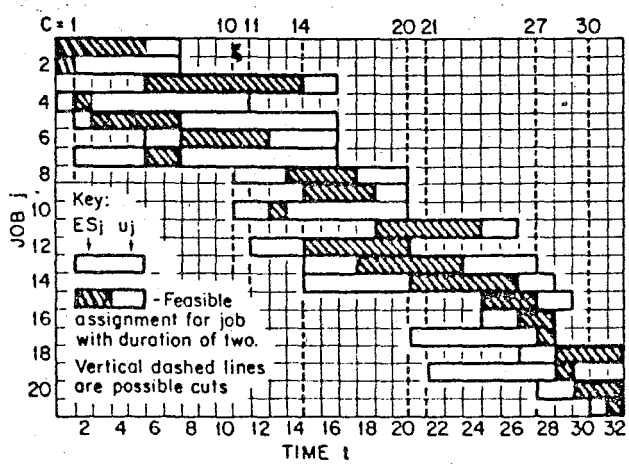



FIGURE 1. An example of network cuts

potential to lead to improved solutions for the entire project, during the augmentation by applying formulas to show the inferiority. If a partial schedule can possibly lead to an improved solution, then augmentation continues. If the partial schedule cannot improve upon the performance of previously generated partial schedules, then backtracking proceeds.

The reason of using network cuts to form partial schedules in the beginning of the algorithm is this:

The cut marks a period for the jobs of the partial schedule identified by this cut, that if any one of them passes this period to finish after the cut, then the passed job starts to share resources with the jobs of the time frame between network cut and the project completion time. The increased resource totals of the period between network cut and the project completion time force the jobs situated in this time frame to move towards the end of the project and increase project's duration. Therefore, network cuts and the partial schedules marked by these cuts are the sensitive parts of the project which play a strong role in determining its duration and because of this property are checked specially.

# CHAPTER II

The objective is to minimize the total resource requirement of the set of jobs active in any one period.

Precedence relationships of the activities are taken into account by setting a specific job's start time equal to or greater than the maximum of all its immediate predecessor jobs' finish times.

Resource requirements, activity durations and finish times of activites are all positive.

## A) ALGORITHM OF THE PROCEDURE

1- Schedule the jobs to their early start times, add the resource requirements of the jobs for each working period. Find the maximum of these resource requirement summations. Save this maximum as the "maximum resource requirement" and the corresponding schedule in an array "best schedule".

2- Change the schedule of at least one activity to obtain the next scheduling combination.

3- Add the resource requirements of the jobs for each working period. Upon completion of each working period summation check if this sum of resource requirements is less than or equal to the "maximum resource requirement" which was found before. If it is less, go to (a) if not, go to (b).

a) Continue the summations until the end of the project. Find the new maximum resource requirement which is smaller than the one found in the last step. Save the new schedule of activities in the best schedule". Discard previosly saved schedule. If there are any scheduling combinations left to be enumerated, go to (2), if not go to (5).

b) Mark the period in which the new total exceeds the saved maximum resource requirement. The set of jobs that are active in this period constitute an inferior partical schedule. Save this inferior partial schedule. (For an inferior partical schedule, we do not compute resource totals, therefore significant computation time is saved). Continue to move the jobs of the project to obtain the succesive scheduling combinations until at least one of the jobs of the saved inferior partial schedule leave the marked period.

4- If, there are any scheduling combinations that are not enumerated yet, go to step (3). If there are not, go to (5).

5- Terminate the enumeration. The last saved "maximum resource requirement" is the optimum solution and the corresponding schedule is the "best schedule" which was saved with the optimum solution.

In addition to these, the following are assumed to be valid for this problem:

Each job may have only one resource type. In practice this would mean that the most important resource is selected for optimization.

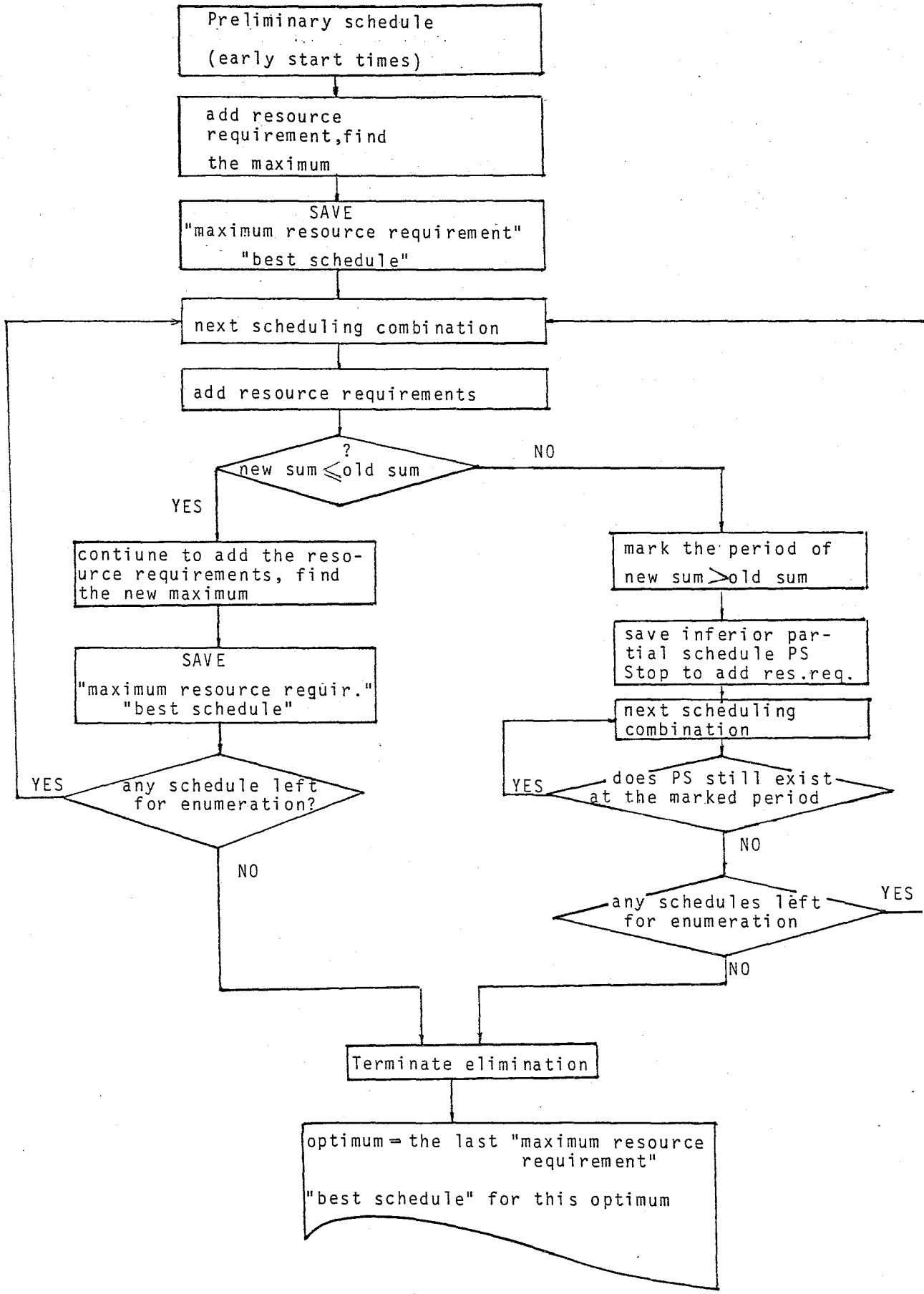



FIGURE 2 Flowchart of the algorithm.

Job splitting is not allowed. However if it is necessary to split a job, this can be done apriori and the job can be represented by two seperate jobs. For example, if a certain rersource is in use only during the first p periods of the job where p $d_j$, then the job is treated as two sequenced subjobs with different resource requirements and with durations of p and $d_j$-P respectively.

Multiple inital or terminal jobs can be considered in formulation as all the other activities and therefore the rule "networks may have only one inital activity (with no predecessor) and only one terminal activity (with no successor)" is not required. A project may have more than one starting point. Supposing this is the case, the computations are artificially brought together to single inital or terminal activities.

Concurrencey and noncurrency: A concurrency constraint on jobs m and n, means that they must be performed simultaneously. It can be obtained by (1) combining resource requirements and treating m and n as a single job or, (2) making necessary adjustments on the network as addition of a dummy activity between the heads or tails of two concurrent activities.

A nonconcurrency constraint on jobs m and n means that they must not be performed simultaneously, but permits them to be performed in any order. This constraint can be met by adjusting the network so that the nonconcurrent activities are not parallel but rather in serial order.

Linearity: In the cost minimization stage of the algorithm, the hiring and firing costs are considered to be fixed for each unit of resource type. In other words, hiring the 100th unit of resource A has the same additional cost as hiring the first unit of the same resource. If this is not the case, necessary adjustments must be added to the program to take the differences of costs into account. But in this case the basic algorithm does not change and the same algorithm can be applied

to both cases. The only change would be the modification of the computer program to take the cost changes in account.

Inflation: All the costs of the project ( unit cost, hiring-firing costs and overtime costs) are considered to be fixed over time. If the project in question has long duration, than this assumption loses its strength and the results of fixed cost computations would be rather misleading. To overcome this deficiency, either discrete or continuous change of the costs can be easily introduced to the related part of the computer program. This adjustment is not made in the basic form of the program because it does not affect the enumeration algorithm but increase the complexity of the computation structure and is not considered to be an important parameter for the development and application of the algorithm.

Productivity: Productivity of the resources is thought to be constant in the overtime period as they are in normal working periods, In reality, this assumption may not hold especially for the excessively overtime loaded cases. When planning a project, this requisition must be kept in mind.

B) THE COMPARISON

The procedure used in this research differs from other optimal techniques. All these optimal techniques attack the scheduling problem to seek a minimum duration solution to the project by satisfying the multiple resource constraints.

In this research however, there is no definite resource constraint that is used in the enumeration procedure, but rather all the resources are considered to be scarce and their use is tried to be minimized.

Another difference comes from the meaning of multiple resource concept. In other techniques a single job may use more

than one resource type in different amounts. But in this research each activity may use only one type of resource. Nevertheless, different jobs may use different resources and there is no limit to the number of unique resource types. Each resource group is taken as a subproject in the enumerative process.

The project's duration which is the amount of time passed from the beginning of the project until the end of the last job, does not play an open role in the mathematical structure of the problem. It is duration that is found by regular CPM computations and retained the same throughout the enumeration. But in other methods, it is tried to be minimized by scheduling the activities so that the constraints are not violated.

Also the critical activities remain as they are and no attempt is made to alter neither start, time nor resource usages for these. Only the non-critical activities are in question and they are shifted to and fro to seek improvements from the preliminary schedule.

This procedure uses the same name "network cut" as the method explained by authors Patterson and Talbot. Both uses are similar in calling a working period as a network cut which passes over some jobs of the project in the Gantt bar chart. But the two methods do not use network cuts in the same way.

Patterson and Talbot's cut is determined once the network is given with the CPM computations. Therefore the number of network cuts are known with certainity before the enumeration. If at any period in the project's life a job has an early start time that comes before than all the independent successor jobs' early start times, the cut occurs. It cuts through some of the activities between their early start and latest possible finish times and they form a partial schedule.

The network cuts used in this study on the other hand, cannot be found before the enumeration starts. They are developed during the elimination by the method of "bounding". A cut occurs in enumeration if a certain schedule results at a greater total resource requirement than the best available total in a specific period in the life of the project. The jobs marked by network cut are moved to leave this period and elimination advances accordingly.

In the last stage, total cost minimization (made of regular, overtime, hiring and firing costs) using overtime scheduling is considered in addition to determination of optimum schedule of the first stage. Other methods mentioned do not attempt to consider costs in that manner at all.

The method in question is an integer bounding technigue in essence and it searches for an optimum solution by systematic exhaustive evaluation of job slacks to meet its objectives of minimum resource usage and costs.

# CHAPTER III

## DATA PREPARATION

### A) NETWORK PARAMETERS

In the beginning, the basic parameters must be defined. Those are number of nodes, number of activities in the network and the number of dangling activities.

The beginning and ending points of activities are called "nodes". Synonyms are"event" and "connector". An event can be presented graphically by a numbered circle, as shown.

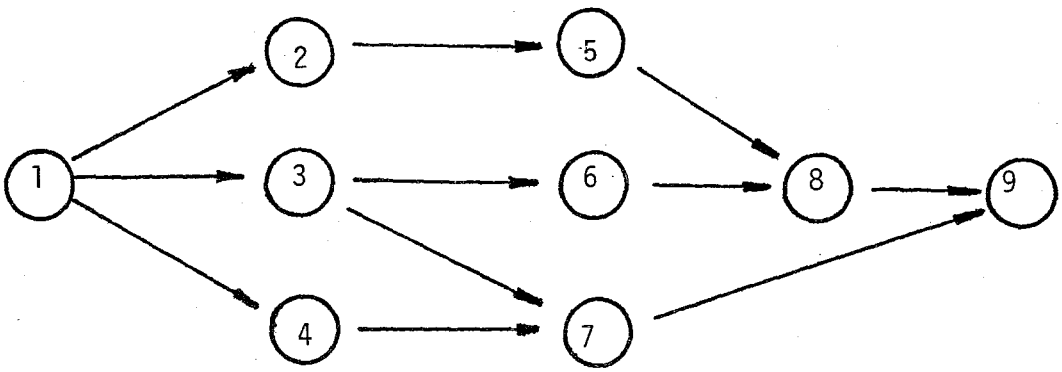



FIGURE 3 The Numbering.

In order to carry out the forward and backward CPM calculations and identifying the set of predecessor activities of each activity in the project, the jobs mut be ranked by a node numbering rule. In making the forward pass computations to

compute an earliest finish time for an activity we must have its earliest start time, which is equal to the latest finish time of its predecessor events. By ranking the activities, it is ensured that the necessary early finish times of preceeding activities are always computed prior to the computations of a given activity. With the activities placed in predecessor event groups they appear in the list according to their position in the network. To accomplish this, the following steps must be taken:

1) Number all the nodes without predecessors first by beginning with 1.
2) Go to the adjacent nodes which are not numbered yet.
3) Number them as long as all the predecessor nodes of any particular node are numbered.
4) Continue the steps until there are no more empty nodes left.

"Number of nodes" is equal to the largest number used during numbering steps.

The second parameter is "the number of activities" which can be found by a simple count of all the arrows of the network diagram. Dummy activities are also taken into account here.

The third parameter is the number of nodes without predecessors (excluding the unique beginning node of the project) and number of nodes without successors (excluding the unique terminal node).

Network parameters define the size and the character of the project. By charecter, two specific characters are called for:

* Single or multi-project network
* Project complexity which is: n.of activities/n.of nodes

B) ACTIVITY PARAMETERS

An activity is defined by five different parameters:

1. An initial number i (number of the tail node),
2. A terminal number j (number of the head node),
3. Mean activity time $d_{ij}$
4. Activity resource requirement corresponding to the mean activity time $r_{ijk}$,
5. Type of resource that is demanded by this activity k.

The initial number must always be less than the terminal number. The size of these numbers do not matter however

In estimating times for each activity, it is important to be as accurate and objective as possible. In time estimation step following assumptions must be used:

1. A normal level of man power, equipment and other resources must be assumed. At this stage, the effect of possible competition between two simultaneous activities for men, space or other resources must be ignored. Such conflicts will be considered, later.

2. Only one job at a time must be considered. For example, a sequence of work may involve delivery of equipment followed by its installation. The time decision for installation should not be affected by the possibility of a late delivery. To guard against bias, each job must be considered by itself. It may also be helpful to skip around from one part of the network to the other, rather than to follow the activities in sequence.

3. A normal work day and work week are assumed. Overtime is not considered in estimating the mean activity time. The known calendar completion dates must not be considered in time estimations. Otherwise, unconcious tries to adjust the durations to make them fit the time available will bias the decisions.

4. Consistent time units must be used. Any time unit (day, week, etc.) may be used but the same unit must be used throughout. If days are used, they must be either work days or calendar days.

Next, the activity resource requirement must be estimated in view of the mean activity time. Most of the rules given for time estimation also help to decide a correct resource requirement. When estimating the resource requirements itmust always be kept in mind that the time and resource requirement decisions along with the construction of the network are the keys to successful management which is expected from CPM/Pert. And accordingly, without them being correct, results of the enumerative process are useless.

At this point, one question remains to be answered. One may think that the procedure would be more powerful and realistic if the probabilistic time estimates rather than mean activity time estimates were used. But probabilities are too abstract concepts for the low levels of management to respond with correct and objective estimates in this country. Instead of dealing with unreliable inputs, they are discarded altogether. But if in the long run the organization succeeds at establishing a data gathering system including probabilistic measures, they may also be incorporated in the method described here.

The jobs which are performed with the same resource type must be grouped according to this particular resource type. While grouping, the priority must be given to the resource that

has the biggest cost contribution. That means the most costly resource type is the first resource type and the second most costly resource type is the second resource type. By acting in the same manner, each activity receives a resource type number in addition to its resource requirement.

The reason of assigning a priority to the most costly resource type by giving the resource type number 1 is as follows. The group of activities which use the first resource type are enumerated first and the procedure spends a greater effort and similarly time to minimize the total resource requirement in the first stage of the exhaustive enumeration. Not only free slacks (the free slack is defined by assuming that all the activities start as early as possible. In this case $FS_{ij}$ for activity (i,j) is the excess of available time ($= ES_j - ES_i$) over its duration ($= d_{ij}$), but also total slacks (the total slack $TS_{ij}$ for activity (i,j) is the difference between the maximum time available to perform the activity ($= LC_j - ES_i$) and its duration ($= d_{ij}$)) are used to attain the utmost reduction in total resource consumption. This is a logical necessity because the resource group in question is the most costly resource group. Decreasing one unit of this resource means the greatest reduction at cost to the project as a whole. So, these jobs are given the privilage to be shifted in their total slacks. But once they are scheduled, procedure schedules the second group of jobs which use the second most costly resource type to their slacks which are left by the first group. More specificly, if a resource 1 type activity precedes a resource 2 type activity, the first one is shifted in its entire total slack and once scheduled, the resource 2 type activity may only be shifted in its remaining slack if the two activities' slacks overlap. Going through the same logic, the highest resource type number is given to the least costly resource group and the enumeration procedure spends the least time to scheduling this group because their total slacks are limited the most.

The costs of the resource groups can be found from different sources. Accounting department of the organization as well as the union contracts and all other related documents are key sources of data for cost figures.

* Unit cost is the money spent for each unit of resource in question which is used for a unit working period,
* Overtime cost is the unit period cost of resource that is used for overtime.
* Hiring and firing costs are costs of hiring or firing one unit of resource in question.
* Overtime duration is the duration of overtime choosen for each working period of the project. Overtime duration must be choosen in such a length that the constant productivity of the unique resource can be attained. Otherwise, prolonged overtime scheduling may violate constant productivity assumption.

# CHAPTER IV

## DETAIL OF COMPUTATIONS

### A) CPM COMPUTATIONS

Basic CPM computations are done using the matrix method explained by Enver Çetmeli in his book "Yatırımların planlanmasında Kritik Yörünge (CPM) ve Pert Metodları" in pages 29-36. Forward and backward pass computations are carried out with this method and early start, late completion times of the jobs are thus found. The other CPM results are obtained from these figures by using activity times.

### B) EXHAUSTIVE ENUMERATION

### B.1.) SCHEDULING (THE FIRST STAGE)

The developed procedure systematically evaluates (enumeration) all the possible job finish times for each activity in the project. The resulting resource requirements are computed for each schedule. From all the possible scheduling combinations, the one that minimizes the resource usage for the entire project (or subproject) is choosen for further cost considerations.

The exhaustive enumeration procedure is carried out in such a systematic way that it permits the use of an elimination acceleration technique by so called "bounding". The efficiency of computations can be enhanced by introducing this concept. Bounding allows for the fact that if a solution of a subproblem yields a worse objective value than the one associated with the best available solution, it does not pay to explore the subproblem

any further. In this case the subproblem is said to be "fathomed" and may be henceforth deleted. In other words, once a feasible solution is found, its associated objective value can be used as a (upper in case of minimization and lower in case of maximization) "bound" to discard inferior subproblems.

During the enumeration, the first combination of jobs is taken as the best schedule with its corresponding resource requirements for each working period. The maximum of these resource requirements is found and it is saved as the upper bound for the subsequent scheduling combinations. The first schedule here ends and one of the activities of this schedule is moved one working period forward or backward to have the second scheduling combination. Here, if any one of the total resource requirements exceed the previously saved upper bound, we have a network cut for the working period in which the upper bound was exceeded. The network cut is the name of period that passes through some of the activities of the project (an inferior partical schedule) when they are depicted in the Gantt bar chart. Once this inferior partial schedule is found as subset of the second scheduling combination, it is unnecessary to evaluate the second combination any more and the third combination is checked.

If in the third scheduling combination, the same set of jobs that took part in the inferior partial schedule (being marked by the network cut) still exist, this third scheduling combination is discarded and the fourth combination is obtained to be checked again. Thus the enumeration jumps forward until at least one of the jobs of this inferior partial schedule leaves the time period of network cut. By doing so, it is made sure that the resource requirement which was made above the best available solution is reduced for the period which is marked by the network cut. Then the enumartion contines until another inferior partial schedule is determined.

Jobs are combinatorially moved, corresponding resource requirements for the working periods are computed end whenever

a better solution is reached, the old feasible solution is replaced by this better one. At the end the last saved schedule is the optimum solution of this scheduling problem.

An Example:

A small project consisting of five activities is given here (Figure 3) along with the first stage computation steps to show the explained part in detail and present the computational logic of the problem.

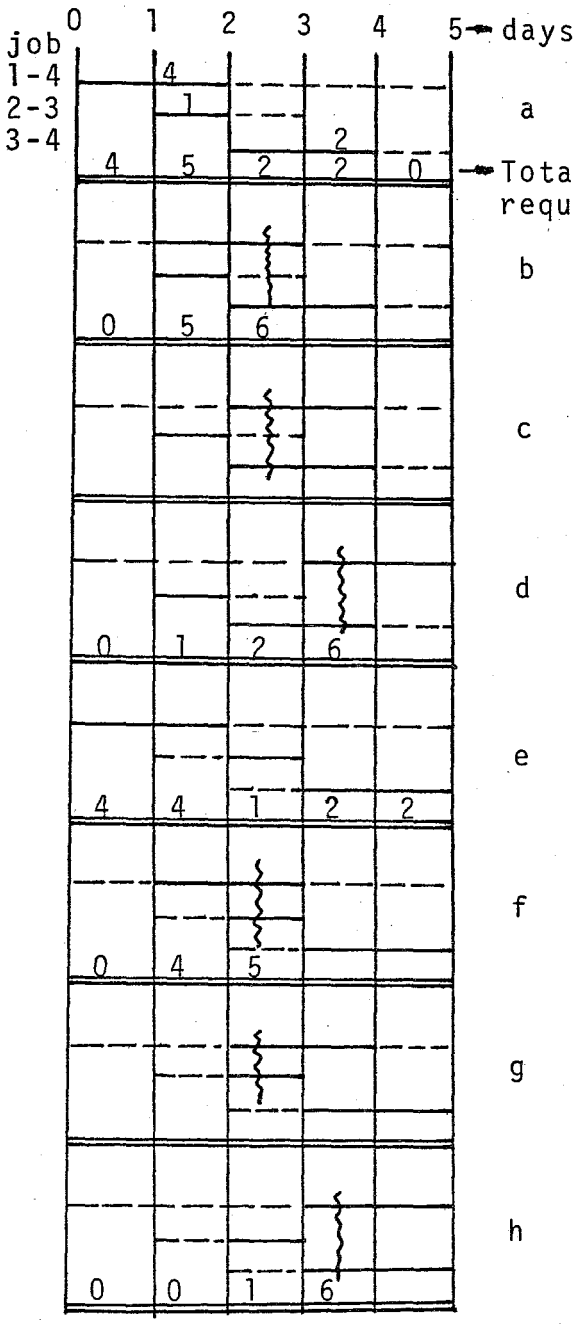
First, the jobs are scheduled to their early start times and corresponding total resource requirements are counted (Figure 3a). The maximum total resource requirement for the first schedule is 5 in period 2 of this figure and the schedule is saved with its maximum requirement which is 5 units of resource.

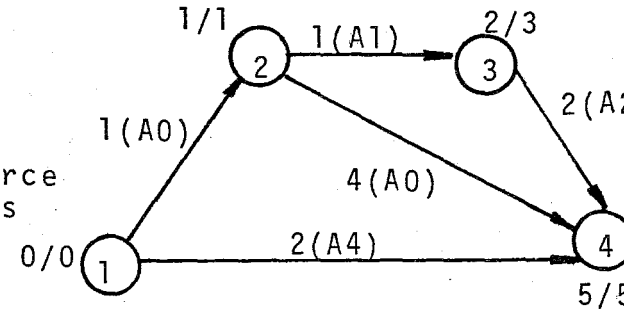
In Figure 3 b, until we come to the third period the best solution of 5 units of resource rquirement was not exceeded. But in the third period it is exceeded and therefore we have a network cut in this period. At least one of the jobs 1-4 or 3-4 must leave this priod so that the total resource requirement 6 which is above of our best solution 5 can be decreased.

In Figure 3 c, job 1-4 is moved one period forward but it is still cut by the third period. Therefore there is no need to count the resource totals. The schedule in this figure is inferior and must be eliminated.

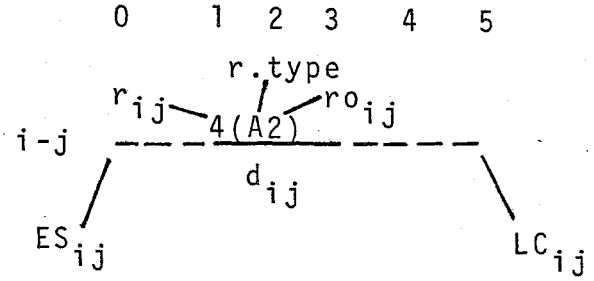
In Figure 3d, we remove the cut from period 3 because there is only job 3-4 left in this period. Resource summations are continued until period 4 where we have a new network cut for jobs 1-4 and 3-4 with total resource requirement 6.

All possible combinations of jobs for example project




Key to figures:




For simplicity, critical activities are not shown in the figures. They can be considered to have zero resource requirements for this project. They do not take a part in the scheduling combinations because of their zero slacks.

FIGURE 4 Example Project

Job 2-3 is moved one period forward in Figure 3 e and as can be seen the network cut at period 4 no more exists. All the summations of resources are done and the maximum 4 replaces the old maximum 5. The schedule in this figure is also saved as the bestschedule found so far. It must be noted that the job 3-4 is also moved one period forward because it is a successor job for the job 2-3 which was also moved in this enumeration.

At Figure 3f, we continue until period 3 in which we exceed our new maximum of 4 resources by 1. This new cut still exists in Figure 9, therefore the schedule in Figure 9 is also a complete elimination.

Finally the new schedule in Figure 3h, is also partially eliminated because of another cut at period 4.

Briefly there are only 2 schedules that are computed as a whole, 4 schedules that are completed partially and 2 schedules are entirely eliminated without ony resource summations.

The total number of resource summations were 5 for each schedule adding to 40 summations for the entire 8 successive schedules but as can be counted from the figures, only about half of them (24) are carried out. The schedules are exhaustively enumerated and the optimum solution (the schedule in Figure 3e with its maximum resource requirement 4) is found for this problem.

At the scheduling stage, each additional activity increase the number of possible scheduling combinations by an amount proportional to its total slack. For example if the project were only made of activity 1-4, the number of scheduling combinations would be equal to its total slack plus 1, which is 4. Adding the activity 2-3 to this project, we have 4 extra combinations. This number is found by multiplying the total slack of job 2-3 (which is 1) with the number of possible combinations

already in hand (which was 4). The effect of job 3-4 is nothing since by precedence constraint, once its predecessor 2-3 is moved, job 3-4 also moves and acts like a continuation of 2-3.

Going through the same discussion, it is possible to draw a curve that shows the upper limit of all possible scheduling combinations for each addition of jobs to the project. We must consider a project of 4 noncritical activities having 2 slack periods each.

| Job | Additional Schedules | | Total Scheduling Combinations |
|---|---|---|---|
| | | 1 | 1 |
| | Slack | | |
| 1 | 2 x 1 | 2 | 3 |
| 2 | 2 x 3 | 6 | 9 |
| 3 | 2 x 9 | 18 | 27 |
| 4 | 2 x27 | 54 | 81 |

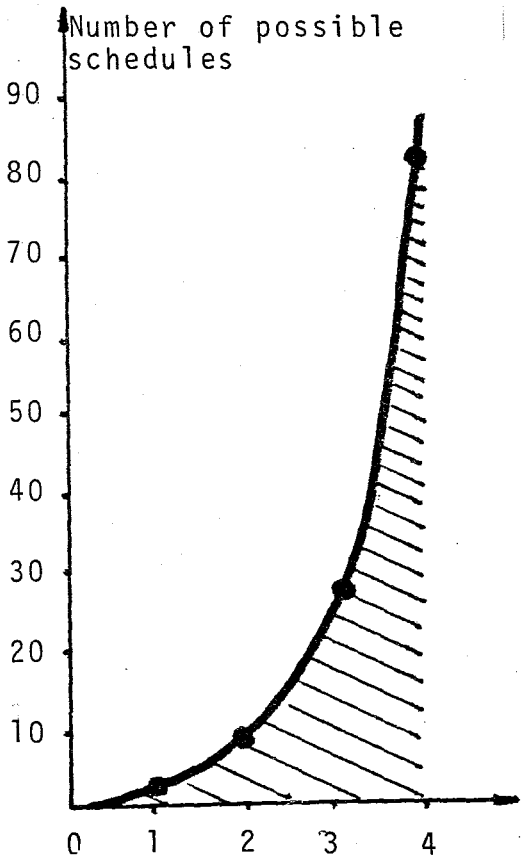



FIGURE 5
Computation steps

The dramatic increase of the slope of this curve is an inevitable result of attacking the problem with an optimal technique. The same point is also mentioned by A.M. Geoffrion (8).

> "The ultimate practical usefulness of any integer programming algorithm depends on the crucial guestion: How fast does solution time increase with problem size?.. If solution times tend to increase exponentially with the number of variables (that its, if the solution time is proportional to some constant greater than unity raised to the nth power) then there is little hope of ever being able to solve really large problems directly".

But it must be stated that the real line of the number of possible solutions lies in the shaded region under the curve. This is because of the eliminations by network cuts and precedence constraints as shown in the example project. Therefore these help the procedure to expand its strength at solving larger networks by decreasing the number of explicit enumerations.

A possibly more important consideration than size in the generalization of these findings has to do with problem characteristics such as network structure and/or resource requirements. If such characteristics are on behalf of implicit enumeration, than the procedure can handle larger problems. But is not possible to state how the problem characteristics and resource requirements would affect the number of steps to optimum solution for the time. It seems to be a detailed guestion, subject to an independent study.

B. 2.) ENUMERATION FOR OVERTIME (SECOND STAGE)

At next stage the minimum resource schedule is used as an input to the problem of overtime scheduling. In other words,

it is checked to see if the given schedule can be carried out more economically with less resources using overtime. The objective is to find the best overtime schedule and amount of resources to minimize the total regular, overtime pay and hiring-firing costs. But it must be kept in mind that the optimum schedule of the jobs that gave the minimum resource usage in the first stage of this enumerative process is not subject to change in the second stage. The finish times found before are now fixed and jobs are not moved anymore.

At this stage, another exhaustive enumeration routine incorporating overtime working for all the possible combination of jobs of the network is applied. For each single combination, the job or jobs that are carried out with overtime are given adjusted resource usages according to overtime working period and resulting total cost is calculated.

It is assumed that if overtime is to be used, the complete job would be performed by overtime work. In other words, performing part of a job by regular and the rest by overtime work is not allowed.

The cost figure found from the first enumeration is used as an upper bound for the subsequent enumeration and whenever a new combination of overtime and regular working schedule gives a smaller total cost than the one already in hand, this new schedule and its cost outcome replace their old companions, receive the names "best available overtime working schedule" and "minumum cost", respectively. After systematic testing and comparison of all the possible overtime working combinations, the schedule and true minimum cost are found.

Throughout the analysis, the productivity of a specific resource is thought to be constant. In other words, given a certain man-hours, required to complete a job when overtime is assigned in the form of extra hours we need less men to finish

the job. In a time unit consisting of 8 working hrs:

$$8.r_{ij} = (8 + \text{overtime duration})\ ro_{ij}$$

$ro_{ij}$ is the resource requirement for the job with overtime. As can be seen from this formula,

$$ro_{ij} = \left(\frac{8}{8 + \text{overtime duration}}\right) r_{ij}$$

$ro_{ij}$ is always less or equal to $r_{ij}$ because multiplier which is less than unity (inside of the paranthesis).

All the possible combinations of jobs are performed with overtime and the resulting resource requirement totals are changed to cost figures by using regular pay, overtime, hiring and firing costs. The resulting total cost and overtime scheduled for this cost is saved if the combination results in a reduced total cost than the one already in hand.

At the end of this procedure the last saved cost is the minimum cost and the corresponding overtime schedule is obtained.

For a project with only 3 activities, the number of all the overtime combinations (Z) are:

$$Z = C_3^1 + C_3^2 + C_3^3$$

$$= \frac{3!}{1!\ (3-1)!} + \frac{3!}{2!\ (3-2)!} + \frac{3!}{3!\ (3-3)!}$$

$$= \frac{6}{2} + \frac{6}{2} + \frac{6}{6}$$

$$= 7$$

As a generalization, for a project with n different jobs, the total number of all overtime combinations are:

$$Z = C_n^1 + C_n^2 + C_n^3 + \ldots + C_n^{n-1} + C_n^n$$

stated in another way,

$$Z = 2^n - 1$$

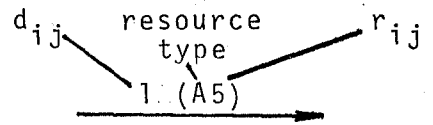
The last formula shows more clearly that the argument of A.M. Geoffrion holds for this stage too, and justifies the bounding method used.

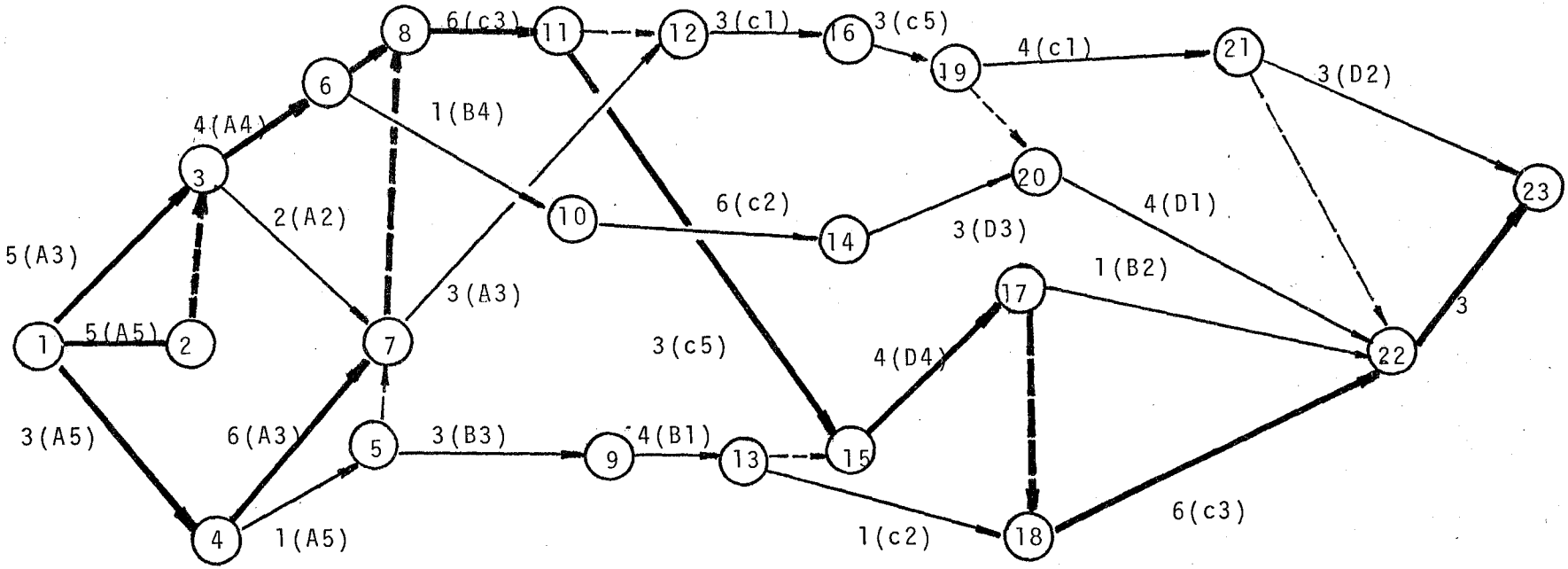
The technique used for this stage is a simplified type of the one employed in stage one. The procedure jumps to the next combination whenever the considered schodule's cost exceed the one already in hand.

# CHAPTER V

## A) TEST PROBLEMS

PROJECT 1




critical activity
activity
dummy activity

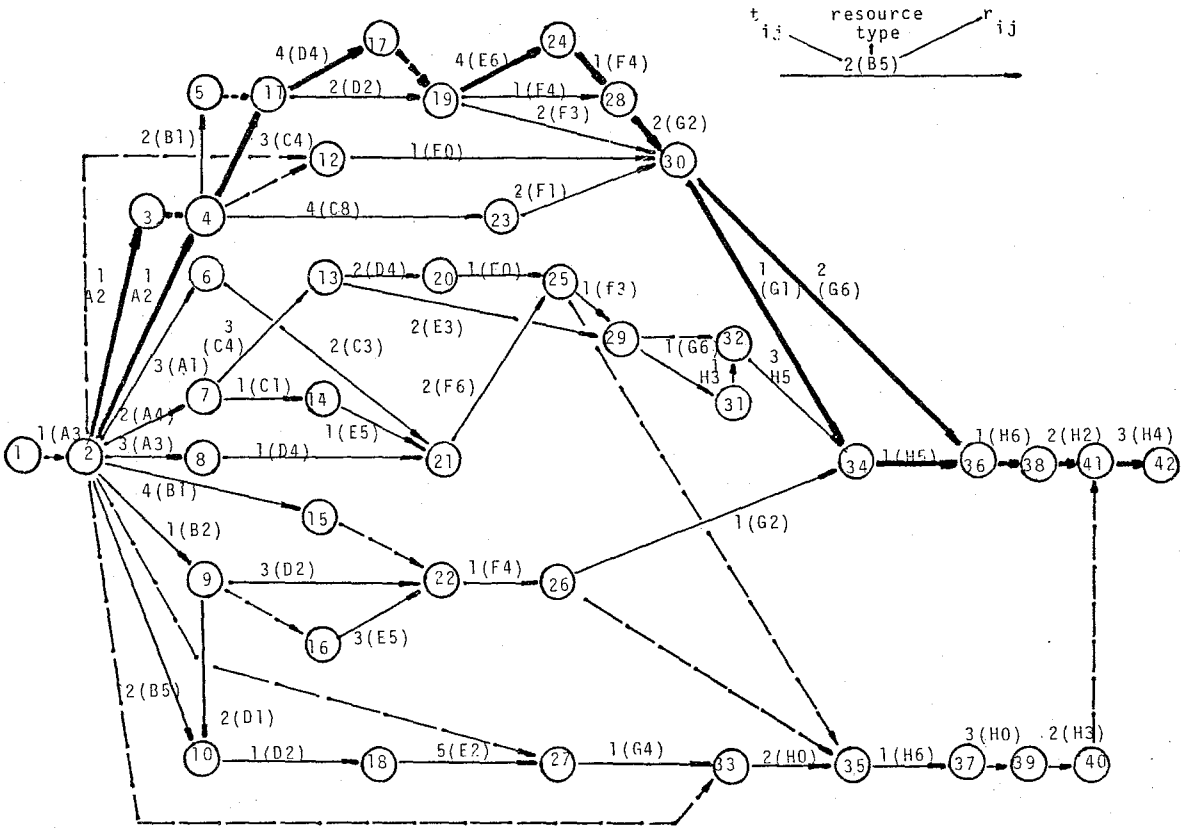
Number of nodes :23
Number of activities:34
Number of resources : 4

| RESOURCE TYPE | UNIT COST | OVERTIME COST | HIRING COST | FIRING COST | OVERTIME DURATION |
|---|---|---|---|---|---|
| A | 1800 | 600 | 100 | 100 | 2 |
| B | 900 | 700 | 200 | 200 | 1 |
| C | 2450 | 500 | 900 | 200 | 3 |
| D | 1500 | 400 | 500 | 300 | 2 |

Network is obtained from (16)

FIGURE 6

PROJECT 2

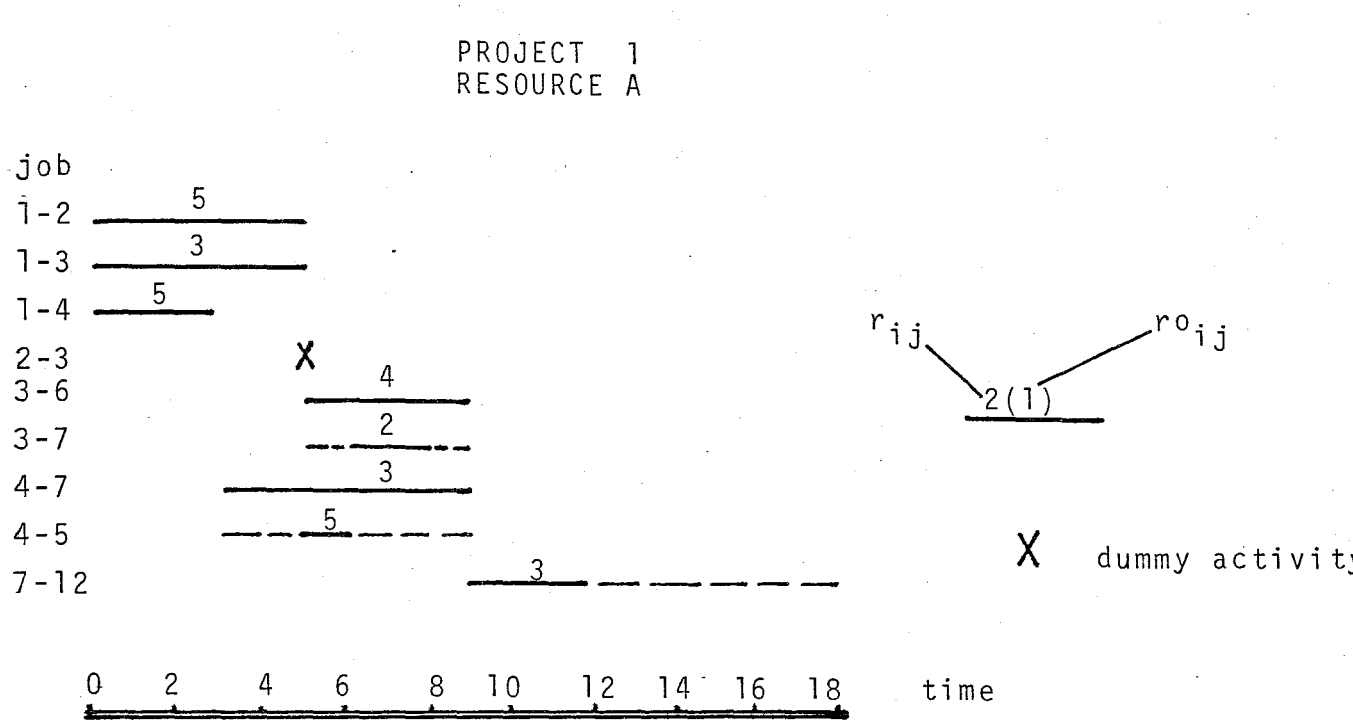


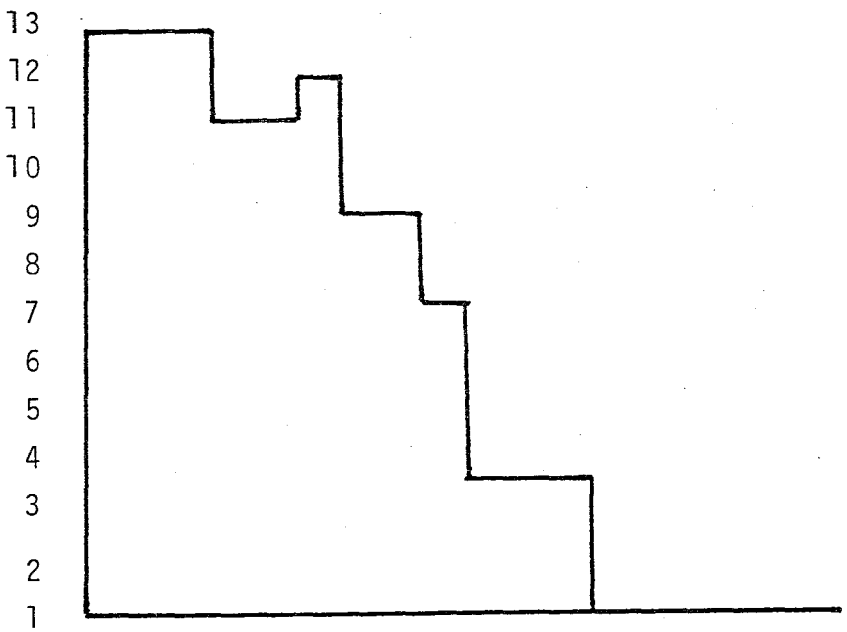

umber of nodes : 42
umber of activities : 65
otal number of resources : 8

critical activitiy :
activitiy :
dummy activity :

| RESOURCE TYPE | UNIT COST | OVERTIME COST | HIRING COST | FIRING COST | OVERTI DURATI |
|---|---|---|---|---|---|
| A | 1500 | 850 | 1000 | 1000 | 1 |
| B | 180 | 60 | 1500 | 900 | 4 |
| C | 2000 | 30 | 2000 | 500 | 5 |
| D | 500 | 40 | 900 | 1500 | 6 |
| E | 1200 | 50 | 1800 | 800 | 3 |
| F | 300 | 60 | 800 | 1000 | 2 |
| G | 400 | 90 | 1000 | 900 | 5 |
| H | 350 | 70 | 9000 | 7000 | 8 |

FIGURE 7

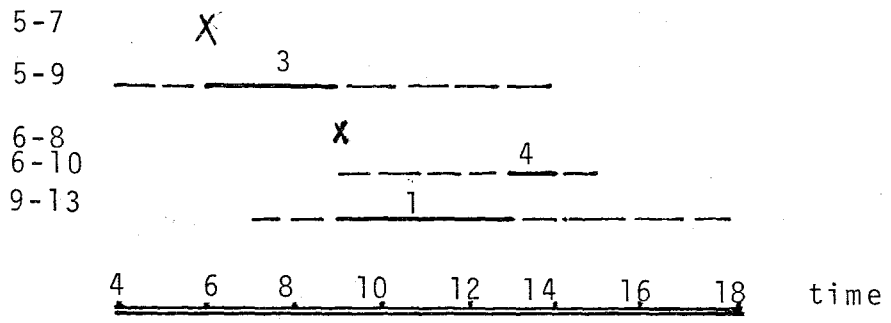


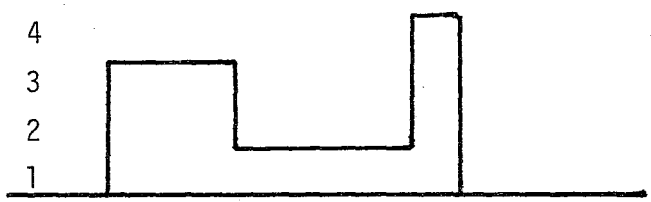




| | NORMAL | OVERTIME |
|---|---|---|
| MAX. RESOURCE | 13 | - |
| COST | 195 400 | - |

FIGURE 8

PROJECT 1
RESOURCE B



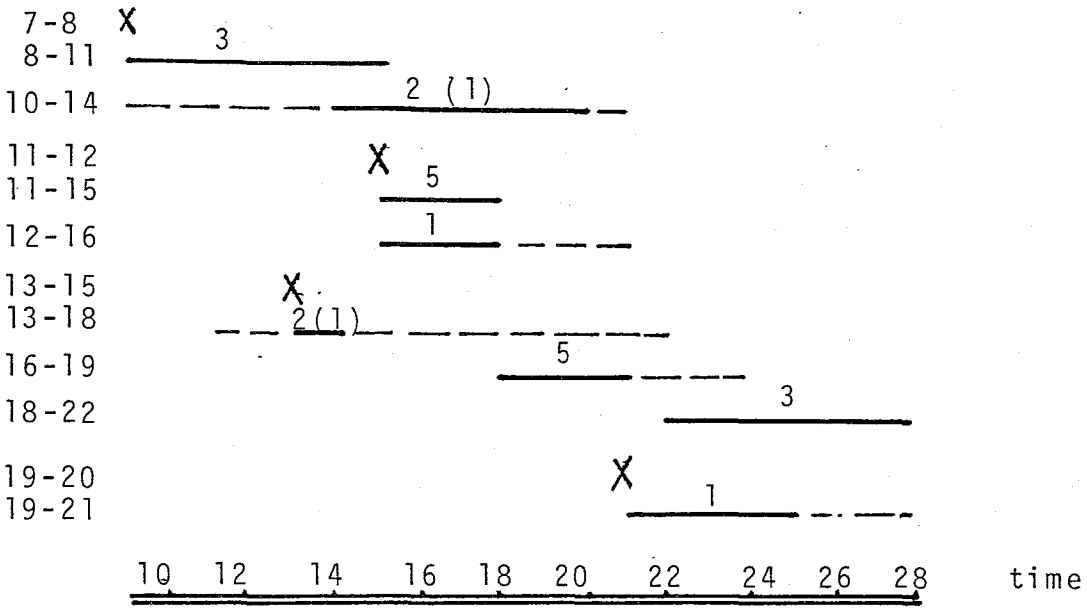



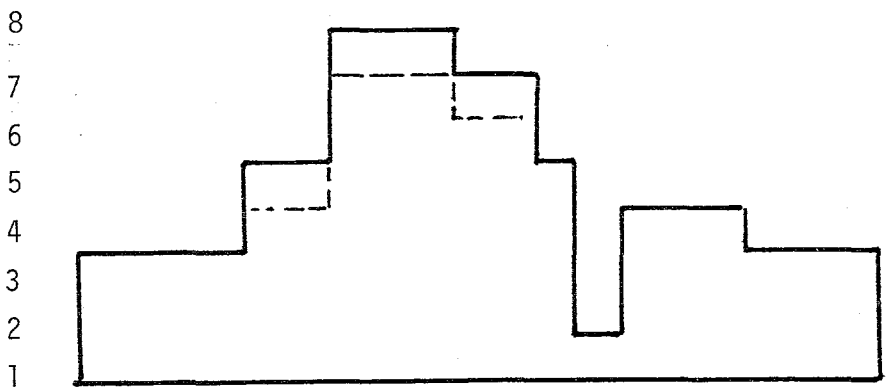

| | NORMAL | OVERTIME |
|---|---|---|
| MAX. RESOURCE | 4 | - |
| COST | 17 700 | - |

FIGURE 9

PROJECT 1

RESOURCE C



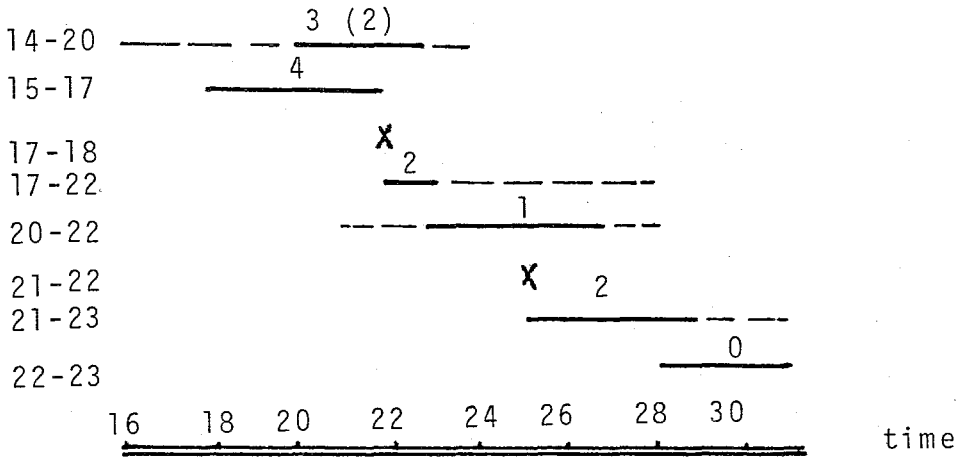



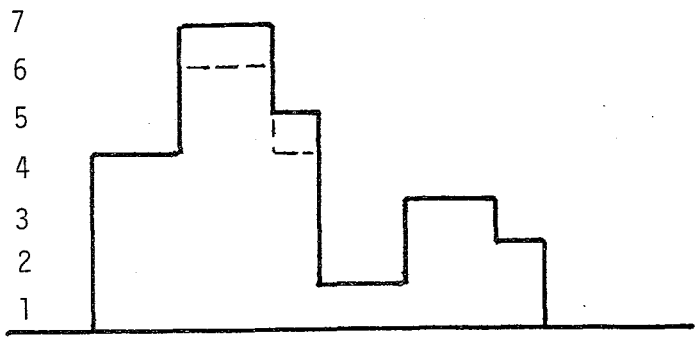

| | NORMAL | OVERTIME |
|---|---|---|
| MAX RESOURCE | 8 | 7 |
| COST | 225 250 | 207 600 |

FIGURE 10

PROJECT 1

RESOURCE D

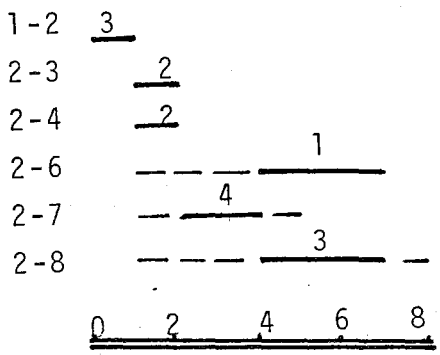


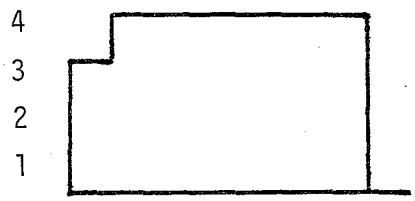




| | NORMAL | OVERTIME |
|---|---|---|
| MAX. RESOURCE | 7 | 6 |
| COST. | 62 700 | 62 200 |

FIGURE 11

PROJECT 2
RESOURCE A

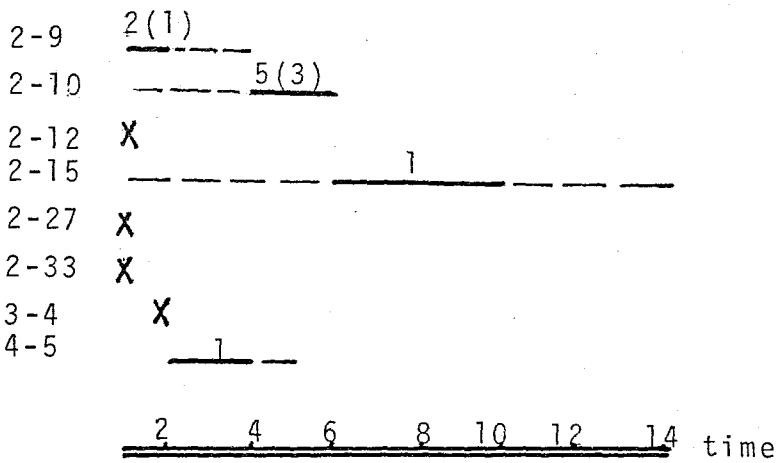


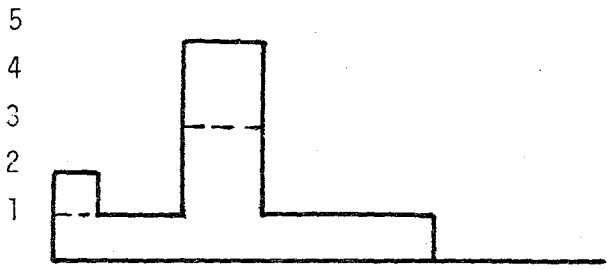




| | NORMAL | OVERTIME |
|---|---|---|
| MAX. RESOURCE | 4 | - |
| COST | 48 500 | - |

FIGUR E 12

PROJECT 2
RESOURCE B

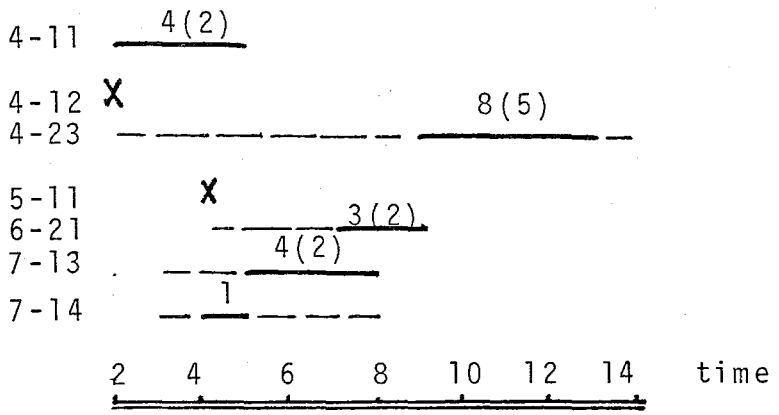


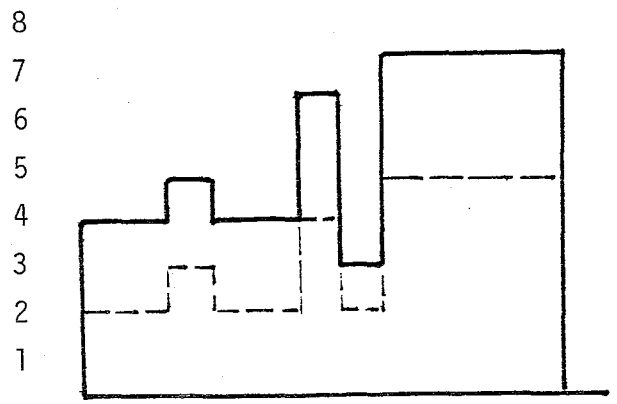




| | NORMAL | OVERTIME |
|---|---|---|
| MAX. RESOURCE | 5 | 3 |
| COST | 17 640 | 11 220 |

FIGURE 13

PROJECT 2
RESOURCE C

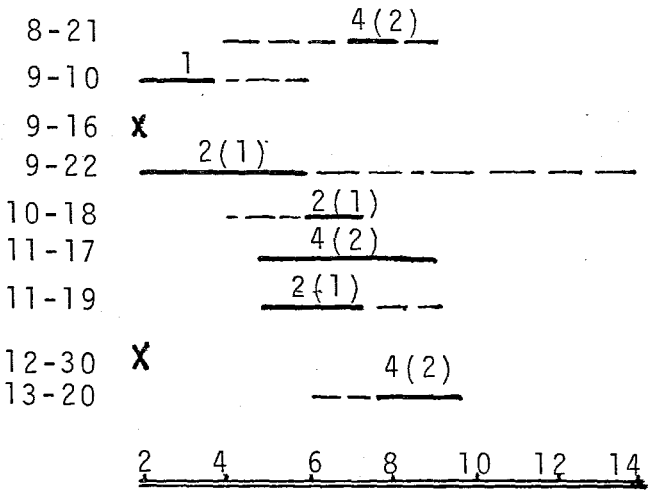


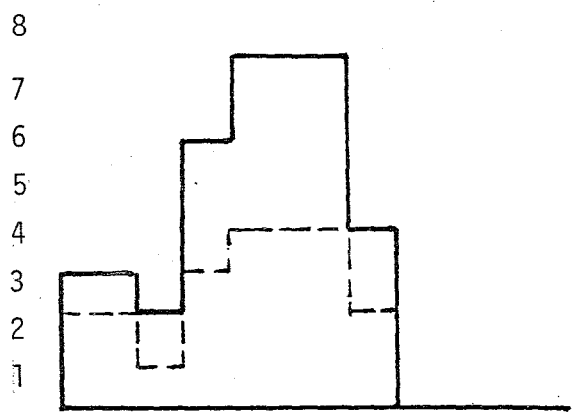




| | NORMAL | OVERTIME |
|---|---|---|
| MAX. RESOURCE | 8 | 5 |
| COST | 168 500 | 99 400 |

FIGURE 14

PROJECT 2
RESOURCE D

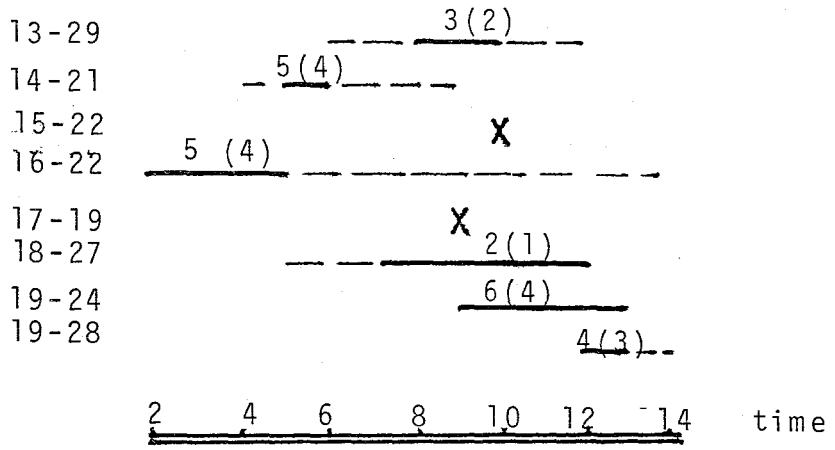


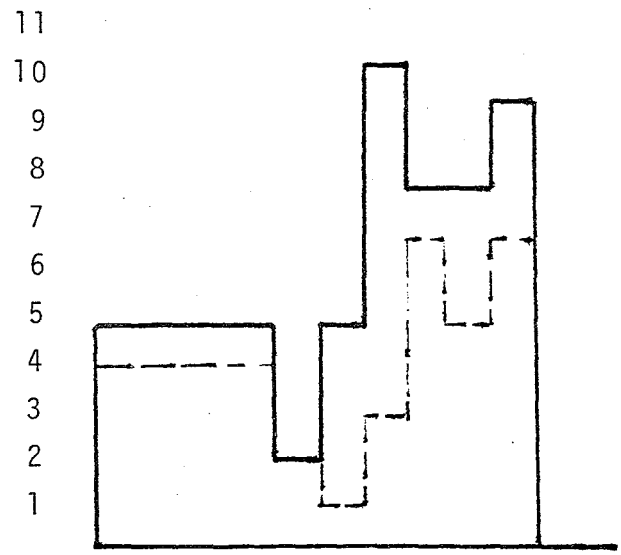




| | NORMAL | OVERTIME |
|---|---|---|
| MAX.RESOURCE | 8 | 4 |
| COST | 42 600 | 47 800 |

FIGURE 15

PROJECT 2
RESOURCE E

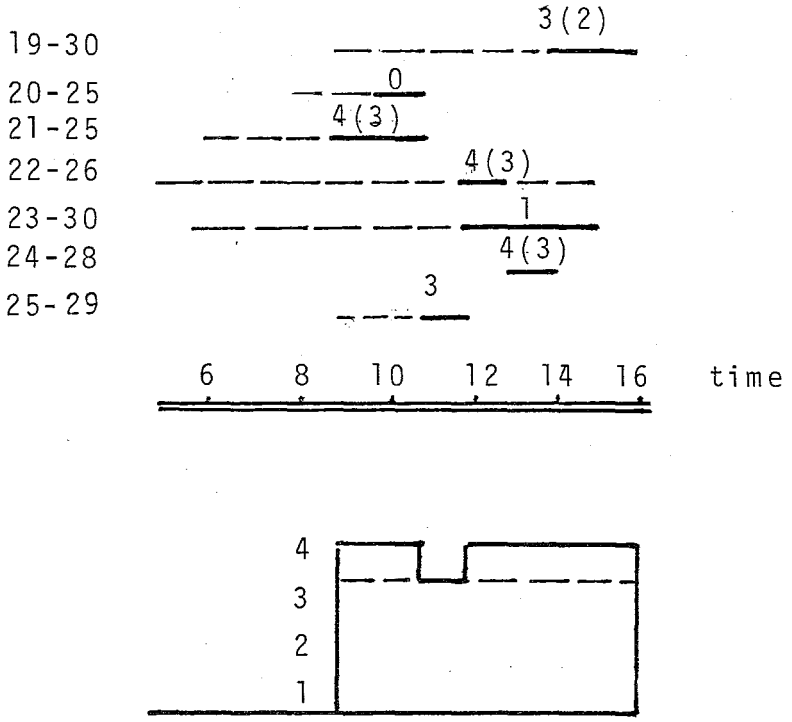



| | NORMAL | OVERTIME |
|---|---|---|
| MAX. RESOURCE | 11 | 7 |
| COST | 118 400 | 100 200 |

FIGURE 16

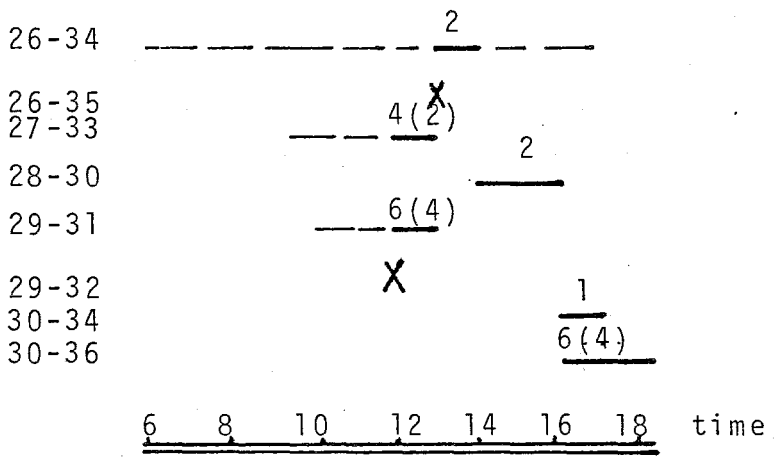
PROJECT 2
RESOURE F

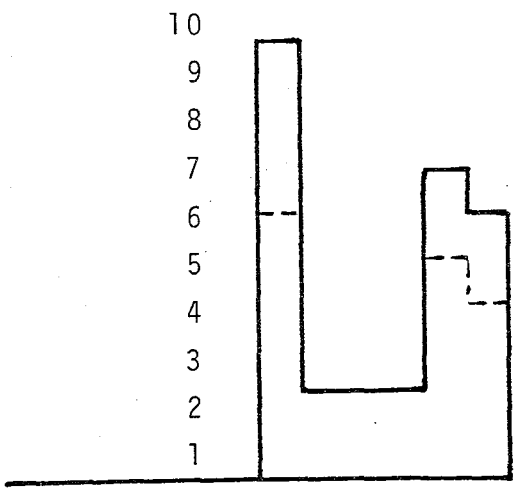



| | | NORMAL | OVERTIME |
|---|---|---|---|
| MAX. | RESOURCE | 4 | 3 |
| COST | | 17 100 | 13 620 |

FIGURE 17

PROJECT 2
RESOURCE G

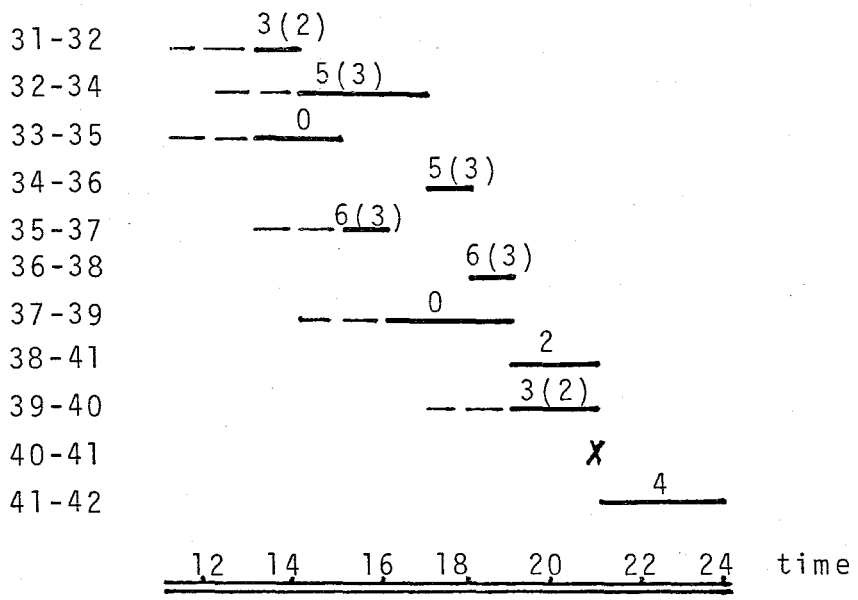


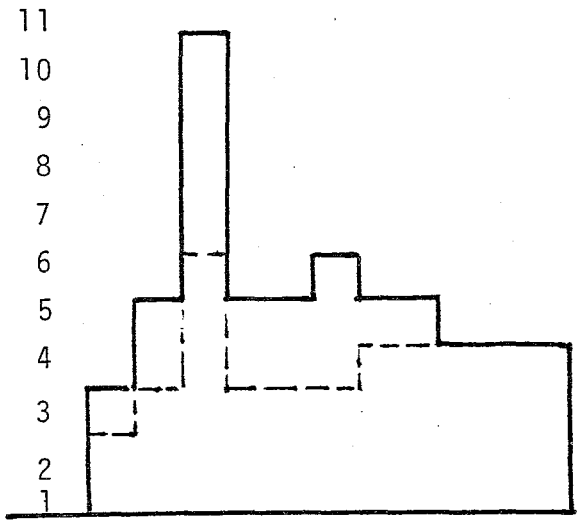




| | NORMAL | OVERTIME |
|---|---|---|
| MAX. RESOURCE | 10 | 6 |
| COST | 40 100 | 31 800 |

FIGURE 18

PROJECT 2
RESOURCE H




| | NORMAL | OVERTIME |
|---|---|---|
| MAX. RESOURCE | 11 | 6 |
| COST | 211 950 | 139 440 |

FIGURE 19

## B) THE COMPUTER CODE

```
      PROGRAM AS6(D6,O6,INPUT,OUTPUT,TAPE5=D6,TAPE6=O6)
      COMMON IT(99,99),ITE(99),ITEG(99)
      COMMON ITG(99),ITGG(99),IR(99,99)
      COMMON IF(99),IPC,IBEST(99),IRTN
      COMMON JN(3000),NM,M,IS,ISAA,ISBB
      COMMON KK,JS,ID,KB(198),KS(198),IY(99)
      COMMON IO(10),IU(10),IRT(99,99),ITF(99)
      COMMON NI,MITNM,GC,JC,COS,IYC(99)
      COMMON CF(10),CO(10),CU(10),CH(10),MM
      COMMON IHO(10)
      WRITE (6,441)
  441 FORMAT (" NAME OF PROJECT:")
      WRITE (6,541)
  541 FORMAT (" ----------------"//)
      WRITE (6,543)
  543 FORMAT (" PROJECT CHARACTERISTICS:")
      WRITE (6,544)
  544 FORMAT (" ------------------------")
      READ (5,443) N,NM,MIRTN
  443 FORMAT (3I3)
      NZ1=N
      WRITE (6,442) N
  442 FORMAT (" NUMBER OF NODES:",I3,)
      WRITE (6,542) NM
  542 FORMAT (" NUMBER OF ACTIVITIES:",I3)
      WRITE (6,642) MIRTN
  642 FORMAT (" TOTAL NUMBER OF RESOURCES:",I3)
      DO 100 I=1,N
      DO 200 J=1,N
      IT(I,J)=-1
  200 CONTINUE
```

```
  100 CONTINUE
      DO 300 I=1,NN
      READ (5,9200) IBNN,ISNN,ITT,JN(I),IF(I)
      IT(IBNN,ISNN)=ITT
      IR(IBNN,ISNN)=JN(I)
      IRT(IBNN,ISNN)=IF(I)
  300 CONTINUE
 9200 FORMAT (5I7)
      DO 348 I=1,MIRTN
  348 READ (5,888) CO(I),CU(I),CF(I),CH(I),IHO(I)
  888 FORMAT (4E10.2,I10)
      READ (5,443) IE,NG
      IF (IE.EQ.0) GO TO 355
      DO 350 J=1,IE
      READ (5,443) I,ID
      ITE(I)=ID
  350 CONTINUE
  355 IF (NG.EQ.0) GO TO 365
      DO 360 J=1,NG
      READ (5,443) I,ID
      ITG(I)=ID
  360 CONTINUE
  365 WRITE (6,872) IE+1
  872 FORMAT (" NUMBER OF NODES WITHOUT PREDECESSORS:",I3)
      WRITE (6,874) NG+1
  874 FORMAT (" NUMBER OF NODES WITHOUT SUCCESSORS:",I3)
      ITE(1)=0
      DO 400 I=2,N
      II=I-1
      M=0
      IS=0
      DO 500 K=1,II
      KK=K
      J=I-K
      IF (IT(J,I).NE.-1) GO TO 10
      IS=IS+1
      GO TO 500
   10 ITEG(J)=IT(J,I)+ITE(J)
      IF (M.LT.ITEG(J))GO TO 30
      GO TO 500
   30 M=ITEG(J)
  500 CONTINUE
      IF (IS.NE.KK) GO TO 35
      GO TO 400
   35 ITE(I)=M
  400 CONTINUE
      ITG(N)=ITE(N)
      N1=N-1
      DO 600 I=1,N1
      KNI1=N-I+1
      IS=0
      M=ITG(N)
      NI=N-I
      DO 700 K=KNI1,N
      IF(IT(NI,K).NE.-1) GO TO 40
      IS=IS+1
      GO TO 700
   40 ITGG(K)=ITG(K)-IT(NI,K)
      IF (M.GT.ITGG(K)) GO TO 60
      GO TO 700
   60 M=ITGG(K)
  700 CONTINUE
      IF (IS.NE.I) GO TO 65
      GO TO 600
   65 ITG(NI)=M
  600 CONTINUE
```

```
      DO 800 I=1,N
      ITEG(I)=ITE(I)
      ITGG(I)=ITG(I)
  800 CONTINUE
      IPC=ITG(N)
      IRTN=1
      IO(IRTN)=1
      K=1
   33 DO 900 I=1,N1
      II=I+1
      DO 950 J=II,N
      IF (IT(I,J).EQ.-1) GO TO 950
      IF (IRT(I,J).EQ.IRTN) GO TO 32
      GO TO 950
   32 ITE(K)=ITEG(I)
      ITG(K)=ITGG(J)
      IF(K)=IT(I,J)
      IY(K)=IRT(I,J)
      ITF(K)=IR(I,J)
      KB(K)=I
      KS(K)=J
      K=K+1
  950 CONTINUE
  900 CONTINUE
      IU(IRTN)=K-1
      IF (IRTN.EQ.MIRTN) GO TO 13
      IRTN=IRTN+1
      IO(IRTN)=K
      GO TO 33


ENUMERATION

   13 DO 1300 I=1,NM
      IR(I)=ITF(I)
      ITF(I)=0
      IRT(I)=IY(I)
      IT(I)=IF(I)
      IF(I)=ITE(I)+IT(I)
 1300 IBEST(I)=IF(I)
      IRTN=1
   15 IS=0
      IAA=IO(IRTN)
      IBB=IU(IRTN)
      IDAA=IAA
      IDBB=IBB
      ISAA=IAA
      ISBB=IBB
      IIAA=IAA
      IIBB=IBB
      IAA1=IAA
      IAA2=IAA
      IBB1=IBB
      IBB2=IBB
      DO 1313 I=IAA,IBB
      IF(I)=IST(I)
 1313 IBEST(I)=IF(I)
      CALL RESORS
      MM=M
      KA=1


      JS=0
      IS=1
      IAT=0
      NH=0
```

```
      DO 1600 JQ=IDAA,IDBB
      IF(IT(JQ).EQ.0) GO TO 1600
      IF (IF(JQ).EQ.ITG(JQ)) GO TO 1600
      J=JQ
  450 IL=1
  460 IF(J)=IF(J)+1
      JN(KA)=J
      IF (KA.EQ.30000) GO TO 8820
      I11=JN(1)
      KA=KA+1
      IF (JS.EQ.0) GO TO 480
      KI=1
      DO 1550 L=IAA,IBB
      IIN=IF(L)-IT(L)
      IF (IIN.LT.ID.AND.ID.LE.IF(L)) GO TO 490
      GO TO 1550
  490 ITEG(KI)=L
      KI=KI+1
 1550 CONTINUE
      KK1=KK-1
      DO 1555 LF=1,KK1
      IF (ITGG(LF).EQ.ITEG(LF)) GO TO 1555
      GO TO 480
 1555 CONTINUE
      IAT=IAT+1
      GO TO 410
  480 CALL RESCRS
      NH=NH+1
      IF (JS.EQ.1)GO TO 410
      IF (MM.LE.M) GO TO 410
      MM=M
      DO 1700 IK=IAA,IBB
 1700 IBEST(IK)=IF(IK)
  410 J=JN(IL)
      IF(J.EQ.JQ) GO TO 420
      DO 2000 JK=I11,IBB
      IF (JK.EQ.J) GO TO 430
      IF(JK)=IST(JK)
 2000 CONTINUE
      PRINT*,2000
      STOP
  420 IF(IF(JQ).EQ.ITG(JQ))GO TO 440
      DO 1800 JK=I11,IBB1
      IF (JK.EQ.J) GO TO 450
      IF(JK)=IST(JK)
 1800 CONTINUE
      PRINT*,1800
      STOP
  440 DO 1900 JK=I11,IBB2
      IF(JK)=IST(JK)
      IF (JK.EQ.J) GO TO 1600
 1900 CONTINUE
      PRINT*,1900
      STOP
  430 IL=IL+1
      GO TO 460
 1600 CONTINUE
      KA1=KA-1



      IAB1=IBB-IAA+1
      DO 2991 I=1,IAB1
      IYC(I)=0
 2991 IY(I)=0
      COS=(99999E99)**2
```

```
      NI=0
      CALL COST
      IF (MITNM.GT.MM) MITNM=MM
      MC=MITNM
      COS=GC
      COSB=COS
      COS1=COS
      MC1=MC
      DO 3000 I=1,IA91
      NI=I
      N=1
      DO 3100 J=1,I
 3100 IY(I-J+1)=IAA+J-1
 3300 CALL COST
      IF (JC.EQ.0) GO TO 3600
      NU=I
      IF (MITNM.GT.MM) MITNM=MM
      MC=MITNM
      COS=GC
      IF (COS.EQ.COSB) GO TO 3600
      DO 3825 L=1,I
 3825 IYC(L)=IY(L)
 3600 IF (IY(N).EQ.IBB-N+1) GO TO 3200
      IY(N)=IY(N)+1
 3400 IF (N.EQ.1) GO TO 3300
      IYD=IY(N)
      N=N-1
      IY(N)=IYD+1
      GO TO 3400
 3200 IF (N.EQ.I) GO TO 3000
      N=N+1
      GO TO 3600
 3000 CONTINUE
      WRITE (6,643)IRT(IAA)
  643 FORMAT (//" RESOURCE TYPE:",I2,"   RESOURCE:")
      WRITE (6,644)
  644 FORMAT (" -------- ----      --------"/)
      WRITE (6,922) KA1
  922 FORMAT (/"    NUMBER OF TRIALS:",I9)
      WRITE (6,449) NH,IAT
  449 FORMAT ("  NORMAL: ",I9,"  JUMPS:  ",I9)
      WRITE (6,524) CU(IRTN)
  524 FORMAT (" UNIT COST:",F8.2)
      WRITE (6,526) CO(IRTN)
  526 FORMAT (" OVERTIME COST:",F8.2)
      WRITE (6,528) CH(IRTN)
  528 FORMAT (" HIRING COST:",F8.2)
      WRITE (6,529) CF(IRTN)
  529 FORMAT (" FIRING COST:",F8.2)
      WRITE (6,527) IHO(IRTN)
  527 FORMAT (" MAXIMUM OVERTIME",I2)
23456789.123456789.123456789.123456789.123456789.123456789.123456789.12
      WRITE (6,645)
  645 FORMAT (" G    I    J    TI    RE    ES    LS    EC    LC    FT    TS    FS
     2   CR    OT")
      WRITE (6,646)
  646 FORMAT (" -    -    -    --    --    --    --    --    --    --    --    --
     8   --    --")
      DO 3500 I=IAA,IBB
      LS=ITG(I)-IT(I)
      IEC=ITE(I)+IT(I)
      ITS=ITG(I)-ITE(I)-IT(I)
      KD=KS(I)
      DO 3505 K=1,NM
      IF (KE(K).NE.KD) GO TO 3505
      KD=K
```

```
      GO TO 3506
 3505 CONTINUE
 3506 IF (KS(I).EQ.NZ1) GO TO 3507
      IFS=ITE(KD)-ITE(I)-IT(I)
      GO TO 3508
 3507 IFS=ITG(I)-ITE(I)-IT(I)
 3508 ICDD=0
      IF (ITS.EQ.0) ICDD=1
      IF (CCS.EQ.COSB) GO TO 3515
      DO 3510 J=1,NU
      IF (IYC(J).NE.I) GO TO 3510
      IOTD=1
      GO TO 19
 3510 CONTINUE
 3515 IOTD=0
   19 WRITE (6,647) I,KB(I),KS(I),IT(I),IR(I),ITE(I),LS,IEC,
     4ITG(I),IBEST(I),ITS,IFS,ICDD,IOTD
  647 FORMAT (I3,I4,I4,11I5)
 3500 CONTINUE
      WRITE (6,927) (IYC(IM),IM=1,20)
  927 FORMAT (20I3)
      GO TO 8888
      WRITE (6,924)
  924 FORMAT (///" HISTOGRAM WITHOUT OVERTIME:"/)
      WRITE (6,941)
  941 FORMAT (" DAYS RES")
      ITC=0
      ISRL=0
      MC1=0
      DO 211 I=1,IPC
      ISR=0
      DO 213 J=IAA,IBB
      IIN=IBEST(J)-IT(J)
      IF (IIN.LT.I.AND.I.LE.IBEST(J)) GO TO 215
      GO TO 213
  215 ISR=ISR+IR(J)
  213 CONTINUE
      ITC=ITC+ISR*CU(IRTN)
      IF (ISR.NE.0) GO TO 216
      GO TO 210
  216 DO 214 L=1,ISR
  214 ITF(L)=0
      WRITE (6,2150) (I,ISR,(ITF(KF),KF=1,ISR))
  210 IF(ISR.LT.ISRL)ITC=ITC+(ISRL-ISR)*CF(IRTN)
      IF(ISR.GT.ISRL)ITC=ITC+(ISR-ISRL)*CH(IRTN)
      ISRL=ISR
      IF (ISR.GT.MC1) MC1=ISR
  211 CONTINUE
 2150 FORMAT (2I4,"   ",60I1)
 2152 FORMAT(2I4)

      WRITE (6,3909) MC1,ITC
 3909 FORMAT (/" MAX. RESOURCE:",I3,"   COST:  ",I10//)
      WRITE (6,926)
  926 FORMAT (" HISTOGRAM WITH OVERTIME:"/)
      WRITE (6,928)
  928 FORMAT (" DAYS RES")
      ITC=0
      ISRL=0
      MS=0
      DO 4100 I=1,IPC
      ISR0=0
      ISR=0
      DO 4200 J=IAA,IBB
      IIN=IBEST(J)-IT(J)
      IF (IIN.LT.I.AND.I.LE.IBEST(J)) GO TO 4110
```

```
      GO TO 4200
4110  IF (CCS.EQ.COSB) GO TO 4305
4110  DO 4300 K=1,NU
      IF (IYC(K).NE.J) GO TC 4300
      RO=8.*IR(J)/(8.+IHO(IRTN))
      IRO=RC
      IF (IRC+.5.LT.RO) IRC=IRC+1
      ISRO=ISRO+ISR
      GO TO 4200
4300  CONTINUE
4305  ISR=ISR+IR(J)
      PRINT*,ISR
4200  CONTINUE
      ITC=ITC+(ISR+ISRO)*CU(IRTN)+ISRO*CO(IRTN)*IHO(IRTN)
      ISR=ISR+ISRO
      IF (ISR.GT.MS) MS=ISR
      IF (ISR.NE.0) GO TO 4230
      GO TO 4120
4230  DO 4250 L=1,ISR
4250  ITF(L)=0
      WRITE (6,1004) (I,ISR,(ITF(KF),KF=1,ISR))
4120  IF(ISR.LT.ISRL)ITC=ITC+(ISRL-ISR)*CF(IRTN)
      IF(ISR.GT.ISRL)ITC=ITC+(ISR-ISRL)*CH(IRTN)
      ISRL=ISR
4100  CONTINUE
1004  FORMAT (2I4,"   ",60I1)
      WRITE (6,3900) MS,ITC
3900  FORMAT (/" MAX  RESOURCE:",I3,"  MIN. COST:  "I10)
8888  IF (IRTN.EQ.*IRTN) STOP
      IRTN=IRTN+1
      GO TO 15
8820  WRITE (6,8821)
8821  FORMAT (" PLEASE INCREASE THE SIZE OF ARRAY JN(I)")
      STOP
      END
```

RESORS:

```
      SUBROUTINE RESORS
      COMMON IT(99,99),ITE(99),ITEG(99)
      COMMON ITG(99),ITGG(99),IR(99,99)
      COMMON IF(99),IPC,IBEST(99),IRTN
      COMMON JN(30000),NM,M,IS,ISAA,ISBB
      COMMON KK,JS,ID,KB(198),KS(198),IY(99)
      COMMON IO(10),IU(10),IRT(99,99),ITF(99)
      COMMON NI,MITNM,GC,JC,COS,IYC(99)
      COMMON CF(10),CO(10),CU(10),CH(10),MM
      COMMON IHO(10)
      M=0
      DO 100 I=1,IPC
      IGR=0
      KK=1
      IAB=ISBB-ISAA+1
      DO 150 N=1,IAB
 150  ITGG(N)=0
      DO 200 J=ISAA,ISBB
      IIN=IF(J)-IT(J)
      IF(IIN.LT.I.AND.I.LE.IF(J)) GO TO 10
      GO TO 200
  10  IGR=IGR+IR(J)
      ITGG(KK)=J
      KK=KK+1
      IF(IS.EQ.0) GO TO 200
      IF (IGR.LE.MM) GO TO 200
      JS=1
```

```
      ID=I
      RETURN
  200 CONTINUE
      IF (IGR.LE.M) GO TO 100
      M=IGR
  100 CONTINUE
      JS=0
      RETURN
      END



      FUNCTION IST(IKZ)
      COMMON IT(99,99),ITE(99),ITEG(99)
      COMMON ITG(99),ITGG(99),IR(99,99)
      COMMON IF(99),IPC,IBEST(99),IRTN
      COMMON JN(30000),NM,M,IS,ISAA,ISBB
      COMMON KK,JS,ID,KB(198),KS(198),IY(99)
      COMMON IC(10),IU(10),IRT(99,99),ITF(99)
      COMMON NI,MITNM,GC,JC,COS,IYC(99)
      COMMON CF(10),CO(10),CU(10),CH(10),MM
      COMMON IHO(10)
      MAX=0
      DO 100 I=1,NM
      IF (KS(I).EQ.KB(IKZ)) GO TO 20
      GO TO 100
   20 IF (I.LT.ISAA.OR.ISBB.LT.I) GO TO 30
      GO TO 40
   30 IF (MAX.GE.IBEST(I)) GO TO 100
      MAX=IBEST(I)
      GO TO 100
   40 IF (MAX.GE.IF(I)) GO TO 100
      MAX=IF(I)
  100 CONTINUE
      IST=MAX+IT(IKZ)
      RETURN
      END


      SUBROUTINE COST
      COMMON IT(99,99),ITE(99),ITEG(99)
      COMMON ITG(99),ITGG(99),IR(99,99)
      COMMON IF(99),IPC,IBEST(99),IRTN
      COMMON JN(30000),NM,M,IS,ISAA,ISBB
      COMMON KK,JS,ID,KB(198),KS(198),IY(99)
      COMMON IC(10),IU(10),IRT(99,99),ITF(99)
      COMMON NI,MITNM,GC,JC,COS,IYC(99)
      COMMON CF(10),CO(10),CU(10),CH(10),MM
      COMMON IHO(10)
      MITNM=0
      ISRL=0
      GC=0
      DO 100 I=1,IPC
      ISR=0
      ISRO=0
      DO 200 J=ISAA,ISBB
      IIN=IBEST(J)-IT(J)
      IF (IIN.LT.I.AND.I.LE.IBEST(J)) GO TO 250
      GO TO 200
  250 IF (NI.EQ.0) GO TO  310
  250 DO 300 K=1,NI
      IF (IY(K).NE.J) GO TO 300
      RO=8.*IR(J)/(8.+IHO(IRTN))
      IRO=RO
      IF (IRO+0.5.LT.RO) IRO=IRO+1
```

```
      ISRO=ISRO+IRC
      GO TO 200
  300 CONTINUE
      ISR=ISR+IR(J)
  200 CONTINUE
      GC=GC+(ISR+ISRO)*CU(IRTN)+ISRO*CO(IRTN)*IHO(IRTN)
      ISR=ISR+ISRO
      IF(ISR.LT.ISRL)GC=GC+(ISRL-ISR)*CF(IRTN)
      IF(ISR.GT.ISRL)GC=GC+(ISR-ISRL)*CH(IRTN)
      IF (GC.GT.COS) GO TO 280
      IF (ISR.GT.MITNM) MITNM=ISR
      ISRL=ISR
      GO TO 100
  280 JC=0
      RETURN
  100 CONTINUE
      JC=1
      RETURN
      END
10.19.58.UCLP, AA, P04     ,       C.546KLNS.
```

## C) THE PROGRAM AND COMPUTATIONAL EXPERIENCE

The procedure developed in this study is used to solve two test problems on an CDC . Computer. The program which performed this task is written in FORTRAN language and measures about 500 lines induding all the subroutines. It is dimentioned to solve projects of up to 99 activities and 10 different resource types. It can be easily adapted to solve larger problems by changing the size of the related COMMON variables.

The program takes about 9 CPU seconds for compilation. The first project which has a project complexity of 1.43 ( =number of activitles/ number of nodes) was solved in about 63 CPU seconds and the second project with a complexity of 1.55 was solved in 20 CPU seconds. Allthough the second project has a greater complexity which is an execution time increasing variable, it is solved in about 1/3 of the first project's solution time. This shows quite clearly that favorable project characteristics ofstructure and resource requirements have a strong effect on the solution time of the procedure.

# CHAPTER VI

## A) CONCLUSIONS

The exhaustive enumeration method presented in this study minimizes the total resourcerequirements and the total cost of the project. Like the other optimal techniques it suffers also from the exponential increase of solution steps for each increase in project size. But these burdens are decreased by incorporating "network cuts" and "bounding".

Although the maximum size of a project that can be solved with this method in a reasonable computer time is not examined, it still can be said that projects with more than 70 activities would not lead to short solution times unless they have network structure and activity resource requirements favorable for the procedure explained here. Solution times of the test problems can be used to support this conclusion.

If the special problem structure does not permit the user to apply the method to the full extend, it may still be used to attain valuable information about the problem. The method proceeds from the most important resource type (the most costly one) to the least important type. It can be stopped anywhere during these steps and the results generated up to the point can be used successfully. Since the solved portion of the project coincides to the most costly part.

The procedure may be stopped even before the optimum solution to the first resource type group is obtained. This is made possible because of the unique nature of the bounding technique. It advances by solutions and each solution on the way

to the optimum solution is a feasible one. Furthermore, each successive solution is a better answer to the problem under consideration. Thus once the enumeration is stopped before the first optimum is reached, the obtained solution is a better one than the preliminary solution which we have without applying the method. These advantages make the proposed method also (at least) partially suitable for large networks that do not have favorable network structure and resource requirements.

B) RECOMMENDATIONS FOR FURTHER WORK

The scope of this study was to examine the presented method for applicability to certain projects and investigate the results. In order to achieve this there had to be certain simplifications made to obtain an "introductory" technique. Therefore the computer program developed for this purpose lacks many handy additions that would make it more flexible and the results more realistic. But once the method has proven to be functional, these improvements can be added to the program to improve it in some ways described below.

The program used for the study assumes that inputs related to network structure are correct. But in many real cases, the possibility of a false entry of network inputs are quite common. If this happens to be the case with this program, it will not detect and diagnose network errors such as loops or nonunique activities at all but will simply continue to run in a vain effort to perform its job. Therefore the inputs of this program must be double-checked to make sure that there is no network error. To change the program suitable to practical uses however, a routine must be added to its main body that would detect the network errors, print out the event numbers that are incorrect. This would greatly aid in the manual search for network or input errors.

CPM/Pert type of programs are used by many levels of

management. It is, therefore, important that the program speak the managers' language as nearly as possible. One of the efforts of this criterion is the use of actual calendar dates in the output phase of the program. Instead of starting at "time zero", for example, projects could be instructed to start on a specific date, such as 18/2/1986 and all output would be expressed in calendar dates. In large scale operations this convenience would save a great deal of clerical effort and avoid confusion over dates. The efficiency of the program would be greater if it gives both catendar dates and elapsed time.

The higher the level of management, greater is the demand to see charts rather than tabulated data. Since digital computers do not easily produce charts, a routine can be added to the program that would prepare bar charts of activities, "drawing" the bars on a time scale by repetition of some character ( *or 0). With this the reports become more readable and useful, and thus the expense of manual preparing and maintaining of timescaled graphics are avoided.

The cost components are considered to be fixed throughout the project's life. In an inflationary environment this assumption loses its strenght even for a short duration project. Either discrete or continious changes of the costs can be introduced to the program with a little extension, making the program more realistic.

Non-linear as well as discrete changes of hiring or firing costs resulting from the change of level of a specific resource type at a working period can also be added as a routine.

The constant productivity assumption may be altered if the prolonged overtime work is to be scheduled and another routine may handle the new stuation by adjusting productivity of the resources.

## REFERENCES:


1. Popescu Calin, "Update of CPM/Pert Glossary of Terms"., ASCE J. of Construction Div., V 104 (1978) pp 35-41.

2. Jenett, E., "Availability of CPM Programs", Project Management Quarterly, Vol 1, No 2 (1970) pp. 10-13

3. Wiest, J.D., "A Heuristic Model for Scheduling Large Projects With Limited Resources"., Man. Sci. vol. 13, No 6 (1967) pp. 359-378

4. Gorenstein, S.,"An Algorithm for Project Sequencing with Resource Constraints", Operations Research, Vol 20, No 4 (1972) pp.835-840.

5. Johnson T.J.R., "An Algorithm for the Resource constrained Project scheduling Problem", school of Management, MIT, August 1967.

6. Schrage, 1., "Solving Resource Constrained Network problems by Implicit Enumeration", Operations Research vol 18, No 2 (1970) pp.225-35.

7. Pritsker A, Watter L.J., Wolfe P.M., " Multiproject Scheduling with Limited Resources , A 0-1 programming Approach", Management Science, vol 16 (1969) pp. 93-108

8. Geoffrion, A. "An Improved Implicit Enumeration Approach for Integer Programming", Oper. Research., vol 17 (1969) pp. 437-454.

9. Patterson, J.H., Huber, W.,"A Horizon Varying 0-1 Approach to Project Scheduling", Man. Sci., Vol 20 (1974) pp.990-998.

10. Patterson, J.H., Talbot, F.B., "An Efficient Integer Programming Algorithm with Network Cuts for Solving Resource Constrained Problems"., Man. Sci., Vol 24. No 11 (1978) pp.1163-1174.

11. Huber, W.D., "Computational Experience with 0-1 Programs for Multi Project Scheduling", Unpublished MBA Proffessional paper, The Pennsylvania State University, August 1971.

12. Brand, J.D., Meyer, W.L., Schaffer L.R., "The Resource Scheduling Problem in Construction", Civil Eng. Studies Report no.5, Urbana, Illinois, Dep. of Civil Eng., W of Illinois, 1964.

13. Horowitz-Joseph, " Critical Path Scheduling-Management Control Through CPM and Pert", Reinhold Ronald Press Company, N.Y. 1967.

14. Moder, Joseph,j., Phillips, Cecil R. "Project Management with CPM and Pert", Reinhold Publishing Corporation, N.Y. 1964.

15. Çetmeli, Enver, "Yatırımların Planlanmasında Kritik Yörünge (CPM) ve Pert Metotları", Teknik Kitaplar Yayınevi, İstanbul-1982.

16. Davis, Edward W., Patterson, James H. "A Comparison of Heuristic and Optimum Solutions In Resource-Constrained Project Scheduling" Management Science, Vol: 21 No 8, April 1975, pp. 944-955.