AN OPTIMIZATION PROCEDURE FOR NETWORK PLANNING SYSTEMS

A Thesis

By

ORLANDO SIA MADRIGAL

Submitted to the Graduate College of the
Texas A&M University in
partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 1967

Major Subject: Industrial Engineering

AN OPTIMIZATION PROCEDURE FOR NETWORK PLANNING SYSTEMS

A Thesis

By

ORLANDO SIA MADRIGAL

Submitted to the Graduate College of the
Texas A&M University in
partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 1967

Major Subject: Industrial Engineering

AN OPTIMIZATION PROCEDURE FOR NETWORK PLANNING SYSTEMS

A Thesis

By

ORLANDO SIA MADRIGAL

Approved as to style and content by:

_____
(Chairman of Committee)

_____
(Head of Department)

_____
(Member)

_____
(Member)

May 1967

ACKNOWLEDGMENT

I wish to express my sincere gratitude to Dr. A. W. Wortham for allowing me to serve as a member of the PERT-CPM research group. He also formulated the problem embodied in this thesis, and I would like to give him full credit for deriving the algebraic analogue to the time compression problem presented in Chapter 6. My sincere appreciation also goes to Prof. Leonard Lamberson for his patience in working with me, and for his invaluable advice during my research. I would also like to thank Prof. J. P. CoVan, and Dr. R. J. Freund for serving as members of my committee.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

C H A P T E R I

INTRODUCTION

Planning and controlling of large projects is one of the most complex problems of modern business. This is true of manufacturing and construction, design and installation of new production lines, equipment installations, maintenance projects, research and development, engineering design, and long-range management projects. In most cases substantial savings or competitive advantage may be obtained by effective solution of such problems.

The Critical Path Method (CPM) is a technique which can be used to plan, schedule, and control a project consisting of a group of interrelated activities or jobs directed towards a common goal.

The use of the critical path method was first introduced by Walker and Kelly (8) of du Pont in 1958. Since then numerous articles and books have been published in this area. This thesis embodies a particular area of CPM with emphasis on cost optimization.

The field of cost optimization as applied to a scheduling network will not only apply to the goal of completing a specified project at minimum cost, but will also involve the possibility of reducing the total duration of a project. The algorithmic process for reducing the duration of a project, hereafter, referred to as time compression, will follow the path of minimum cost. The development of the algorithm is discussed in the succeeding chapters.

As one will note later, the area of time compression will only be of practical use if a computer is available for computation or if

the network is comparatively simple. A computer program has been written that accomplishes the time compression algorithm. Chapter 7 gives a description of this program, and the program listing is contained in Appendix II.

It is assumed here that the reader is familiar with CPM nomenclature and numerical calculations. For those without any CPM background, it will be well to review texts and publications in this area such as Moder and Phillips (5), Evarts (6), and Martino (4).

C H A P T E R I I

PROBLEM STATEMENT

Since its inception in 1958, the Critical Path Method has been widely used in almost all phases of industry. With many people involved in the use of CPM, those schooled in the area of Operations Research have developed and published articles related to the numerous variations and uses of this technique. Dozens of computer programs are available to the CPM analyst and of course revisions to any method are always in order.

The National Aeronautics and Space Administration has a technique known as PERT Time which is different from the original PER (Program Evaluation and Review Technique) in that PERT Time employs the single time estimate of CPM instead of the probabilistic approach to PERT. The use of PERT Time does not involve any phase of cost optimization or time compression, although it provides most of the information that one would normally require in the application of the CPM technique to a scheduling system. Dr. Jerome Wiest (2) of UCLA has published a dissertation on project scheduling with limited resources. The actual technique is not clearly defined in Dr. Wiest's publication so that one cannot fully grasp the usefulness or the applications area of his technique. C-E-I-R, Inc. has a proprietary computer program called RAMPS (Resource Allocation and Multi-Project Scheduling) (3). This has some rather interesting heuristics for determining job scheduling priorities. It will be assumed here that RAMPS may have the necessary algorithm for cost optimization and even time compression.

However, one will not know this until the proprietors of RAMPS decide on publishing the algorithmic process used in their computer program. Many other companies and individuals are working toward some new methods in the use of CPM as a scheduling and planning tool.

In the present use of CPM, the time estimate for each activity in a network is treated independently from the cost for accomplishing the activity which of course embodies both direct and indirect costs. These different costs may involve such things as manpower load, equipment, etc. By examining any activity in a CPM network, it is readily seen that an activity has definable costs both direct and indirect which are related to time or the duration for that particular activity. One is then faced with the question as to whether these cost factors can be used in an optimum fashion to determine a time estimate for an activity.

The next chapter shows the mathematical development of the optimization procedure for a network planning system directly relating it to a time-compression procedure. Chapter 4 of this thesis gives a thorough treatment of the routine for time-compression purposes.

C H A P T E R I I I

MATHEMATICAL DEVELOPMENT OF THE OPTIMIZATION TECHNIQUE

For any activity in a CPM network, a time estimate is assigned with the assumption that the available men and equipment for this activity are sufficient to complete it on schedule. Prior to the start of the project, the various costs for each activity are definable. Also, the factors involved from one activity to another are going to be similar. These various costs can be used to determine the optimum time and manpower load for an activity. Different companies or individuals may label the costs in a different way. The following variables are definable for each activity:

N = Number of men assigned to the activity

M = The fixed number of man-hours required to accomplish an activity

C_t = The time variable cost related to the activity in dollars/unit-time

C_s = Initial fixed cost necessary for the activity under consideration

C_b = The time variable manpower cost in dollars/unit-time

C_e = The load cost due to placing an employee on the job

From the above variables, one can compute the total cost for an activity which would be:

$$K = \frac{M}{N} C_t + C_b M + C_e N + C_s \qquad (1)$$

In equation (1), K is a function of N, since all the other factors are fixed.

With the application of differential calculus, one can obtain the minimum cost as follows:

Form:
$$\frac{dK}{dN} = -\frac{M}{N^2} C_t + C_e \qquad (2)$$

By equating equation (2) to zero, one can obtain the optimum value of the number of men for each activity, say N_o, this gives:

$$-\frac{M}{N^2} C_t + C_e = 0$$

Therefore

$$N_o = \sqrt{\frac{M\ C_t}{C_e}} \qquad (3)$$

Since the time for each activity is equal to $\frac{M}{N}$; an optimum time for an activity can be obtained by replacing the value of N with the optimum number of men N_o.

Therefore, one can obtain the optimum time T_o as follows:

$$T_o = \frac{M}{N_o} = \sqrt{\frac{M\ C_e}{C_t}}$$

From equation (1), one can obtain the total cost for an activity under optimum conditions by replacing equation (3) for the value N. This will give the following results:

Let K_o = The optimum cost

$$K_o = \frac{M}{N_o} C_t + C_b\ M + C_e\ N_o + C_s$$

$$K_o = \frac{M}{\sqrt{\frac{M\ C_t}{C_e}}}\ C_t + C_b\ M + C_e\sqrt{\frac{M\ C_t}{C_e}} + C_s$$

which gives

$$K_o = 2\sqrt{M\ C_t\ C_e} + C_b\ M + C_s$$

A sample activity will be shown here to illustrate the time cost relationship that can be obtained from the above equations.

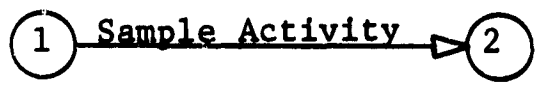

Figure 1. Sample Activity

Figure 1 is a sample activity with the following definable values:

$M = 32$

$N = 3$

$C_t = 1$

$C_s = 2$

$C_b = 7$

$C_e = 2$

The optimum time for this activity will be:

$$T_o = \sqrt{\frac{M\ C_e}{C_t}}$$

$$= \sqrt{\frac{(32)\ (2)}{1}} = 8$$

The optimum total cost here is:

$$K_o = 2\sqrt{M\ C_t\ C_e} + C_b\ M + C_s$$

$$= 2\sqrt{(32)\ (1)\ (2)} + (7)\ (32) + 2$$

$$= \$242.00$$

The optimum value of N is:

$$N_o = \sqrt{\frac{M\ C_t}{C_e}}$$

$$= \sqrt{\frac{(32)\ (1)}{2}}$$

$$= 4$$

The above values show an optimum activity duration of 8 units of time. These units may be in days, weeks, or even months.

In considering the sample activity, it is trivial to see that the duration of 8 units of time may be compressed or reduced in a way that the increase in cost will follow the path of minimum cost. It will then be necessary to determine the change in cost each time the activity duration is reduced by one unit of time.

Let K_o = the optimum or original cost

K_1 = cost of the activity after compression of time T (one unit at a time)

therefore since $T = \frac{M}{N}$;

$$K_o = T_o\ C_t + C_b\ M + C_e\ \frac{M}{T_o} + C_s$$

and

$$K_1 = T_1\ C_t + C_b\ M + C_e\ \frac{M}{T_1} + C_s$$

The increase or change in cost after time compression is designated as ∇K and this is equal to:

$$\nabla K = K_1 - K_o$$

$$= (T_1 - T_o)\, C_t + C_e\, M\, (\frac{1}{T_1} - \frac{1}{T_o})$$

But since compression is achieved one unit at a time,

$$T_1 - T_o = -1$$

and

$$\frac{1}{T_1} - \frac{1}{T_o} = \frac{1}{T_o\,(T_o - 1)}$$

One will now have the change in cost ∇K as:

$$\nabla K = \frac{C_e\, M}{T_o\,(T_o - 1)} - C_t$$

Referring back to the sample activity of Figure 1, one can obtain the change in costs (∇K) each time the activity duration is reduced by one unit of time.

Table 1 shows the different total costs for the sample activity each time the duration undergoes compression. It is also shown that by increasing the duration above the optimum time of 8 units, the total cost also increases.

From the cost values on Table 1, a time versus cost curve can be shown indicating the non-linear relationship between the decrease in time and its corresponding cost increase. Figure 2 shows this relationship with the ordinate representing the total cost and the abscissa scale representing the time or duration.

TABLE 1

COST VALUES

Duration	Total Cost
12	$ 246.33
11	245.82
10	245.40
9	245.11
8	242.00
7	242.52
6	243.66
5	245.86
4	249.19
3	258.86
2	289.86

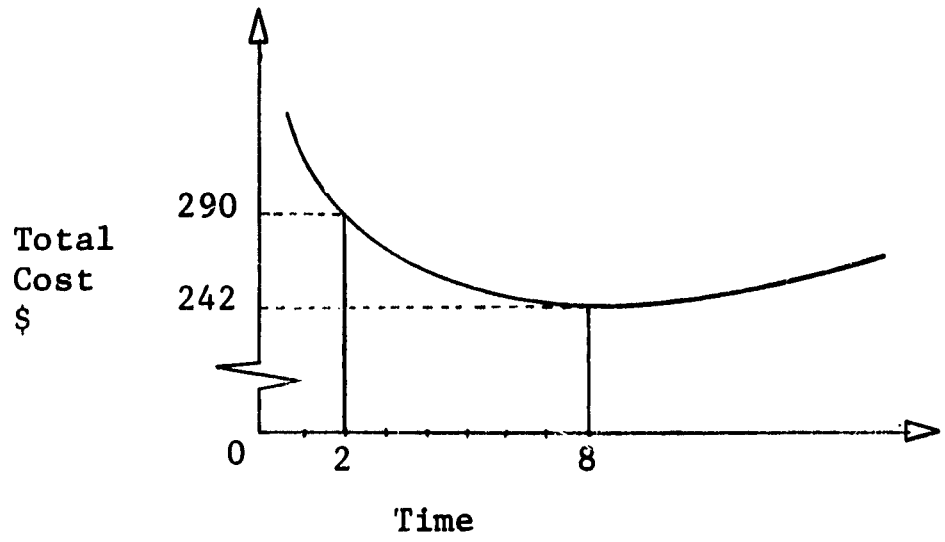

Figure 2. Time Versus Cost Curve

The determination of the increase in cost from the optimum value may be obtained either by direct computational methods as

shown earlier, or the cost values can be read directly from the curve shown in Figure 2.

C H A P T E R I V

TIME COMPRESSION

The time compression routine involves the reduction of the project duration by one unit at a time. The algorithm for this process produces the path of minimum cost. Two constraints may be applied to this procedure, these are time constraint and cost ($) constraint. Each of these constraints will be explained separately although the effect of both constraints in a single case will also be fully explained in another section.

Time Constraint. Figure 3 is a simple CPM network with a single critical path. All the activities with darkened arrowheads are critical, i.e. the activities along path 1, 3, 4, and 6.

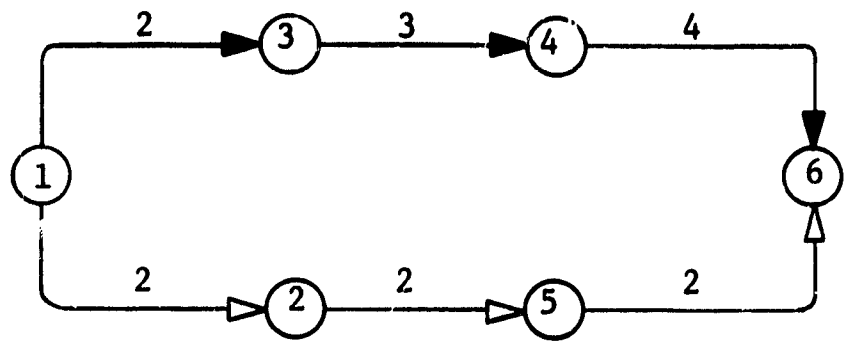

Figure 3. CPM Network A

Table 2 gives the different variables related to the activities of Figure 3. The optima values for the total cost, duration, and the number of men for each activity are also given. These values were computed by using the formulae developed in Chapter 3.

TABLE 2

COST TABLES

Activity	M	N	C_e	C_t	C_s	C_b	K_o	T_o
1-3	4	2	1	1	3	2	15	2
3-4	9	3	3	3	2	2	38	3
4-6	16	4	5	5	3	3	91	4
1-2	16	8	1	4	2	5	98	2
2-5	16	8	1	4	2	5	98	2
5-6	16	8	1	4	2	5	98	2

As one notes from Figure 3, the duration for this network is 9 units of time. It will be assumed that one wishes to reduce the project duration to 7 units of time. By inspection, one can reduce the duration of the entire network by reducing by one unit of time the duration of any one of the activities along the critical path. This will indeed reduce the network duration by one unit of time but one will not have the assurance that the time reduction will follow the path of minimum cost.

In the previous chapter, a development of the mathematical model has been shown and by applying the equation to determine the change in cost with a time reduction of one unit one finds that activity (1-3) has a ∇K of 1; activity (3-4) has a ∇K of 1.5; and activity (4-6) has a ∇K of 1.68 dollars (refer to Table 2 for the variables to compute ∇K).

At this stage one is confronted with 3 activities along a path, any one of which may be compressed by one unit of time to reduce the network duration by one unit of time. The ∇Ks as mentioned earlier are as follows:

$$\nabla K_{1-3} = 1$$

$$\nabla K_{3-4} = 1.5$$

$$\nabla K_{4-6} = 1.68$$

One immediately notices that activity (1-3) with a ∇K equal to 1 has the lowest change in cost along the critical path. Further investigation reveals that the duration of this activity is greater than 1. One will therefore pick this activity as the best choice for time compression at this stage. Activity (1-3) has a duration of 2 units of time and reducing its duration by one unit will mean that this duration will be down to just 1 unit of time.

The network duration at this stage is 8 units of time. The time constraint for this problem is seven units which means that further compression is required.

Before going to the next time compression stage, one will need to check the entire network to see if a new critical path has developed. An examination of the network shows that at this stage the critical path is still the same.

By inspection, one notices that of the 3 activities along the critical path only activity (3-4) and activity (4-6) may be considered for time compression. This is so because activity (1-3) now has a duration of only one unit of time which is the minimum

duration that an activity may have unless it represents a dummy activity (an activity with a duration of zero).

At this point, it will be quite simple to decide on which activity to reduce by one unit of time since only two activities are involved. The change in cost or ∇K for these two activities are as follows:

$$\nabla K_{3-4} = 1.5 \quad (1)$$

$$\nabla K_{4-6} = 1.68 \quad (2)$$

The above results have been obtained by going through the usual computations for the change in cost for any activity to be considered for time compression. This procedure is the same as the one used in the previous stage where the network duration was reduced from 9 units of time to 8 units.

As one examines equations (1) and (2) one will immediately decide on activity (3-4) as the activity to be reduced time-wise since the change in cost (∇K) is only 1.5 dollars compared to the change in cost for activity (4-6) which is 1.68 dollars.

It is necessary to examine the entire network again and go through the usual steps involved in critical path computations.

By inspection, one notices that the total duration for this particular network has already been reduced to 7 units of time which is the time constraint for the network. It is then decided that the project duration has been reduced from the total duration of 9 units of time to a duration of 7 units of time at minimum cost.

One might ask the question as to the increase in cost after the project duration was reduced from 9 units of time to 7 units. This increase in cost may be obtained by summing the change in cost each time the network duration is reduced by one unit of time until the time constraint is reached. The total cost of the project will then be equal to the result after summing the ∇Ks plus the original total cost of the project before time compression was applied.

By applying the above procedure to the network illustration, one can sum up the total of ∇Ks which is $\nabla K_1 + \nabla K_2$ or 2.5 dollars. The original cost of the project before applying the time compression procedure was 438 dollars. With these results it is easily seen that the total cost of the project after the time compression routine will be in the amount of 440.5 dollars.

Cost Constraint. A cost constraint may also be applied in the time compression procedure. Basically, the algorithm will be the same as the procedure used in reaching the total project duration of 7 units of time. The main difference, however, is that each time the project duration is reduced by one unit of time one will sum up the total cost of the project at that stage and compare the cost for that stage to a set cost constraint. The main assumption here will be that time compression will always increase the total project cost.

A time compression with a cost constraint may be applied to the network represented in Figure 3. It will be assumed that the various costs for each activity will remain the same. The cost constraint for the project will be set at 440.5 dollars.

The first stage for this procedure will be the computation of the initial total cost of the entire project. Summing up the costs for all the activities give a total initial cost of 438 dollars. Since this is less than the set cost constraint, one can conclude that there is a possibility for the time compression of the total project duration.

The critical path calculations give the result of just one critical path with a duration of 9 units of time. This is also the total project duration, and the total cost of the project at this stage (before time compression) is at a minimum.

The section on Time Constraint gives the steps to be followed in compressing the time for the network of Figure 3. This will be the same procedure one will follow to reach the cost constraint.

As was shown earlier, the first stage for time compression may be achieved by reducing activity (1-3) by one unit of time since this activity has the lowest change in cost (∇K). By reducing activity (1-3) by one unit of time, the entire network duration may be reduced to 8 units of time. The decrease in the project duration increases the total project cost. In this case, the increase in cost for this stage is 1 dollar which gives a total project cost of 439 dollars. Reference to the cost constraint shows that the total project cost at this stage is still below the cost constraint of 440.5 dollars. Therefore, further time compression is possible.

The reduction of activity (1-3) by one unit of time still gives the same critical path along the network. But only two out of

the three activities along the critical path may be compressed time-wise since activity(1-3) now has a duration of one unit of time, and as mentioned in earlier sections of this thesis, no activity may be compressed if it has a duration of one unit or less.

It will now be possible to go through the compression of activity (3-4) which has a ∇K of 1.5 dollars. Activity (3-4) has been chosen for time compression at this point since activity (4-6) has a ∇K of 1.68 dollars. This reduces the project duration by one unit of time. One will now realize that the total project duration has been reduced to 7 units of time with a total project cost of 440.5 dollars. Reference to the cost constraint shows that the total cost of the project at 7 units of time is the minimum time that may be acceptable in terms of cost since at this point (7 units of time) the total project cost has reached the 440.5 - dollar cost constraint. Therefore, further compression of the activity duration may not be applied.

It will be mentioned here that the illustration on the time constraint problem and also the cost constraint problem have been worked out to have the same results so as to better acquaint the reader with the basic algorithm used in this thesis.

Time and Cost Constraint. In the previous sections, the sample problem dealt with a specific constraint - either cost or time. It can also be shown that a network for a project can be bounded by both time and cost constraints. Project planners interested in the area of time compression may be faced with a limited amount of cash although the reduction of the project duration can be reduced as far as

possible as long as it is within the set cost constraints.

It will then be very likely that when a project has a set time constraint a limiting amount of cash will also be a boundary to the project duration.

Therefore, one can conclude that in order to have the simultaneous cost and time constraints, it will be necessary to specify the priority of constraints, and the time or duration must be minimized subject to the cost constraint.

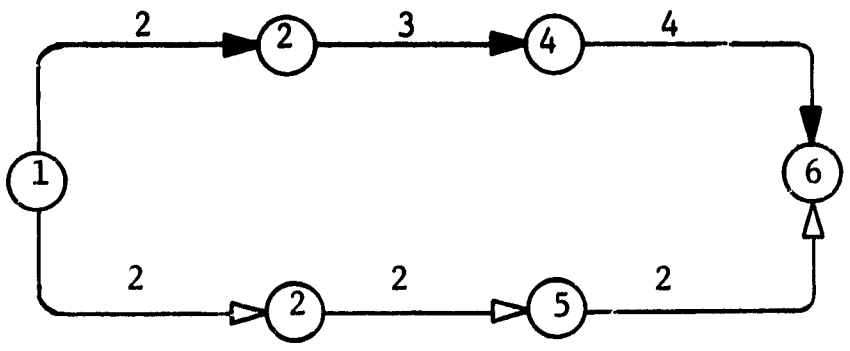

Figure 4. Revised CPM Network

Figure 4 is a reproduction of Figure 3 where the total project duration is 9 units of time and a total project cost of 438 dollars as the minimum cost.

One involved in the project construction may wish to reduce the project duration to 7 units of time at a cost of 440.5 dollars. However, the finance officer of the project may claim that the maximum amount to be spent on this specific project should not exceed 439 dollars. As one will recall, 439 dollars is the total cost of the project if its duration is reduced to 8 units of time. One can

therefore conclude that because of cost limitation, the project may not be completed until after the $8^{\underline{th}}$ unit of time even though the project planner's wish was to finish the entire project in 7 units of time.

With both time and cost constraints going into a time compression procedure, the basic algorithm in the solution to the problem will still be the same as the routine used in either the problem with the cost constraint or the problem with the time constraint except that each time the project duration is reduced by one unit of time, the total project cost for that stage must be compared with the given cost constraint to insure that there will not be any possibility of oversubscribing the available amount of cash for the project.

C H A P T E R V

MORE THAN ONE CRITICAL PATH

The figure illustrated in Figure 5 is the same type of network shown in Figure 4 of Chapter 4 except for the total project duration which is 7 units of time in Figure 5 while the total duration of the network in Figure 4 is 9 units of time.

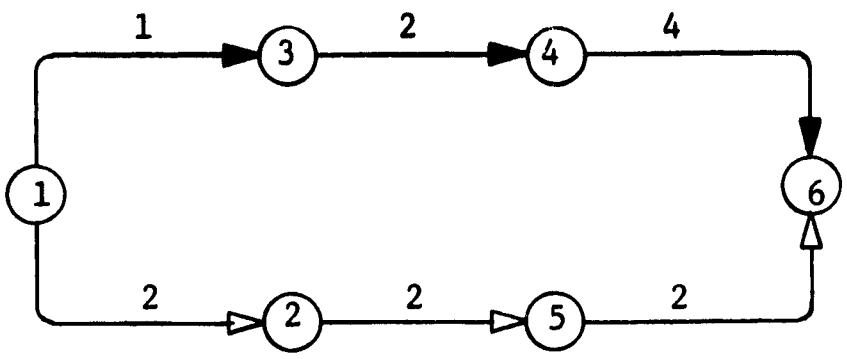

Figure 5. Compressed CPM Network A

It was pointed out that even though the duration of the network illustration has been reduced from 9 units of time to 7 units, the critical path through the network remained the same - that is, no new critical paths were developed. Of course in Figure 5 only 2 paths exist through the network, and therefore, only 2 possible critical paths can exist.

In the critical path analysis, all the activities with zero slack are considered critical. Figure 6 shows that the activities along nodes 1, 3, 4, 6 are critical activities while those along path 1, 2, 5, 6 do not have zero slack and therefore, do not follow the path of criticality.

However, the network in Figure 5 may still be compressed timewise. Further compression of the time or the duration in any network will also mean that eventually, more than one critical path will exist in a network, and there will always be the possibility that all the activities in a network will become critical.

Referring back to Figure 5, activities (1-3), (3-4), and (4-6) are along the critical path. Of these three activities, only activities (3-4), and (4-6) may be compressed since activity (1-3) now has a duration of only one unit of time. The ∇K for the two activities are as follows:

$$\nabla K_{3-4} = 10.5$$

$$\nabla K_{4-6} = 1.68$$

From the above information, one will then choose activity (4-6) as the activity to be reduced by one unit of time. This time reduction will result in Figure 6.

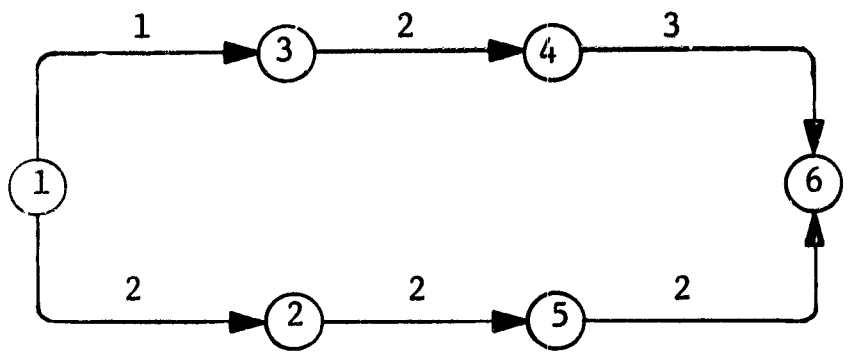

Figure 6. Compressed CPM Network B

Figure 6 gives a total network duration of 6 units of time and a total project cost of 442.18 dollars.

Critical path analysis further shows that at this stage all the activities in the network are critical. This means that none of the activities in the network have slack greater than zero.

For time compression to be applicable in Figure 6, it will be necessary to examine the two critical paths along the network. This is so because it is quite trivial to see that if only one path is reduced by one unit of time, the total duration of the project will still be 6 units of time.

This is a case where one is faced with parallel paths. The time compression routine can still be applied with each stage of the time compression following the path of minimum cost.

One can achieve the step-by-step time compression at minimum cost by choosing one activity from each path that has the least change in cost after its duration is reduced by one unit of time. By going through the algebraic computation for the change in cost for each activity in the network one concludes that along path 1, 3, 4, 6 activity (4-6) has the least ∇K which is equal to 8.3 dollars. Path 1, 2, 5, 6 gives activity (1-2) as the activity with the least change in cost and this is equal to 4 dollars. (This is a case where all the activities along a path have equal ∇Ks. We therefore, take the liberty of choosing the first activity along the path). One will then have to reduce both activity (4-6) and activity (1-2) by one unit of time in order to reduce the total project duration by also one unit of time. This stage results in a total project duration of 5 units of time. Further compression of the duration is possible by following

the above procedure. One will also notice by inspection that the total duration of the network in Figure 6 cannot be compressed any further after the total duration is reduced to 3 units of time. This is so because there are only 3 activities in each path and as mentioned in earlier statements, an activity may not be compressed any farther if its duration is less than or equal to 1 unit of time.

So far, the illustration for the time compression procedure involved a simple CPM network with six activities and only two paths along the network. One realizes, however, that in an actual case a CPM network may have hundreds or even thousands of activities and plenty of paths that are critical can exist through the network.

The time compression procedure at minimum cost can also be applied to large scheduling networks. However, hand or manual computations will be tedious if not an impossible process. Appendix I illustrates the step-by-step process involved in compressing an 11-activity network, and although this is a small network, it is shown that hand computation results in a long process.

However, a computer can simplify the whole process of both the CPM and the time compression analysis. For a computer to be useful in this area, a program is necessary and this program is described in Chapter 7.

The Routine for Time Compression. It will now be well to present the steps that one might follow in order to accomplish the time compression algorithm.

(1) From a given set of data, compute the optimum cost (K_o) and the duration (T_o) for each activity.

(2) Sum all the activity costs and the result will be the optimum total cost for that project.

(3) Perform the CPM analysis and the resulting total project duration will be the optimum duration for the project.

(4) Set the constraints.

(5) List all the activities along the critical path or paths in the network.

(6) Compute the change in cost (∇K) for every activity that is critical if its duration is reduced by one unit of time. It must be noted here that any activity with a duration of one unit or less cannot be considered for time compression.

(7) Pick the activity or the set of activities that will give the least change in cost ∇K when the total project duration is reduced by one unit of time. The figure representing the change in cost after one unit of time compression is added to the previous total cost of the project and the result gives the total project cost at this stage with a duration of T_i units of time.

(8) Compare the project duration and or total cost at this stage with the set time and or cost constraint. If either the time and cost constraint has been reached computation ceases, or else, perform the usual CPM analysis using the present duration for each activity. Repeat steps 5 through 8.

C H A P T E R V I

THE ALGEBRAIC ANALOGUE FOR TIME COMPRESSION

In Chapter 5, a case example was shown regarding a CPM network that has more than one critical path. One might have noticed that the application of the time compression routine to a CPM network with more than one critical path will present some problems since it will be necessary to pick a combination of activities that will give the minimum cost increase after the project duration is reduced by one unit of time.

The case example in Chapter 5 was an easy case for time compression purposes since only two parallel paths existed in the network, and of course, in practical cases, a network may have hundreds of activities with many critical paths. The critical paths in a large network may not always be parallel to each other. There will always be the possibility of crossed paths in a network that are all critical and to apply the time compression routine to this type of network, one will have to pick a set or combination of activities that will reduce the total project duration by one unit of time when each of the duration of the activities in the set or combination are also reduced by one unit of time. One will also note that there will be many sets or combinations of activities that may be considered for time compression. It will then be necessary to enumerate all the possible sets or combinations and from these sets or combinations one will take the specific set or combination of activities that will reduce the total project duration by one unit of time at a minimum cost increase.

An algebraic analogue will be presented here in regard to the problem of enumerating the many possible combinations of activities that can be used to reduce the total project duration by one unit at a time. It will be assumed that more than one critical path can exist in a CPM network and the illustrative example for the algebraic analogue will further assume that all the activities are critical with each activity having a duration greater than one unit of time.

Figure 7 is a simple CPM network with crossed paths and as mentioned earlier, all the activities in the network are critical.

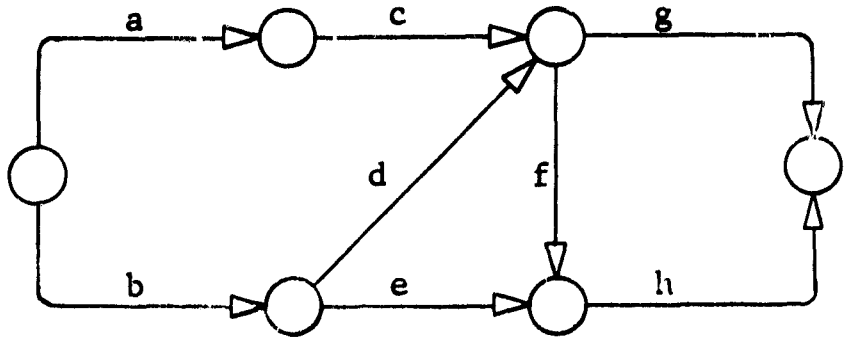

Figure 7. CPM Network B

It must be noted here that for illustrative purposes, the nodes are left unmarked and each activity in the network is identified by a letter of the alphabet.

The network in Figure 7 has 8 activities, and by visual means, it is readily apparent that there are five possible paths through the network. These paths are:

(1) a - c - g

(2) a - c - f - h

(3) b - d - g

(4) b - d - f - h

(5) b - e - h

By visual inspection, one can deduce that a number of activity combinations can be formed wherein the reduction of one unit of time from each activity in any particular combination will also reduce the total project duration by one unit of time. It will then be necessary to figure out all these possible sets or combinations since each set or combination of activities will in most cases yield a different change in cost (defined as ∇K in the previous chapters) when the total project duration is reduced by one unit of time, and as mentioned earlier, the first choice for time reduction will be the set or combination of activities yielding the least change in cost ∇K.

The algebraic analogue for selecting the time reduction possibilities will start with a tabular set-up listing all the possible critical paths in a network. Table 3 lists all the paths from the CPM network in Figure 7 since it was earlier assumed that all the paths in the network are critical.

One can then concentrate on a specific path from Table 3 , and from this path, choose the first activity and mark this activity with an asterisk (*). This activity is labeled as activity a. Inspect all the other paths, (i.e. paths 2 through 5), and see if any of the paths have activity a along its route. One finds that only path number 2 has activity a in its route, and it will then be necessary to eliminate path number 2 from further consideration for this specific combination.

Each activity along path number 2 will then be crossed out to indicate its elimination.

TABLE 3

LIST OF PATHS (A)

Path #	Activities Along a Path			
1	a	c	g	
2	a	c	f	h
3	b	d	g	
4	b	d	f	h
5	b	e	h	

One will now check the remaining paths, (i.e. paths 3, 4, and 5), and see if any activity in these paths are the same as the other activities of path number 1 (i.e. activities c and g). By inspection, one will cross out activity g of path number 3, and all the rest will be left uncrossed since neither activity c or activity g is present in the other paths. (i.e. paths 4 and 5).

TABLE 4

LIST OF PATHS (B)

Path #	Activities Along a Path			
1	*a	~~c~~	~~g~~	
2	~~a~~	~~c~~	~~f~~	~~h~~
3	b	d	~~g~~	
4	b	d	f	h
5	b	e	h	

Table 4 represents the development to this stage. It must be noted here that activity c and activity g of path number 1 are also eliminated from further consideration.

Now, the next step will be to choose the first uncrossed activity in Table 4 and mark this activity with an asterisk. This is activity b of path number 3. It will be necessary to inspect the remaining paths, (i.e. paths 4 and 5), and see if activity b is present in either one of these paths. One will find that both paths 4 and 5 contain activity b along its route so that these last two paths will be eliminated from further consideration. Activity d of path number 3 will also be crossed out.

TABLE 5

LIST OF PATHS (C)

Path #	Activities Along a Path			
1	*a	~~c~~	~~g~~	
2	~~a~~	~~c~~	~~g~~	~~d~~
3	*b	~~c~~	~~g~~	
4	~~b~~	~~c~~	~~g~~	~~d~~
5	~~b~~	~~e~~	~~h~~	

Table 5 shows the nature of the problem at this stage. It can be noted here that activity a and activity b are the only activities with asterisks before it and all the other activities have been crossed out so that one can conclude that reducing activity a and activity b by one unit of time will give one possible combination that will reduce

the total project duration by one unit of time.

Another possible combination may be obtained by referring back to Table 4 where one can mark an asterisk before activity d of path number 3 instead of the previous choice where one choosed activity b of path number 3. Choosing activity d of path number 3 will automatically eliminate path number 4 from consideration at this stage since activity d is also present in path number 4. This choice will also cross out activity b of path number 5 since activity b is also in path number 3.

Table 6 shows the nature of the problem at this stage.

TABLE 6

LIST OF PATHS (D)

Path #	Activities Along a Path			
1	*a	~~c~~	~~g~~	
2	~~a~~	~~c~~	~~g~~	~~k~~
3	~~b~~	*d	~~g~~	
4	~~b~~	~~d~~	~~g~~	~~k~~
5	~~b~~	e	h	

By inspection, one will note that two more activities in path number 5 will have to be considered for this specific combination, and of course, marking either one of these activities (i.e. either activity e or activity h) with an asterisk will eliminate the other activity from further consideration.

Table 7 shows the final set-up for this specific combination. We now have another set or combination of activities which will reduce the total project duration by one unit of time if each activity in this specific combination is also reduced by one unit of time. These activities are a, d, and e.

TABLE 7

LIST OF PATHS (E)

Path #	Activities Along a Path
1	*a
2	
3	*d
4	
5	*e

So far, two possible sets or combinations of activities have been obtained either one of which may be used to compress the total project duration by one unit of time. It is quite obvious that many other sets or combinations of activities do exist in the network of Figure 8, but the routine for finding the other remaining combinations will be similar to the method employed in obtaining the combinations illustrated in Table 3 through Table 7

Obtaining all the possible combinations in either a small network or a large network will be a tedious job when done by hand computations, but utilizing a computer will allow for the practical application of this algebraic analogue.

It must also be noted here that the time compression computer program listed in Appendix II utilizes this algebraic analogue.

One might come to the conclusion that this particular method of obtaining the right combination of activities to compress the total duration of a network by one unit at a time will be quite impractical for CPM networks with many critical paths. Of course, this will not always be the case since crossed paths will always exist in a network and with more paths crossing each other, the combination possibilities will also be greatly reduced.

In regard to cases of large networks with many critical paths that are all parallel to each other, the application of the algebraic analogue presented here will not be recommended since, one can just go through each critical path and from each path, choose the activity that will give the least change in cost ∇K when its duration is reduced by one unit of time.

C H A P T E R V I I

THE COMPUTER PROGRAM

A computer program written in FORTRAN language has been developed to accomplish both the CPM and the time compression analysis. This program has been written for the IBM 7094 computer (9) although with little modification, it can be used in most data processing installations that have a FORTRAN compiler in its system.

The imput to the program is similar to any CPM-type problem except for the addition of the direct and indirect cost values related to each activity.

Programming Procedure. The computer program has 3 main parts. The first part accomplishes the usual CPM analysis where activity slack is figured out. This stage also outputs on a reel of tape all the activities that are critical.

The second part of the program utilizes the tape containing the activities that are critical and the data on this reel of tape is processed in the core storage. The processing at this stage involves the search for all the possible paths that are critical in the network. This process is relatively simple if only one path in a network is critical. But when more than one path exists, (i.e. if all activities in the network are critical), the determination of all possible paths will be quite complicated and in large networks a computer would be a necessity. This section of the program outputs on another reel of tape all the critical paths through the network and now the stage

is set for the third part which is the part where the time compression occurs.

Basically, the computer algorithm for the time compression is the same as the procedure described in the section where time and cost constraints were taken into account. This algorithm utilizes the algebraic analogue described in Chapter 6. The whole routine just goes through the procedure of choosing the activity(s) that will give the least cost increase when its total duration is reduced by one unit of time.

The entire programming procedure is utilized each time the network duration is reduced by one unit of time and computation ceases whenever either or both time and or cost constraints have been reached.

Figures 15 through 20 of Appendix II illustrate the flow charts which show the computer algorithm. Appendix II also contains a listing of the computer program with instructions on how to prepare the input data.

C H A P T E R V I I I

CONCLUSION

The time compression algorithm for the critical path method will be of great value to the CPM analyst as a scheduling and planning tool and also as a subject for further research.

Broad applications of the computer program is in sight, and of course, additional development and testing of the model presented in this thesis is underway.

In regard to the aspect of further research in this area, it is assumed here that some programming formulation may be applied as an extension to the algebraic analogue presented in Chapter 6. This extension should be such that it will eliminate the necessity of enumerating all the possible combinations for time compression in cases where more than one critical path exist in a network, and these paths are not all parallel to each other.

However, the present state of the computer program is applicable to medium-sized projects. The program is written in a high level computer language (FORTRAN IV), and therefore, compatible to most large-scale computer systems.

It can also be mentioned here that the approach for determining the optimum time compression represents a significant advance in network scheduling theory; however, several extensions are clearly open for further development. The possible extensions will be mentioned at this time.

The total project duration frequently has an associated cost function representing rental on equipment and facilities, penalties, bonuses, etc. This will result in a monotonic increasing cost function which is the dual to the activity cost function as treated in this study. If the two functions were incorporated then the optimization procedure could be extended to determine a minimum cost criterion using these opposing functions.

Although it is readily seen that this routine will produce an optimum cost compression routine, it should be pointed out that the routine is dependent upon the time interval used in compression. If a truly mathematical technique from the area of non-linear programming could be found this would not be the case. However, it was found that the existing mathematical programming techniques will not work on the type of functions used in this study. This is another rather complicated extension which is beyond the scope of this thesis.

However, I can mention here that Professor Leonard Lamberson, a candidate for the Doctor of Philosophy degree, is completing a dissertation that will give a full treatment of the above-mentioned problem areas. This research is also sponsored by the Industrial Engineering Department, Texas A&M University.

There are many models that are being developed in project scheduling, and the worth of the various scheduling rules can only be measured through experience in their use. Time compression is just one of these models. As a whole, it is believed that with the aid of the computational power of the digital computer to most approaches to

project scheduling, it will contribute in many ways to solving some of the complex scheduling problems of large project management.

A P P E N D I C E S

APPENDIX I

THE APPLICATION OF TIME COMPRESSION TO A CPM NETWORK

The time compression algorithm is presented here in regard to its application to a small CPM network. One will note that each CPM network has been drawn on a time scale to provide the reader with a visual observance each time the network is compressed by one unit of time.

The time scale does not have any specific unit representation so that it may represent days, weeks, or even months.

The dotted lines shown on some activities indicate that any path going through that particular activity is not critical. It can also be used to indicate the amount of slack along a path.

Table 8 shows the different cost values related to each activity. No specific cost unit is used here but one can just assume that each cost unit represents a hundred dollars. The optimum time and total cost for each activity has also been figured out using the formulae derived in Chapter 3 to find the optima values. The formulae are as follows:

1. Optimum value of N:

$$N_o = \sqrt{\frac{M\ C_t}{C_e}}$$

2. The optimum time:

$$T_o = \sqrt{\frac{M\ C_e}{C_t}}$$

3. Optimum cost:

$$K_o = 2\sqrt{M\ C_t\ C_e} + C_b\ M + C_s$$

The change in cost for an activity that is compressed by one unit of time has also been computed for those activities that are critical under optimum conditions. These values are listed under the heading " ∇K."

For each stage of the solution to this problem, a similar type of information is shown.

TABLE 8

DATA LISTING

Activity	M	C_t	C_s	C_b	C_e	N	T_o	K_o	∇K
1-2	4	3	2	1	3	2	2	18	3
1-3	1	2	1	5	2	1	1	10	
1-4	9	1	3	2	1	3	3	27	
2-5	16	4	4	3	4	4	4	84	1 1/3
3-5	4	4	2	3	4	2	2	30	
3-6	25	5	4	3	5	5	5	129	
4-7	4	5	1	2	5	2	2	2[illegible]	
5-8	16	6	1	2	6	4	4	81	2
6-9	1	3	3	4	3	1	1	13	
7-9	9	2	2	3	2	3	3	41	
8-10	16	4	2	3	4	4	4	82	1 1/3
9-10	4	3	4	3	3	2	2	28	

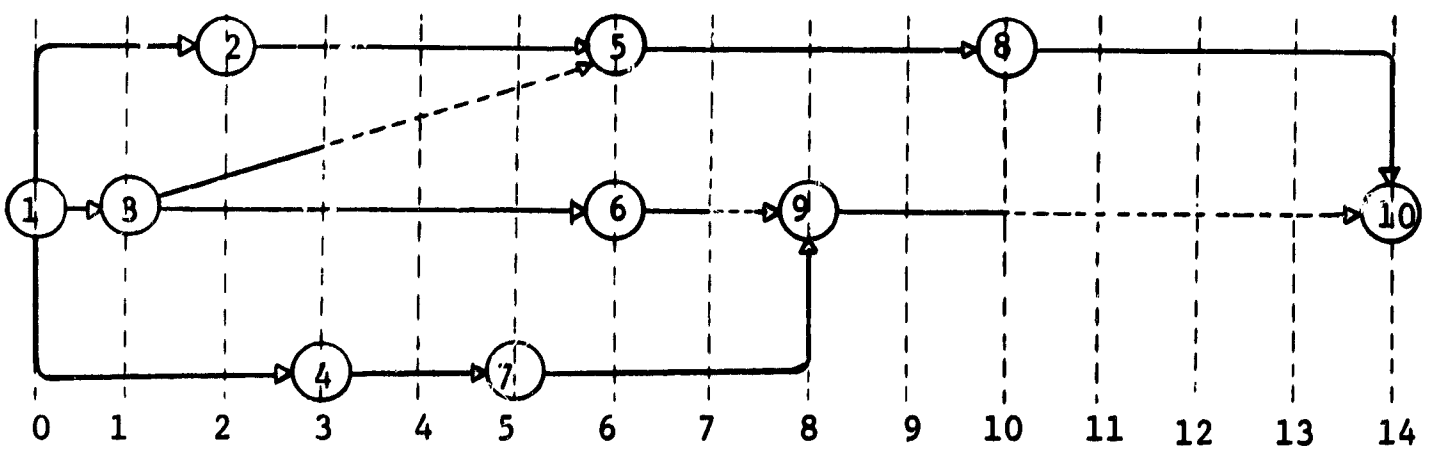

Figure 8. Sample CPM Network Before Time Compression

Step number 1.

Optimum project duration:	14 Units.
Project duration at this stage:	14 Units.
Optimum total cost:	572.00.
Critical path(s) at this stage:	1-2-5-8-10.
Activity(s) to be reduced:	(2-5).
Change in cost:	1 1/3.
Cost constraint:	600.00.

Remarks: The change in cost for both activity (2-5) and activity (8-10) are equal and both are along the critical path. For consistency, the activity with the earliest finish time will be compressed first.

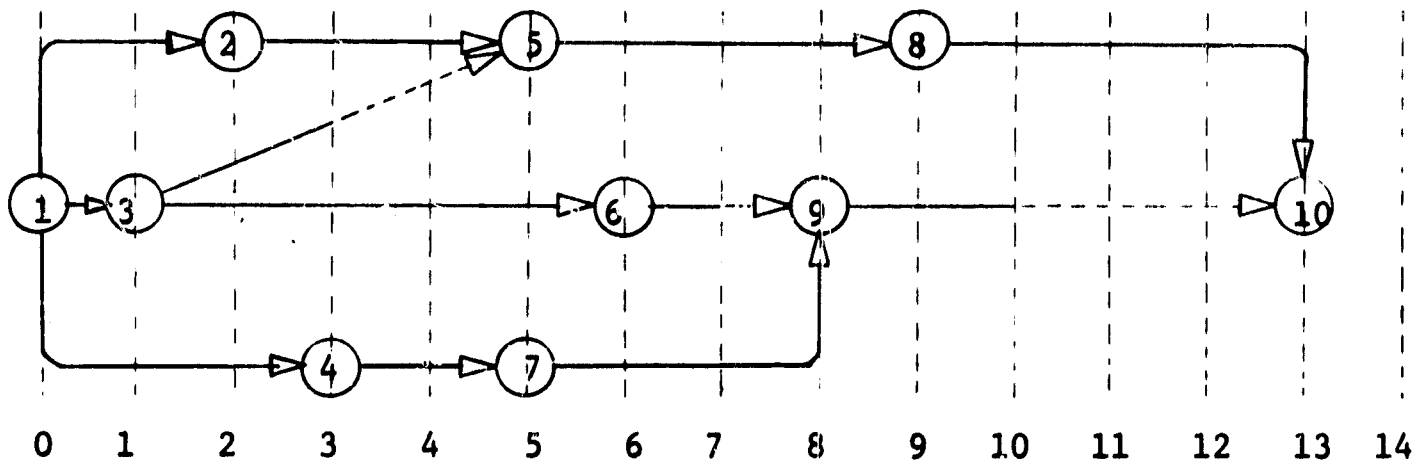

Figure 9. Sample CPM Network (after step 1)

Step number 2.

Optimum project duration:	14 Units.
Project duration at this stage:	13 Units.
Total Cost at this stage:	573 1/3.
Critical path(s) at this stage:	1-2-5-8-10.
Activity(s) to be reduced:	(8-10).
Change in Cost:	1 1/3.
Cost Constraint:	600.00.

Remarks: Activity (8-10) has the lowest change in cost among the activities along the critical path, therefore, it is the best choice for time compression at this stage.

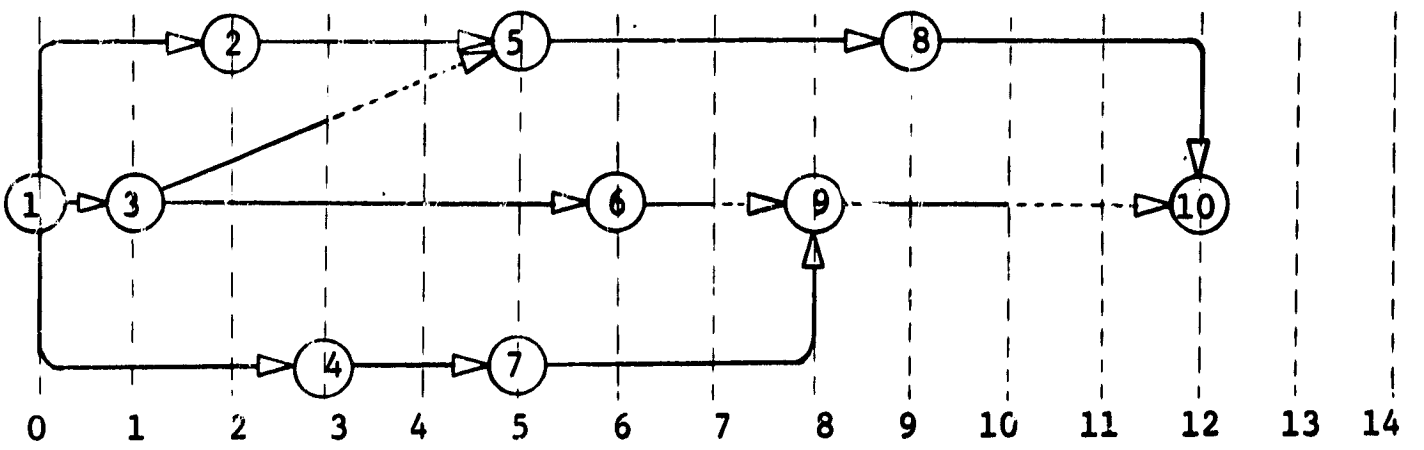

Figure 10. Sample CPM Network (after step 2)

Step number 3.

Optimum project duration:	14 Units.
Project duration at this stage:	12 Units.
Total cost at this stage:	575 2/3.
Critical path(s) at this stage:	1-2-5-8-10.
Activity(s) to be reduced:	(5-8).
Change in cost	2.
Cost constraint:	600.00.

Remarks: It will be noted here that so far the critical path has remained the same.

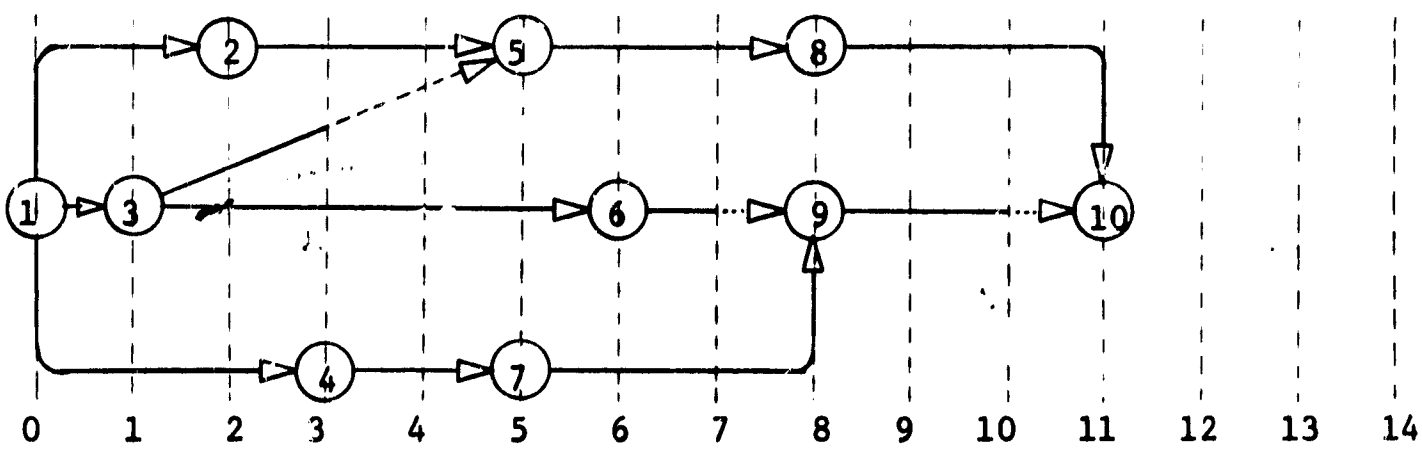

Figure 11. Sample CPM Network (after step 3)

Step number 4.

Optimum project duration:	14 Units.
Project duration at this stage:	11 Units.
Total cost at this stage:	577 2/3.
Critical path(s) at this stage:	1-2-5-8-10.
Activity(s) to be reduced:	(1-2).
Change in cost:	3.
Cost constraint:	600.00.

Remarks: The critical path is still the same.

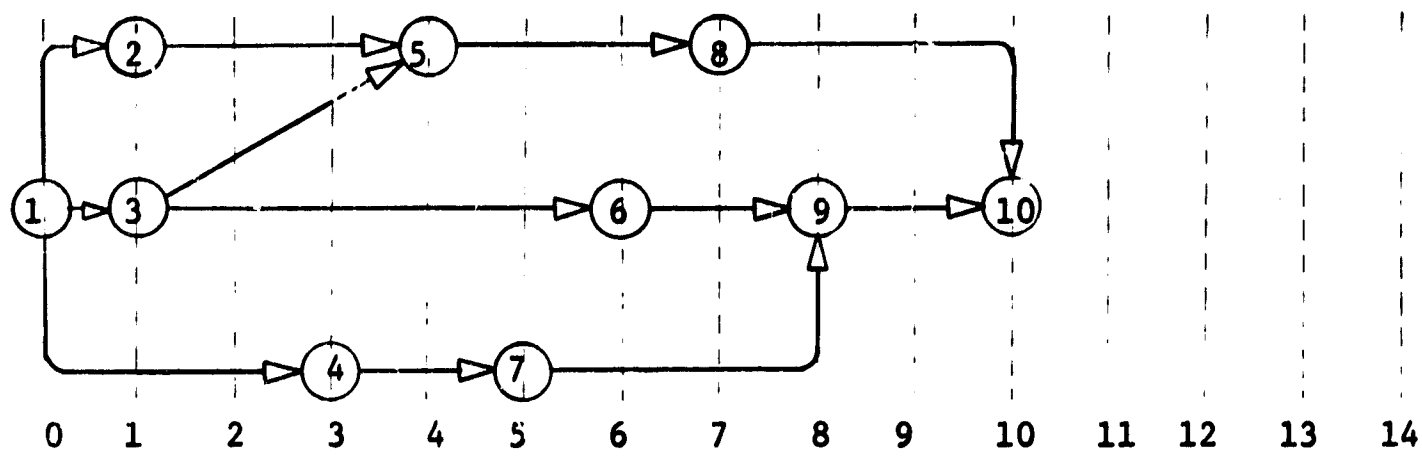

Figure 12. Sample CPM Network (after step 4)

Step number 5.

Optimum project duration:	14 Units.
Project duration at this stage:	10 Units.
Total cost at this stage:	580 2/3.
Critical path(s) at this stage:	(1-2-5-8-10), and (1-4-7-9-10).
Activity(s) to be reduced:	(2-5), and (1-4).
Change in cost:	7 1/6.
Cost constraint:	600.00.

Remarks: There are now two paths along the network that are critical so that two activities (one from each path) will have to be compressed time-wise in order to reduce the total network duration by one unit of time.

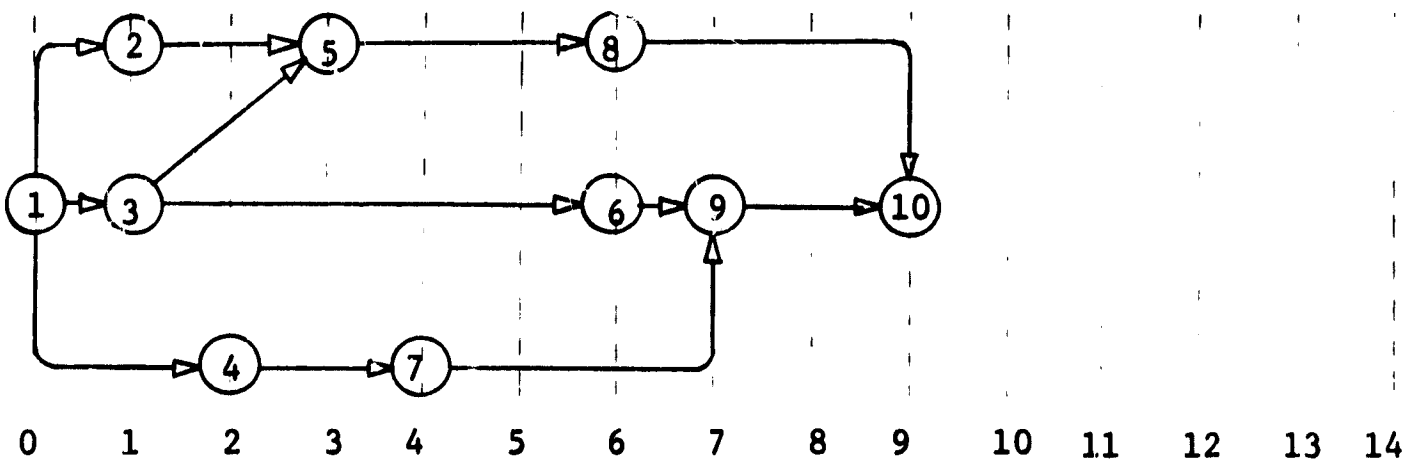

Figure 13. Sample CPM Network (after step 5)

Step number 6.

Optimum project duration:	14 Units.
Project duration at this stage:	9 Units.
Total cost at this stage:	587 5/6.
Critical path(s) at this stage:	(1-2-5-8-10), (1-3-6-9-10), (1-4-7-9-10), and (1-3-5-8-10).
Activity(s) to be reduced:	(8-10), and (9-10).
Change in cost:	9 2/3.
Cost constraint:	600.00.

Remarks: This is a case where all the activities in the network are critical. One will have to pick an activity or a set of activities that will reduce the project duration by one unit of time at minimum cost. These activities are (8-10), and (9-10).

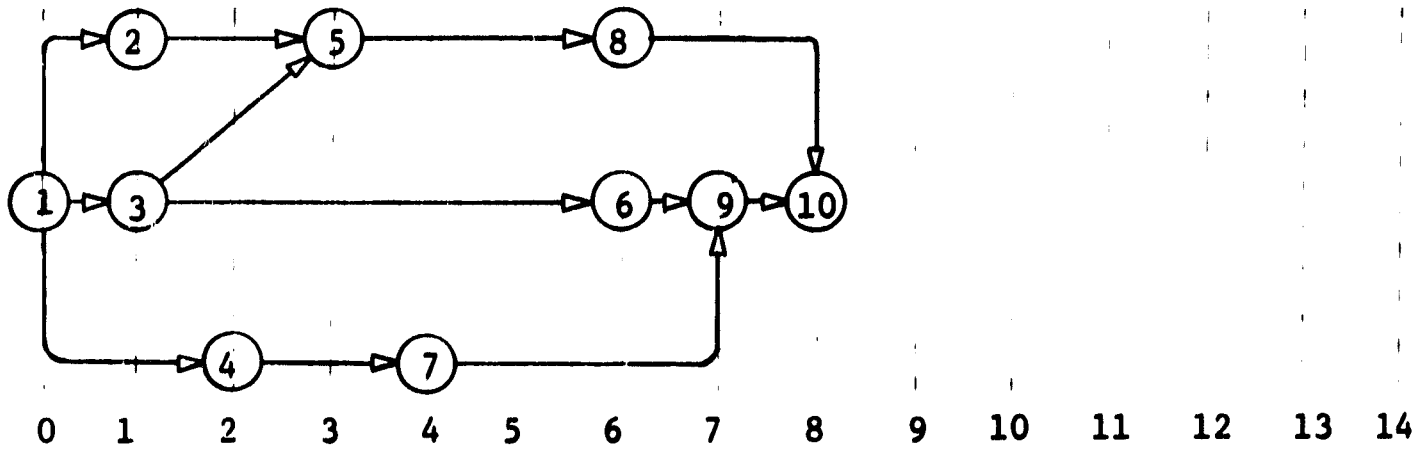

Figure 14. Sample CPM Network (after step 6)

Step number 7.

Optimum project duration:	14 Units.
Project duration at this stage:	8 Units.
Total cost at this stage:	597 1/2.
Critical path(s) at this stage:	(1-2-5-8-10), (1-3-6-9-10), (1-4-7-9-10), and (1-3-5-8-10).
Activities to be reduced:	(5-8), (3-6), and (7-9).
Change in cost:	12 1/4.
Cost constraint:	600.00.

Remarks: One will now realize that the total cost for this stage is almost equal to the cost constraint,and it can be noted that further compression of the duration will increase the total cost such that it will become greater than the set cost constraint. Therefore, with a given cost constraint of 600.00, the total network duration will be 8 units of time.

APPENDIX II

FLOW DIAGRAMS AND PROGRAM LISTINGS

This section contains the flow diagrams and listings of the subroutines that make up the source program for the time compression algorithm.

The entire program contains five subroutines plus the main calling program.

Subroutine Orland inputs the data from tape number 5 so that the input data may be processed by subroutine Fincpm which takes care of the CPM calculations.

Subroutine Phil merges the results obtained from subroutine Fincpm to incorporate the cost parameters related to each activity. The output of subroutine Phil is processed by subroutine Crpath which utilizes the time compression of the activity or set of activities chosen by subroutine Crpath.

The main calling program checks the results obtained from subroutine Merge to see if the constraints have been reached, otherwise, the whole cycle is repeated starting with a call on subroutine Fincpm.

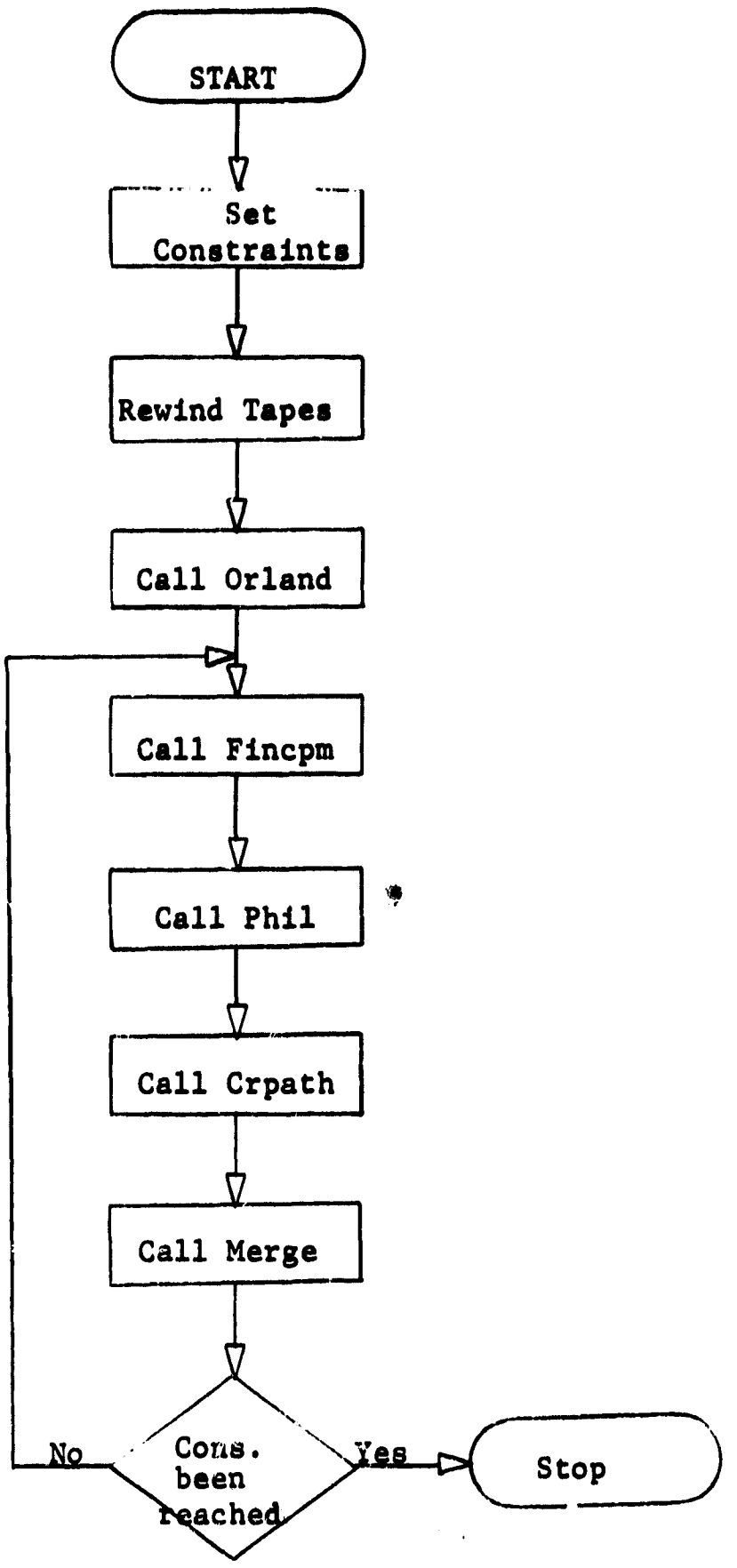

Figure 15. Main Program

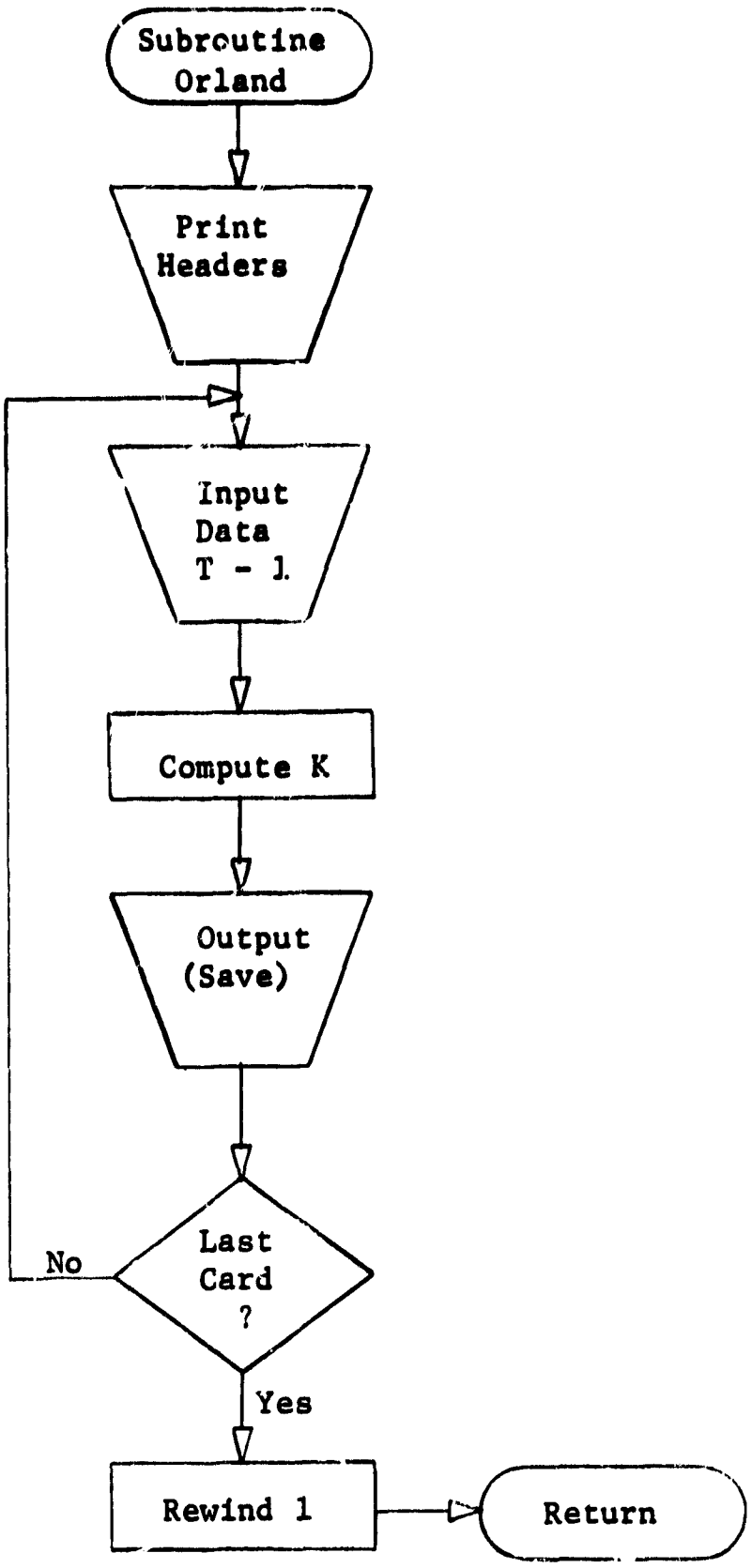

Figure 16. Subroutine Orland

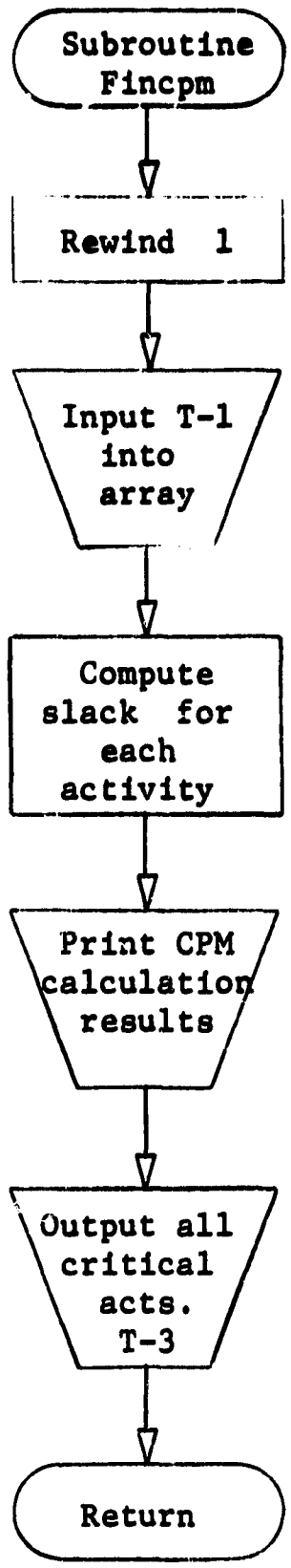

Figure 17. Subroutine Fincpm

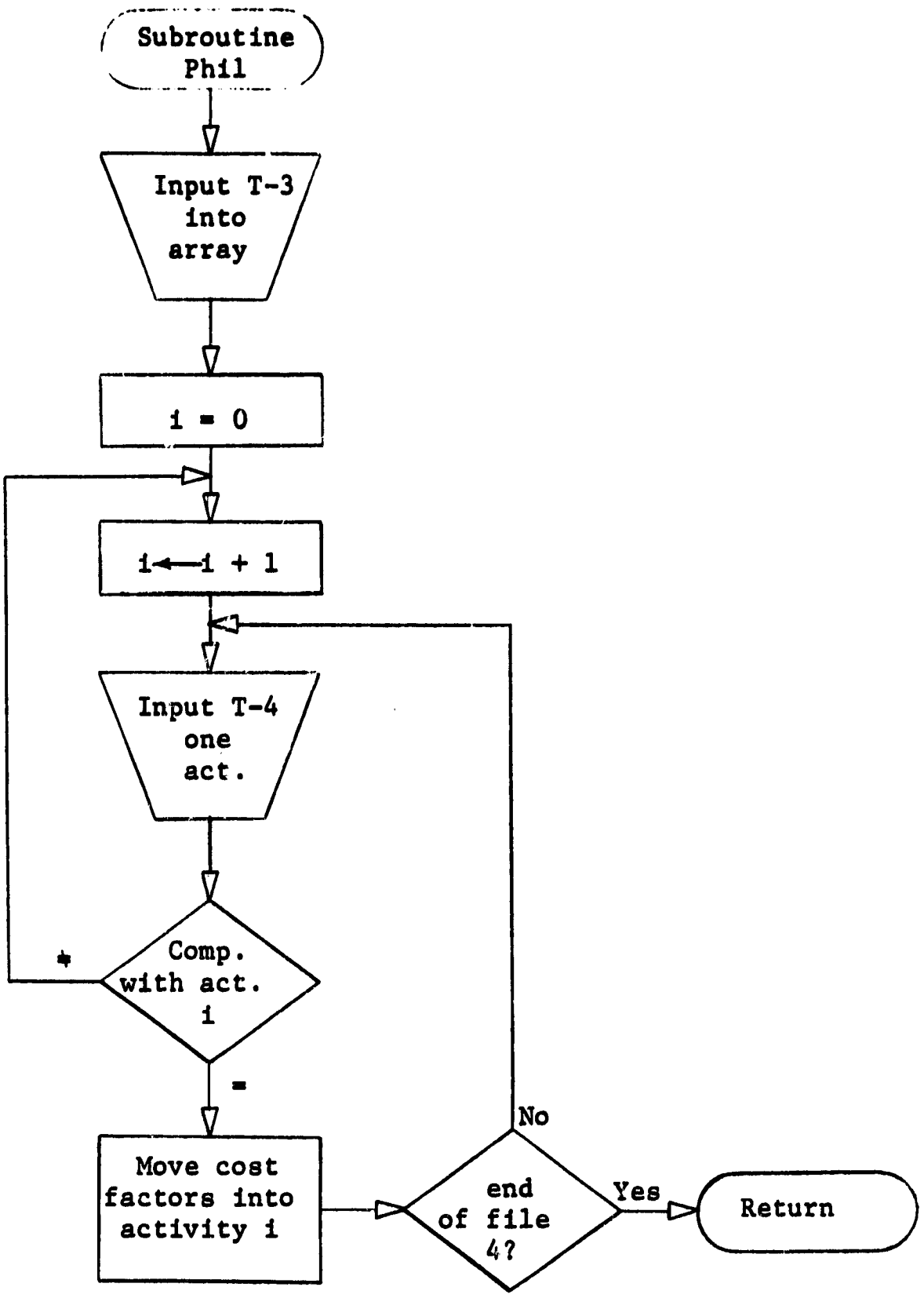

Figure 18. Subroutine Phil

Figure 19. Subroutine Crpath

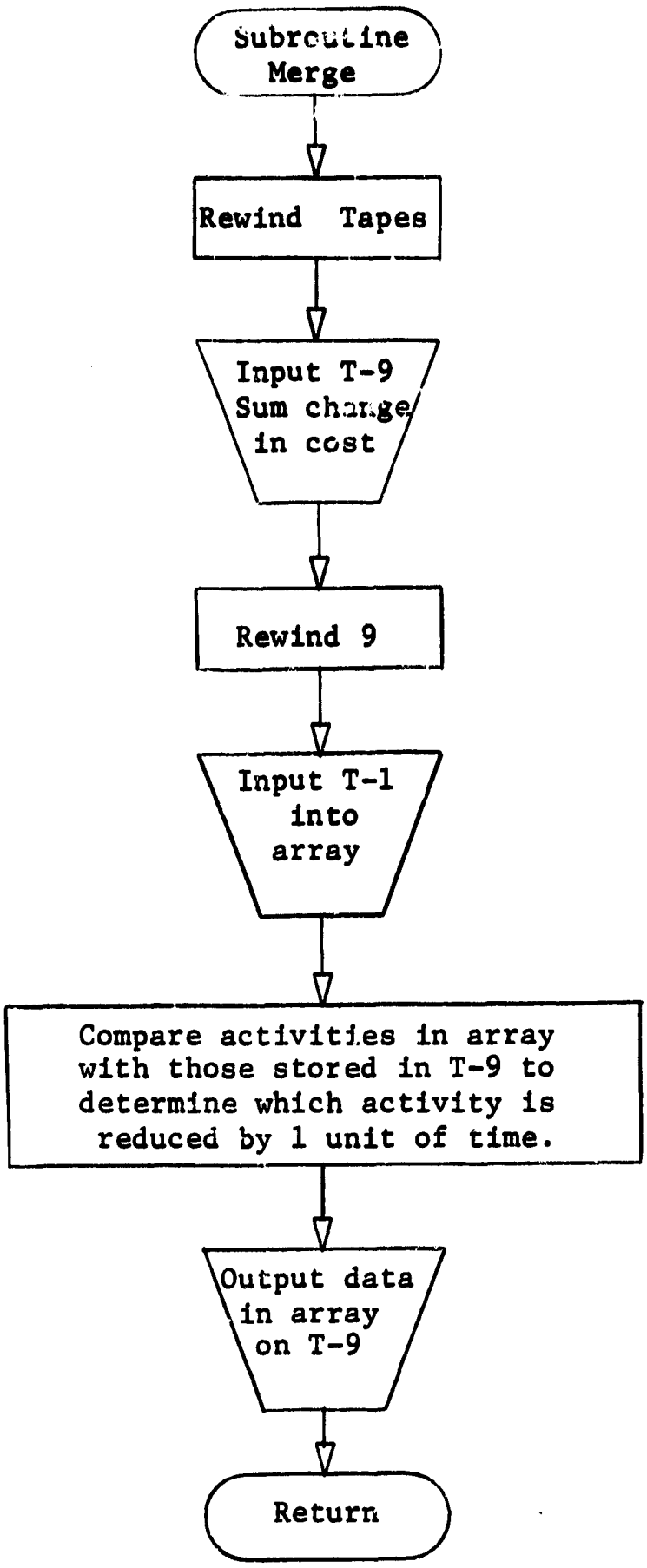

Figure 20 . Subroutine Merge

```
$JOB  918015366T1005003000650343  MADRIGAL,ORLANDO S. CPM TIME COMP.
$IBBOX          03-B      ORLANDO S. MADRIGAL  CPM TIME COMPRESSION THESIS
$* PLEASE MOUNT SCRATCH TAPES ON LOGICAL 1,2,3,4,7,8,9,10
$EXECUTE        IBJOB
$IBJOB OSM      FIOCS,MAP
$IBFTC ORL
       DIMENSION ICPATH(300),NAME(4,300),NONE(300),NTWO(300),INT(300),
      1ITF(300)
       DIMENSION MM(300),MN(300),MT(300),MS(300),MB(300),ME(300)
       COMMON/B/ICPATH,NAME,NONE,NTWO,INT,ITF,MM,MN,MT,MS,MB,ME
       COMMON/BB/KKOST,KCOST,KONS
       COMMON/C/NAM(4)
       COMMON/BBB/KCOSE
       COMMON/A/KTIME
C
C    ORLANDO SIA MADRIGAL    INDUSTRIAL ENGINEERING
C     TEXAS A+M UNIVERSITY    1967
C
C THIS PROGRAM UTILIZES THE TIME COMPRESSION ALGORITHM
C
C  INPUT DATA MUST BE AS FOLLOWS
CFIRST CARD COLUMN 1-6 AMOUNT OF TIME REDUCTION
C                      7-12  COST CONSTRAINT
C 3 CARDS MUST FOLLOW THE ABOVE - THESE CARDS MAY BE LEFT BLANK
C OR IT CAN REPRESENT THE PROJECT HEADINGS
C ACTIVITY CARDS ARE AS FOLLOWS
C   COLUMN
C    1-5  PREDECESSOR NODE
C    6-10  SUCCESSOR NODE
C   11-14  MAN HOURS REQUIRED FOR THE ACTIVITY
C   15-18  NUMBER OF MEN ASSIGNED TO THE ACTIVITY
C   19-22  TIME VARIABLE COST
C   23-26  INITIAL FIXED COST
C   27-30  TIME VARIABLE MANPOWER COST
C   31-34  LOAD COST DUE TO PLACING AN EMPLOYEE ON THE JOB
```

```
C  AFTER THE ACTIVITY CARDS, A CARD WITH ZEROS ON FIRST 5 COLUMNS
C MUST FOLLOW
C  LAST CARD MUST HAVE THE WORD END ON THE FIRST 3 COLUMNS
      KONS=0
      READ(5,101)KONSS,KONST
101   FORMAT(2I6,68X)
      REWIND 3
      CALL ORLAND
10    CALL FINCPM
      CALL PHIL
      CALL CRPATH
      CALL MERGE
      WRITE(6,102)KCOSE
102   FORMAT(1HO,1X,28HTHE OPTIMUM TOTAL COST IS $ ,I6//)
      WRITE(6,100)KCOST
100   FORMAT(1HO,1X,34HTHE TOTAL COST AT THIS STAGE IS $ ,I6//)
      WRITE(6,103)KONST
103   FORMAT(1HO,1X,25HTHE COST CONSTRAINT IS $ ,I6//)
      KTIME=KTIME-1
      WRITE(6,104)KTIME
104   FORMAT(1HO,31H THE DURATION AT THIS STAGE IS ,I6//)
      WRITE(6,105)KONS
105   FORMAT(1HO,18H THIS DURATION IS ,I6,3X,33HTIME UNITS BELOW THE OPT
     1IMUM TIME//)
      IF(KONS.GE.KONSS)GO TO 111
      IF(KCOST.GE.KONST)GO TO 111
      GO TO 10
111   RETURN
      END
$ORIGIN        CHAS
$IBFTC READ2
      SUBROUTINE PHIL
      DIMENSION ICPATH(300),NAME(4,300),NONE(300),NTWO(300),INT(300),
     1ITF(300)
      DIMENSION MM(300),MN(300),MT(300),MS(300),MB(300),ME(300)
```

```
      COMMON/B/ICPATH,NAME,NONE,NTWO,INT,ITF,MM,MN,MT,MS,MB,ME
      REWIND 3
      REWIND 4
      REWIND 8
      IZ=1
      WRITE(6,100)
100   FORMAT(1H1)
      DO 1 I=IZ,300
40    IF(EOF(3).NE.0.) GO TO 20
      READ(3,400)ICPATH(I),(NAME(J,I),J=1,4),NONE(I),NTWO(I),INT(I),
     2ITF(I)
200   FORMAT(A6,4A6,2I4,2I6,6I4)
30    IF(EOF(4).NE.0.)GO TO 50
51    READ(4,300)IPRE,ISUC,NM,NN,NT,NS,NB,NE
300   FORMAT(2I4,6I4,2X)
      IF(IPRE-NONE(I))30,10,30
10    IF(ISUC-NTWO(I))30,11,30
11    MM(I)=NM
      MN(I)=NN
      MT(I)=NT
      MS(I)=NS
      MB(I)=NB
      ME(I)=NE
      GO TO 1
50    REWIND 4
      GO TO 51
1     CONTINUE
400   FORMAT(A6,4A6,2I4,2I6,4X)
20    IZ=1
      MZ=I-1
      DO 2 I=IZ,MZ
      WRITE(8,200)ICPATH(I),(NAME(J,I),J=1,4),NONE(I),NTWO(I),INT(I),
     3ITF(I),MM(I),MN(I),MT(I),MS(I),MB(I),ME(I)
2     CONTINUE
      END FILE 8
```

```
      REWIND 8
      RETURN
      END
$ORIGIN        CHAS
$IBFTC PERT
      SUBROUTINE FINCPM
      DIMENSION NAME(4,300),NODE1(300),NODE2(300),INT(300),ITS(999),
     1     ITL(999),NCRIT(300),IALFA(4),JOBIND(300),NPRIOR(300)
      DIMENSION IHEDER(4)
      COMMON/B/NAME,INT
      REWIND 3
      WRITE (6,2000)
      READ (1,1001) (IALFA(I),I=1,4)
1     IF(EOF(1).NE.0.)RETURN
      NOGO=0
      IZ = 1
      IERR = 0
      NN = 0
      ICRIT = 0
   72 DO 3 I=IZ,300
      READ (1,1001) (NAME(J,I),J=1,4), JOBIND(I), NPRIOR(I), NODE1(I),
     1     NODE2(I), INT(I)
C
C        THE FOLLOWING SECTION DETERMINES THE TYPE OF CARD READ
C                  JOB CARD -- THE DATA IS CHECKED AND ENTERED INTO CORE
C                  HEADER CARD -- THE TITLE IS READ INTO CORE
C                  END CARD -- THE READING OPERATION CEASES
C
      IF (NODE1(I)) 6,63,5
C
C        JOB CARD DATA CHECK
C
    5 IF (INT(I)) 6,303,303
  303 IF (NODE2(I)) 6,6,3
   70 M = 4
```

```
      GO TO 38
    6 IF (IERR) 2,2,30
    2 M = 1
   38 WRITE (6,2007) (NAME(JO,1),JO=1,4)
      IERR = 1
      GO TO (30,307,1407,71,213,183), M
   30 WRITE (6,2001) (NAME(J,I),J=1,4)
      NOGO = 1
      GO TO 3
   71 WRITE (6,2010)
      IZ = 2
      NOGO = 1
      GO TO 72
C
C        END OR HEADER CARD CHECK
C
   63 JOBIND(I) = 1
      IF (IALFA(1) - NAME(1,I)) 3,64,3
   64 DO 65 J=2,4
      IF (IALFA(2) - NAME(J,I)) 3,65,3
   65 CONTINUE
   67 NNN = I - 1
      GO TO 9
    3 CONTINUE
C
C        IF TOO MANY CARDS READ FOR DIMENSION SIZE, EXECUTION DELETED
C
 8000 IF (IERR) 7000,7000,7100
C
C        THE FOLLOWING SECTION CHECKS  NODES 1 AND 999 FOR VALIDITY
C
    9 NSEERR = 1
      DO 180 I=1,NNN
      IF (NODE1(I) - 1) 180,181,180
  180 CONTINUE
```

```
      GO TO 190
  307 WRITE (6,2009) NODE2(IP)
      NOGO = 1
      GO TO 1107
 1407 WRITE (6,2008) NODE1(IP)
      NOGO = 1
      GO TO 1007
  181 DO 182 I=1,NNN
      IF (NODE2(I) - 1) 182,190,182
  182 CONTINUE
      DO 185 I=1,NNN
      IF (NODE1(I) - 999) 185,190,190
  185 CONTINUE
      DO 184 I=1,NNN
      IF (NODE2(I) - 999) 184,199,190
  184 CONTINUE
  190 IF (IERR) 191,191,183
  191 M = 6
      GO TO 380
 7000 M = 4
  380 WRITE (6,2007) (NAME(JO,1), JO = 1,4)
      IERR = 1
      GO TO (3100,307,1407,7100,213,183), M
 3100 WRITE (6,2001) (NAME(J,I), J = 1,4)
      NOGO = 1
      GO TO 8000
 7100 WRITE (6,2010)
      IZ = 2
      NOGO = 1
      GO TO 72
721   RETURN
  183 WRITE (6,2012)
      NOGO = 1
C
C         THE FOLLOWING SECTION DETERMINES THE PRIORITY LEVEL OF EACH
```

```
C         JOB AND CHECKS FOR CYCLING ERRORS
C
  199 NP = 0
  210 NP = NP + 1
      IADD = 0
      DO 200 IP=1,NNN
      IF (JOBIND(IP)) 201,201,200
  201 DO 203 JP=1,NNN
      IF (JOBIND(JP)) 204,204,203
  204 IF (NODE1(IP) - NODE2(JP)) 203,205,203
  203 CONTINUE
      GO TO 200
  205 NPRIOR(IP) = NP
      IADD = 1
  200 CONTINUE
      DO 206 IP=1,NNN
      IF (NPRIOR(IP) - NP) 207,206,206
  207 JOBIND(IP) = 1
  206 CONTINUE
      IF (IADD) 208,208,209
  209 IF (NP - NNN) 210,210,211
  211 IF (IERR) 212,212,213
  212 M = 5
      GO TO 380
  213 WRITE (6,2011)
      DO 214 IP=1,NNN
      IF (JOBIND(IP)) 215,215,214
  215 DO 216 JP=1,NNN
      IF (JOBIND(JP)) 217,217,216
  217 IF (NODE2(IP) - NODE1(JP)) 216,218,216
  216 CONTINUE
      GO TO 214
  218 WRITE (6,2006) NODE1(IP), NODE2(IP), (NAME(J,IP),J=1,4)
  214 CONTINUE
      NOGO = 1
```

```
C
C         THE FOLLOWING SECTION CHECKS EACH NODE FOR A LEAVING JOB AND
C         AN ENTERING JOB
C
  208 DO 35 IP=1,NNN
      IF (NODE1(IP) - 1) 35,35,170
  170 DO 171 JL=1,NNN
      IF (NODE1(IP) - NODE2(JL)) 171,35,171
  171 CONTINUE
      IF (IERR) 111,111,42
  111 M= 3
      GO TO 380
   42 WRITE (6,2008) NODE1(IP)
      NOGO = 1
   35 CONTINUE
 1007 DO 34 IP = 1,NNN
      IF (NODE2(IP) - 999) 172,34,34
  172 DO 173 JL=1,NNN
      IF (NODE2(IP) - NODE1(JL)) 173,34,173
  173 CONTINUE
      IF (IERR) 114,114,37
  114 M = 2
      GO TO 380
   37 WRITE (6,2009) NODE2(IP)
      NOGO = 1
   34 CONTINUE
 1107 IF (NOGO) 80,80,1
C
C         THE FOLLOWING SECTION CALCULATES THE EARLY TIME FOR EACH NODE
C
   80 DO 10 I=1,999
   10 ITE(I) = 0
      DO 12 J=1,NP
      DO 12 I=1,NNN
      IF (NPRIOR(I) - J + 1) 12,13,12
```

```
   13 NODE = NODE1(I)
      ITRY = ITE(NODE) + INT(I)
      NODE = NODE2(I)
      IF (ITE(NODE) - ITRY) 14,12,12
   14 ITE(NODE) = ITRY
   12 CONTINUE
C
C        THE FOLLOWING SECTION CALCULATES THE LATE TIME FOR EACH NODE
C
      DO 15 I=1,999
   15 ITL(I) = ITE(999)
      DO 16 J=1,NP
      K = NP - J
      DO 16 I=1,NNN
      IF (NPRIOR(I) - K) 16,17,16
   17 NODE = NODE2(I)
      ITRY = ITL(NODE) - INT(I)
      NODE = NODE1(I)
      IF (ITL(NODE) - ITRY) 16,16,18
   18 ITL(NODE) = ITRY
   16 CONTINUE
C
C        THE FOLLOWING SECTION CHECKS FOR HEADER CARDS AND OUTPUTS THEM
C
      DO 19 I=1,NNN
      IF (NODE1(I)) 20,20,21
   20 IF (NODE1(I+1)) 22,22,23
   22 WRITE (6,2002) (NAME(J,I),J=1,4)
      GO TO 19
   23 WRITE (6,2003) (NAME(J,I),J=1,4)
      GO TO 19
C
C        THE FOLLOWING SECTION CALCULATES THE EARLY START AND FINISH,
C        THE LATE START AND FINISH, AND THE FLOAT FOR EACH JOB
C
```

```
C          IF THE JOB HAS A BLANK NAME, IT IS IGNORED ON OUTPUT
C
   21 NONE = NODE1(I)
      NTWO = NODE2(I)
      IES = ITE(NONE)
      IEF = IES + INT(I)
      ILF = ITL(NTWO)
      ILS = ILF - INT(I)
      ITF = ILS - IES
      IFF = ITE(NTWO) - IEF
C
C          THE FOLLOWING SECTION CHECKS TO SEE WHETHER OR NOT THE JOB IS
C          ON THE CRITICAL PATH, SETS AN OUTPUT INDICATOR, AND STORES NODE
C          NUMBERS AS BEING ON THE CRITICAL PATH
C
      IF (ITF) 24,24,25
   24 ICPATH = IALFA(3)
      ICRIT = ICRIT + 1
      NCRIT(ICRIT) = I
   26 DO 95 J=1,4
      IF (NAME(J,I) - IALFA(2)) 91,95,91
   95 CONTINUE
      GO TO 19
   25 ICPATH = IALFA(2)
      GO TO 26
C
C          THE FOLLOWING SECTION OUTPUTS THE DATA FOR EACH JOB
C                    ASTERISKS BEFORE AND AFTER THE OUTPUT FOR A JOB
C                    INDICATE THE JOB IS ON THE CRITICAL PATH
C
   91 WRITE (6,2004) ICPATH, (NAME(J,I),J=1,4), NONE, NTWO, INT(I),
     1    IES, ILS, IEF, ILF, ITF, IFF, ICPATH
   19 CONTINUE
C
C          THE FOLLOWING SECTION OUTPUTS THE CRITICAL PATH
```

```
C
      ICTP1 = ICRIT + 1
      WRITE (6,2005) ICTP1
      DO 29 I=1,NP
      DO 28 J=1,ICRIT
      NTRY = NCRIT(J)
      IF (NPRIOR(NTRY) - I+1) 28,125,28
  125 WRITE (6,2006) NODE1(NTRY), NODE2(NTRY), (NAME(JO,NTRY),JO=1,4)
   28 CONTINUE
   29 CONTINUE
C
C        THE FOLLOWING SECTION ARRANGES THE JOBS IN ORDER OF ASCENDING
C        PRIORITY AND OUTPUTS THE CALCULATED DATA FOR ALL JOBS IN ORDER
C
      WRITE (6,2013)
      WRITE (6,2014)
      DO 401 I=1,NP
      WRITE (6,2016)
      DO 401 J=1,NNN
      IF (NODE1(J)) 403,403,402
  403 DO 404 K=1,4
  404 IHEDER(K) = NAME(K,J)
      GO TO 401
  402 IF (NPRIOR(J) - I + 1) 401,405,401
  405 NONE = NODE1(J)
      NTWO = NODE2(J)
      IES = ITE(NONE)
      IEF = IES + INT(J)
      ILF = ITL(NTWO)
      ILS = ILF - INT(J)
      ITF = ILS - IES
      IFF = ITE(NTWO) - IEF
      IF (ITF) 408,408,409
  408 ICPATH = IALFA(3)
      GO TO 410
```

```
  409 ICPATH = IALFA(2)
  410 WRITE (6,2015) ICPATH,I         ,(NAME(K,J),K=1,4),IHEDER,NONE,
     C   NTWO,INT(J),IES,ILS,IEF,ILF,ITF,IFF
      WRITE(3,2017)ICPATH,(NAME(K,J),K=1,4),NONE,NTWO,INT(J),ITF
2017  FORMAT(A6,4A6,I4,I4,I6,I6,4X)
  401 CONTINUE
2018  FORMAT(A6,48X)
      END FILE 3
      REWIND 3
      GO TO 1
C
 1001 FORMAT (4A6,2I3,3I5)
 2000 FORMAT (1H1/1HO/1H3,54X,22HCRITICAL PATH SCHEDULE////
     1      59X,14HO. S. MADRIGAL///
     2      53X,26HINDUSTRIAL ENGINEERING 691)
 2001 FORMAT (1HO,33X,19HDATA ERROR IN JOB (4A6,22H) -- EXECUTION DELETE
     1D)
 2002 FORMAT (1H1/1HO/1H3,53X,4A6)
 2003 FORMAT (1H1,54X,4A6////27X,8HJOB NAME,17X,4HSPAN,4X,8HDURATION
     1      ,7X,5HSTART,10X,6HFINISH,9X,5HFLOAT/72X,11HEARLY  LATE,
     2      4X,11HEARLY  LATE,4X,11HTOTAL  FREE//)
 2004 FORMAT (13X,5A6,5X,I4,3H TO,I4,I6,5X,2I6,3X,2I6,3X,2I6,2X,A6)
 2005 FORMAT (1H1,49X,19HTHE CRITICAL PATH (I3,10H NODES) IS)
 2006 FORMAT (1HO,43X,I4,3H TO,I4,9X,4A6)
 2007 FORMAT (1H1,54X,4A6///)
 2008 FORMAT (1HO,37X,33HDATA ERROR -- NO JOB ENDS AT NODEI4,21H -- EXEC
     1UTION DELETED)
 2009 FORMAT (1HO,36X,35HDATA ERROR -- NO JOB BEGINS AT NODEI4,21H -- EX
     1ECUTION DELETED)
 2010 FORMAT (1HO,46X,40HTOO MANY DATA CARDS -- EXECUTION DELETED)
 2011 FORMAT (1H4,36X,59HDATA ERROR -- THE FOLLOWING JOBS CYCLE -- EXECU
     1TION DELETED)
 2012 FORMAT (1HO,36X,59HDATA ERROR IN STARTING OR ENDING NODES -- EXECU
     1TION DELETED)
 2013 FORMAT (1H1,44X,43HCOMPLETE LIST OF ALL JOBS BY PRIORITY LEVEL////)
```

```
     C    )
 2014 FORMAT (1HO/3X,8HLEVEL OF,9X,8HJOB NAME,18X,7HSECTION,16X,4HSPAN,
     C    4X,8HDURATION,7X,5HSTART,10X,6HFINISH,9X,5HFLOAT/3X,8HPRIORITY,
     C    78X,11HEARLY  LATE,4X,11HEARLY  LATE,4X11HTOTAL  FREE///)
 2015 FORMAT (A6,I2,4X,4A6,2X,4A6,3X,I4,3H TO,I4,I6,5X,2I6,3X,2I6,3X,2I6
     C    )
 2016 FORMAT (1HO)
C
      RETURN
      END
$ORIGIN        CHAS
$IBFTC INCARD
      SUBROUTINE ORLAND
      COMMON/B/ID,AM,AN,CT,CS,CB,CE,NUM,NAME
      DIMENSION PIRST(19)
      DIMENSION FIRST(20)
      COMMON/BB/KKOST,KCOST
      COMMON/BBB/KCOSE
      DIMENSION NAME(3)
      REWIND 10
      REWIND 1
      REWIND 4
      KKOST=0
      WRITE(6,111)
111   FORMAT(1H1)
      READ(5,10)FIRST
      WRITE(1,10)FIRST
      WRITE(10,10)FIRST
      WRITE(6,10)FIRST
      READ(5,10)FIRST
10    FORMAT(20A4)
      WRITE(6,10)FIRST
      WRITE(1,10)FIRST
      WRITE(10,10)FIRST
      READ(5,10)FIRST
```

```
      WRITE(6,10)FIRST
      WRITE(1,10)FIRST
      WRITE(10,10)FIRST
      WRITE(6,500)
500   FORMAT(1H1,57X,18HORLANDO S MADRIGAL,57X/)
501   FORMAT(48X,36HDEPARTMENT OF INDUSTRIAL ENGINEERING,48X)
502   FORMAT(56X,20HTEXAS A+M UNIVERSITY,56X)
503   FORMAT(62X,5H 1966,61X//)
504   FORMAT(55X,23H**** PROGRAM INPUT ****,54X///)
505   FORMAT(1H ,3X,4HNODE,4X,13HACTIVITY NAME,5X,9HMAN HOURS,4X,
     26HNUMBER,4X,9HTIME COST,2X,10HFIXED COST,2X,8HMANPOWER,4X,4HLOAD,
     36X,7HNO. MEN,4X,8HDURATION,3X,10HTOTAL COST,2X)
506   FORMAT(1X,3HPRE,3X,3HSUC,33X,6HOF MEN,4X,10H(VARIABLE),1X,
     29H(INITIAL),4X,4HCOST,7X,4HCOST,5X,9H(OPTIMUM),3X,9H(OPTIMUM),
     32X,9H(DOLLARS),3X)
507   FORMAT(55X,7HDOLLARS,3X,7HDOLLARS,4X,7HDOLLARS,4X,7HDOLLARS,38X//)
      WRITE(6,501)
      WRITE(6,502)
      WRITE(6,503)
      WRITE(6,504)
      WRITE(6,505)
      WRITE(6,506)
      WRITE(6,507)
1     READ(5,100)IPRE,ISUC,AM,AN,CT,CS,CB,CE,NAME
100   FORMAT(2I5,6F4.0,3A6)
      IF(IPRE-00000)3,3,2
2     HMEN=SQRT(AM*CT/CE)
      HTIME=SQRT(AM*CE/CT)
      COST=AM*CT/HMEN+CB*AM+CE*HMEN+CS
      MMEN=HMEN+.5
      MTIME=HTIME+.5
      KOST=COST+.5
      KKOST=KKOST+KOST
      KCOST=KKOST
      KCOSE=KKOST
```

```
      MM=AM
      MN=AN
      MT=CT
      MS=CS
      MB=CB
      ME=CE
      WRITE(6,508)IPRE,ISUC,NAME,MM,MN,MT,MS,MB,ME,MMEN,MTIME,KOST
      WRITE(1,200)NAME,IPRE,ISUC,MTIME
      WRITE(4,300)IPRE,ISUC,MM,MN,MT,MS,MB,ME
300   FORMAT(2I4,6I4,2X)
200   FORMAT(3A6,12X,3I5,45X)
508   FORMAT(1H ,I5,1X,I5,1X,3A6,2X,7(I4,7X),I4,9X,I6,3X)
      GO TO 1
3     READ(5,400)ENDE
      WRITE(1,400)ENDE
      WRITE(10,400)ENDE
      WRITE(6,4001)KCOST
4001  FORMAT(1H0,29H THE OPTIMUM TOTAL COST IS $ ,I6)
400   FORMAT(A3,77X)
      END FILE 1
      REWIND 1
      END FILE 4
      REWIND 4
      END FILE 10
      REWIND 10
      RETURN
      END
$ORIGIN        CHAS
$IBFTC CPATH
      SUBROUTINE CRPATH
      DIMENSION ICP(100,50),ICOUNT(100),KOUNT(100),ICANT(100),KANT(100)
      DIMENSION NAME(4,300),NONE(300),NTWO(300),INT(300),ITF(300),
     1MM(300),MN(300),MT(300),MS(300),MB(300),ME(300),ICPATH(300)
      DIMENSION IA(100)
      DIMENSION NAM(4)
```

```
      COMMON/B/ICPATH,NAME,NONE,NTWO,INT,ITF,MM,MN,MT,MS,MB,ME
      COMMON/C/NAM
      COMMON/A/KTIME
      WRITE(6,100)
100   FORMAT(1H1)
      REWIND 4
      REWIND 7
      REWIND 8
      KK=0
      KL=0
      JA=0
      JB=0
      I=0
      J=1
      DO 1007 I=1,300
      NONE(I)=0
      NTWO(I)=0
1007  CONTINUE
      DO 1006 I=1,100
      IA(I)=0
      ICOUNT(I)=0
      KOUNT(I)=0
      ICANT(I)=0
      KANT(I)=0
1006  CONTINUE
      DO 1000 M=1,50
      DO 1001 N=1,100
      ICP(N,M)=0
1001  CONTINUE
1000  CONTINUE
C
C*** READ ONE DATA CARD AND COMPUTE DELTA K.
C
1     READ(8,101)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
101   FORMAT(A6,4A6,2I4,2I6,6I4)
```

```
      IF(IT)102,102,103
103   IF(EOF(8).NE.0.)GO TO 90
      GO TO 1
102   IF (IN-1)104,104,105
104   MBB=0
3     WRITE(7,106)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 103
105   AINT=IN
      AMM=MMM
      AMT=MTT
      AME=MEE
      DCOST=((AMM*AME)/(AINT*(AINT-1.0)))-AMT
C
C*** DELTA K IS STORED IN MBB
C
      MBB=DCOST+.5
      GO TO 3
90    END FILE 7
      REWIND 7
4     DO 1002 I=1,300
      READ(7,106)ICPATH(I),(NAME(J,I),J=1,4),NONE(I),NTWO(I),INT(I),
     2ITF(I),MM(I),MN(I),MT(I),MS(I),MB(I),ME(I)
      IF(EOF(7).NE.0.)GO TO 1003
1002  CONTINUE
1003  MZ=I
      I=0
      I=I+1
5     J=J+1
      IF(NTWO(I)-NONE(J))60,61,60
61    IF(NTWO(J)-999)63,62,62
C
C*** IF 62 IS NEXT- ONLY ONE CRITICAL PATH
C
63    I=J
      GO TO 5
```

```
60      I=0
        J=1
        K=0
6       I=I+1
        J=J+1
        K=K+1
        IA(I)=K
7       IF(NONE(I)-NONE(J))65,64,65
65      IF(NTWO(I).GE.999)GO TO 66
67      IF(NTWO(J).GE.999)GO TO 66
        J=J+1
        IF(J.GE.301)GO TO 750
        IF(NTWO(J).GE.999)GO TO 66
        GO TO 7
750     WRITE(6,751)
751     FORMAT(1H0,13HOUT OF ARRAY2)
        WRITE(6,7510)J
7510    FORMAT(1X,4HJ = ,I4)
        STOP
64      IA(J)=K
        K=K+1
        IF(NTWO(J).GE.999)GO TO 8880
68      I=J
        J=J+1
        IF(J.GE.301)GO TO 750
        GO TO 7
66      J=I
8880    K=K+1
        IA(I)=K
8       JA=JA+1
        KOUNT(JA)=K-1
        ICOUNT(JA)=NONE(J)
C
C*** ABOVE ROUTINE KEEPS TRACK OF BRANCHES
C
```

```
      K=0
      I=0
      J=1
9     IF(IA(J))69,69,70
69    I=J-1
      GO TO 6
70    IF(NTWO(J).GE.999)GO TO 711
72    J=J+1
      IF(J.GE.301)GO TO 750
      GO TO 9
711   J=J+1
      IF(NTWO(J).LE.0)GO TO 71
      GO TO 9
71    I=0
      M=0
      N=0
      J=1
      JA=0
      JC=0
      M=M+1
      N=N+1
      JC=JC+1
      I=I+1
      J=J+1
      JA=JA+1
10    IF(NONE(I)-ICOUNT(JA))20,21,20
20    JA=JA+1
      IF(JA-100)10,720,720
720   JA=0
      GO TO 20
21    JB=JA
      KANT(JB)=KOUNT(JA)
      ICANT(JB)=ICOUNT(JA)
      ICP(N,M)=NONE(I)
      KANT(JB)=KANT(JB)-1
```

```
      JA=JA+1
      N=N+1
      JB=JA
12    IF(NTWO(I)-NONE(J))73,14,73
73    IF(NTWO(I).GE.999)GO TO 730
      IF(J.GE.300)GO TO 733
7330  J=J+1
      GO TO 12
733   J=1
      GO TO 12
730   ICP(N,M)=NTWO(I)
C***************
C***************
      GO TO 78
14    ICP(N,M)=NONE(J)
      N=N+1
C*****************
C*****************
15    IF(NONE(J).EQ.ICOUNT(JA))GO TO 23
16    JA=JA+1
      IF(JA-100)15,15,24
24    JA=0
      GO TO 16
23    JB=JA
      KANT(JB)=KOUNT(JA)
      ICANT(JB)=ICOUNT(JA)
      KANT(JB)=KANT(JB)-1
      IF(KANT(JB))13,13,25
13    I=J
      GO TO 73
25    MRAY=M
      NRRY=N-1
      ID=M
17    M=M+1
      N=1
```

```
400   KI=0
      KJ=0
      IF(ICP(N,M).LE.0)GO TO 402
      M=M+1
      GO TO 400
402   ICP(N,M)=ICP(N,ID)
      N=N+1
      IF(NRRY.LE.N)GO TO 403
      GO TO 402
403   MY=M
      MYY=MRAY
      KA=MY-MYY
      IF(KANT(JB).LE.KA)GO TO 405
      GO TO 17
405   M=MRAY
      MR=MRAY
      N=NRRY
406   M=M-1
407   IF(ICP(N,M).EQ.ICP(N,MR))GO TO 408
      N=NRRY
      GO TO 879
411   I=1
      KJ=0
      M=MRAY
      N=NRRY
412   IF(ICP(N,M).EQ.NONE(I))GO TO 414
413   I=I+1
      GO TO 412
408   IF(N.LE.1)GO TO 409
      N=N-1
      GO TO 407
409   KI=KI+1
      N=NRRY
879   IF(M.LE.1)GO TO 411
      GO TO 406
```

```
414   KJ=KJ+1
      IF(KI.LT.KJ)GO TO 415
      GO TO 413
415   IF(NTWO(I).GE.999)GO TO 417
      KI=0
      J=I
      N=N+1
      JA=1
      KL=KL+1
      GO TO 12
417   N=N+1
      KI=0
      ICP(N,M)=NTWO(I)
428   M=1
      N=1
418   IF(ICP(N,M).LE.0)GO TO 420
      IF(ICP(N,M).GE.999)GO TO 421
      N=N+1
      GO TO 418
425   IF(ICP(N,M).LE.0)GO TO 2000
      GO TO 428
421   M=M+1
      N=1
      IF(M.GT.50)GO TO 2000
      GO TO 418
420   IF(N.LE.1)GO TO 78
      N=N-1
      IF(ICP(N,M).GE.999)GO TO 421
      MRAY=M
      NRRY=N
      MR=M
      GO TO 406
78    I=1
C************
C************
```

```
      KC=0
      JA=1
      JB=1
      KK=KK+1
42    IF(NONE(I)-ICOUNT(JA))43,44,43
43    JA=JA+1
      IF(JA.GE.101)GO TO 932
      GO TO 42
C************
800   DO 353 M=1,10
      WRITE(6,351)(ICP(N,M),N=1,10)
353   CONTINUE
351   FORMAT(1HO,10I4)
      STOP
C************
C***********
932   WRITE(6,212)
212   FORMAT(1HO,8HJA-LARGE)
      STOP
C***********
44    IF(NONE(I).EQ.ICANT(JB))GO TO 46
45    JB=JB+1
      IF(JB.GE.101)GO TO 931
      GO TO 44
C*****
931   WRITE(6,213)
213   FORMAT(1HO,8HJB-LARGE)
      STOP
C***********
46    IF(KANT(JB))425,425,47
47    N=1
C********
C**********
      KC=0
      M=M+1
```

```
888     IF(ICP(N,M).LE.0)GO TO 777
        M=M+1
        GO TO 888
777     ICP(N,M)=NONE(I)
        KB=KOUNT(JA)-KANT(JB)
        I=1
48      IF(NONE(I)-ICP(N,M))49,50,49
49      I=I+1
        IF(I.GE.100)GO TO 531
        GO TO 48
C*******************
531     WRITE(6,215)
215     FORMAT(1HO,9HLOOP-ON-I)
        GO TO 800
C*******************
50      KC=KC+1
        IF(KC-KB)49,49,51
51      N=N+1
        KANT(JB)=KANT(JB)-1
        JB=JB+1
        J=I
        GO TO 12
C**********
2000    REWIND 3
        I=0
        M=1
        N=1
        J=1
        ICPAT=J
509     WRITE(3,550)ICPAT
550     FORMAT(1X,I6,125X)
501     I=I+1
        IF(ICP(N,M).EQ.NONE(I))GO TO 505
        IF(I.LT.300)GO TO 501
        I=0
```

```
      GO TO 501
505   N=N+1
      IF(ICP(N,M).EQ.NTWO(I))GO TO 507
      N=N-1
      GO TO 501
507   ICPATH(I)=0
      WRITE(3,551)ICPATH(I),(NAME(L,I),L=1,4),NONE(I),NTWO(I),INT(I),
     1ITF(I),MM(I),MN(I),MT(I),MS(I),MB(I),ME(I)
551   FORMAT(1X,I6,4A6,2I4,2I6,6I4,57X)
      IF(NTWO(I).LT.999)GO TO 501
C******* 5555 INDICATE THE END OF THE PATH.
508   ICPAT=5555
      WRITE(3,550)ICPAT
      M=M+1
      N=1
      I=0
      IF(ICP(N,M).LE.0)GO TO 560
C****** 560 IS A PRINT ROUTINE.
      J=J+1
      ICPAT=J
      GO TO 509
C*********   FOLLOWING CODES ARE TEMPORARY
560   END FILE 3
      REWIND 3
      WRITE(6,100)
5600  KTIME=0
562   READ(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      WRITE(6,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      KTIME=KTIME+IN
      IF(EOF(3).NE.0.)GO TO 561
      IF(ICPAT.EQ.5555)GO TO 5600
      GO TO 562
561   I=0
      IPCOUN=0
      REWIND 3
```

```
      REWIND 7
      REWIND 8
      REWIND 9
      ITAPE=7777
      GO TO 949
900   I=LH
      WRITE(9,551)ICPATH(I),(NAME(L,I),L=1,4),NONE(I),NTWO(I),INT(I),
     1ITF(I),MM(I),MN(I),MT(I),MS(I),MB(I),ME(I)
      I=1
901   IF(ITAPE.EQ.7777)GO TO 902
903   IF(EOF(8).NE.0.)GO TO 930
      READ(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
904   IF(NON.GT.0)GO TO 908
      IF(ITAPE.EQ.7777) GO TO 907
      WRITE(3,550)ICPAT
      GO TO 901
902   IF(EOF(3).NE.0.)GO TO 930
      READ(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 904
907   WRITE(8,550)ICPAT
      GO TO 901
905   IF(ITAPE.EQ.7777)GO TO 9050
      WRITE(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 901
9050  WRITE(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 901
908   I=LH
      IF(ICPAT.EQ.9999)GO TO 914
      IF(ICPAT.EQ.5555)GO TO 924
      IF(ICPAT.EQ.8888)GO TO 905
      IF(NONE(I).NE.NON)GO TO 905
      IF(NTWO(I).NE.NTW)GO TO 905
913   ICPAT=9999
914   IF(ITAPE.EQ.7777)GO TO 915
      WRITE(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
```

```
917   IF(ITAPE.EQ.7777)GO TO 918
      IF(EOF(8).NE.0.)GO TO 930
      READ(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 923
915   WRITE(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 917
918   IF(EOF(3).NE.0.)GO TO 930
      READ(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
923   IF(ICPAT.NE.5555)GO TO 914
924   IF(ITAPE.EQ.7777)GO TO 925
      WRITE(3,550)ICPAT
927   IF(ITAPE.EQ.7777)GO TO 929
      IF(EOF(8).NE.0.)GO TO 930
      GO TO 903
929   IF(EOF(3).NE.0.)GO TO 930
      GO TO 902
925   WRITE(8,550)ICPAT
      GO TO 927
930   IF(ITAPE.EQ.7777)GO TO 933
      END FILE 3
      ITAPE=7777
      REWIND 3
      REWIND 8
935   I=1
      MK=0
      IF(ITAPE.EQ.7777)GO TO 938
      IF(EOF(8).NE.0.)GO TO 945
      READ(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
941   IF(NON.GT.0)GO TO 976
      IF(ITAPE.EQ.7777)GO TO 943
      WRITE(3,550)ICPAT
      GO TO 935
943   WRITE(8,550)ICPAT
      GO TO 935
933   END FILE 8
```

```
      ITAPE=6666
      REWIND 3
      REWIND 8
      GO TO 935
938   IF(EOF(3).NE.0.)GO TO 945
      READ(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 941
945   IF(ITAPE.EQ.7777)GO TO 947
      END FILE 3
      ITAPE=7777
      GO TO 948
947   END FILE 8
      ITAPE=6666
948   REWIND 3
      REWIND 8
      I=0
949   I=I+1
      KK=0
      IF(ITAPE.EQ.7777)GO TO 951
      READ(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
952   ICPATH(I)=ICPAT
      DO 9520 J=1,4
      NAME(J,I)=NAM(J)
9520  CONTINUE
      NONE(I)=NON
      NTWO(I)=NTW
      INT(I)=IN
      ITF(I)=IT
      MM(I)=MMM
      MN(I)=MNN
      MT(I)=MTT
      MS(I)=MSS
      MB(I)=MBB
      ME(I)=MEE
      GO TO 953
```

```
951   READ(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB.MEE
      GO TO 952
953   IF(ICPAT.NE.5555)GO TO 949
      I=1
      IH=0
      JH=0
      KH=0
      MH=0
955   I=I+1
      IF(ICPATH(I).EQ.9999)GO TO 956
      IF(ICPATH(I).EQ.8888)GO TO 962
      IF(ICPATH(I).EQ.5555)GO TO 968
      MH=MH+1
960   IF(MH.GT.1)GO TO 961
      GO TO 9600
964   IH=NONE(I)
      JH=NTWO(I)
      KH=MB(I)
      LH=I
      KK=1
962   IF(NTWO(I).LT.999)GO TO 955
      GO TO 968
961   IF(KH.LE.MB(I))GO TO 962
9600  IF(INT(I).GT.1)GO TO 964
      GO TO 962
956   KK=0
      IF(ITAPE.EQ.7777)GO TO 956
      IF(EOF(8).NE.0.)GO TO 973
967   I=0
      GO TO 949
966   IF(EOF(3).NE.0.)GO TO 973
      GO TO 967
968   IF(KH.LE.0)GO TO 9560
969   IF(ITAPE.EQ.7777)GO TO 971
      IF(EOF(8).NE.0.)GO TO 973
```

```
972    IPCOUN=IPCOUN+1
       GO TO 900
9560   IF(MH.LE.0)GO TO 956
       GO TO 969
971    IF(EOF(3).NE.0.)GO TO 973
       GO TO 972
973    IF(KK.LE.0)GO TO 974
       I=LH
       WRITE(9,551)ICPATH(I),(NAME(L,I),L=1,4),NONE(I),NTWO(I),INT(I),
      3ITF(I),MM(I),MN(I),MT(I),MS(I),MB(I),ME(I)
974    END FILE 9
       REWIND 9
       WRITE(6,100)
       WRITE(6,9990)
9990   FORMAT(1HO,47H THE ACTIVITY(S) TO BE COMPRESSED AT THIS STAGE///)
       WRITE(6,9991)
9991   FORMAT(3X,4HNODE,8X,13HACTIVITY NAME,8X,8HDURATION,4X,
      114HCHANGE IN COST,68X)
       WRITE(6,9992)
9992   FORMAT(1X,3HPRE,2X,3HSUC,120X///)
999    READ(9,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
       WRITE(6,5510)NON,NTW,NAM,IN,MBB
5510   FORMAT(1X,I4,1X,I4,3X,4A6,2X,I4,10X,I4)
       IF(EOF(9).NE.0.)GO TO 996
       GO TO 999
996    WRITE(6,9960)
9960   FORMAT(1HO,1H ///,40H COMPRESSION GIVES THE FOLLOWING RESULTS///)
       RETURN
976    I=I+1
       IF(ICPATH(I).EQ.5555)GO TO 9350
       IF(ICPATH(I).EQ.8888)GO TO 976
       IF(ICPAT.EQ.9999)GO TO 9920
       IF(ICPAT.EQ.8888)GO TO 9820
       IF(ICPAT.EQ.5555)GO TO 982
       MK=MK+1
```

```
      IF(NONE(I).NE.NON)GO TO 976
      IF(NTWO(I).NE.NTW)GO TO 976
      GO TO 985
9920  IF(ITAPE.EQ.7777)GO TO 9921
      WRITE(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 992
9921  WRITE(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 992
9820  IF(ITAPE.EQ.7777)GO TO 9821
      WRITE(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 935
9821  WRITE(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 935
9350  IF(MK.LE.0)GO TO 935
      IF(ITAPE.EQ.7777)GO TO 9351
      WRITE(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 935
9351  WRITE(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 935
982   IF(ITAPE.EQ.7777)GO TO 989
      WRITE(3,550)ICPAT
      GO TO 935
989   WRITE(8,550)ICPAT
      GO TO 935
985   ICPAT=8888
      IF(ITAPE.EQ.7777)GO TO 986
      WRITE(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 935
986   WRITE(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 935
992   IF(ITAPE.EQ.7777)GO TO 993
      READ(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      IF(ICPAT.EQ.5555)GO TO 982
      WRITE(3,551)ICPAT,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 992
```

```
993   READ(3,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      IF(ICPAT.EQ.5555)GO TO 982
      WRITE(8,551)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      GO TO 992
      WRITE(6,217)
217   FORMAT(1H0,11HFINISH-HERE)
      GO TO 800
C************
      N=1
87    IF(ICP(N,M))85,85,86
86    M=M+1
      GO TO 87
85    NHOLD=M
      DO 2002 M=1,NHOLD
      WRITE(6,2003)(ICP(N,M),N=1,100)
2002  CONTINUE
      STOP
C
C*** PATHS ARE ON OUTPUT TAPE
C
2003  FORMAT(1H0,10X,I4)
62    REWIND 3
      WRITE(3,1060)
1060  FORMAT(1X,6H     1,125X)
      DO 2004 I=1,MZ
      ICPATH(I)=0
      WRITE(6,106)ICPATH(I),(NAME(J,I),J=1,4),NONE(I),NTWO(I),INT(I),
     3ITF(I),MM(I),MN(I),MT(I),MS(I),MB(I),ME(I)
      WRITE(3,106)ICPATH(I),(NAME(J,I),J=1,4),NONE(I),NTWO(I),INT(I),
     4ITF(I),MM(I),MN(I),MT(I),MS(I),MB(I),ME(I)
2004  CONTINUE
211   FORMAT(1H0,10X,11HERROR-CHECK)
106   FORMAT(1X,A6,4A6,2I4,2I6,6I4,57X)
      WRITE(3,1061)
1061  FORMAT(1X,6H  5555,125X)
```

```
      END FILE 9
      REWIND 9
      GO TO 560
      END
$ORIGIN        CHAS
$IBFTC MERG
      SUBROUTINE MERGE
      DIMENSION FIRST(20)
      DIMENSION ICPATH(300),NAME(4,300),NONE(300),NTWO(300),INT(300),
     1ITF(300),MM(300),MN(300),MT(300),MS(300),MB(300),ME(300)
      COMMON/B/ICPATH,NAME,NONE,NTWO,INT,ITF,MM,MN,MT,MS,MB,ME
      COMMON/BB/KKOST,KCOST,KONS
      COMMON/C/NAM(4)
      REWIND 9
      REWIND 3
      REWIND 10
      REWIND 1
      REWIND 4
101   READ(9,100)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
      KCOST=KCOST+MBB
      IF(EOF(9).NE.0.)GO TO 102
      GO TO 101
102   REWIND 9
103   FORMAT(4A6,6X,3I5)
100   FORMAT(1X,I6,4A6,2I4,2I6,6I4,57X)
      DO 104 I=1,300
      NTWO(I)=0
104   CONTINUE
      DO 105 I=1,300
      READ(1,103)(NAME(J,I),J=1,4),NONE(I),NTWO(I),INT(I)
      IF(EOF(1).NE.0.)GO TO 106
105   CONTINUE
106   M=I
      I=4
108   READ(9,100)ICPAT,NAM,NON,NTW,IN,IT,MMM,MNN,MTT,MSS,MBB,MEE
```

```
110   IF(NONE(I).NE.NON)GO TO 107
      IF(NTWO(I).NE.NTW)GO TO 107
      INT(I)=INT(I)-1
      IF(EOF(9).NE.0.)GO TO 109
      GO TO 108
107   IF(NTWO(I).GE.999)GO TO 111
      I=I+1
      GO TO 110
111   I=4
      GO TO 110
109   REWIND 9
      REWIND 1
      READ(10,1000)FIRST
      WRITE(1,1000)FIRST
      READ(10,1000)FIRST
      WRITE(1,1000)FIRST
      READ(10,1000)FIRST
      WRITE(1,1000)FIRST
1000  FORMAT(20A4)
10    FORMAT(1H1)
113   DO 114 I=4,M
      WRITE(1,103)(NAME(J,I),J=1,4),NONE(I),NTWO(I),INT(I)
114   CONTINUE
      END FILE 1
      REWIND 1
      REWIND 10
      REWIND 3
      REWIND 9
      KONS=KONS+1
      RETURN
      END
$DATA
```

APPENDIX III

A SAMPLE PROBLEM

A sample problem is shown here to illustrate the type of output generated by the computer program that utilizes the time compression algorithm.

Page 91 is a listing of the data input. This list also includes the computed values of the optimum duration, optimum number of men, and the optimum cost for each activity. The last line on this page shows the optimum total cost of the project.

Page 92 through page 95 illustrate the critical path analysis and time compression results. It must be noted here that the final stage of the time compression routine is 5 time units below the optimum duration of the project.

In regard to the listing of the critical path results, those jobs that are marked with asterisks are the critical jobs in the network. It must be noted here that the computer output only show the first stage and the last stage of the time compression results.

ORLANDO S MADRIGAL
DEPARTMENT OF INDUSTRIAL ENGINEERING
TEXAS A+M UNIVERSITY
1966

**** PROGRAM INPUT ****

NODE PRE	NODE SUC	ACTIVITY NAME	MAN HOURS	NUMBER OF MEN	TIME COST (VARIABLE) DOLLARS	FIXED COST (INITIAL) DOLLARS	MANPOWER COST DOLLARS	LOAD COST DOLLARS	NO. MEN (OPTIMUM)	DURATION (OPTIMUM)	TOTAL COST (DOLLARS)
1	2	LEAD TIME	11	7	7	8	4	9	3	4	105
1	5	LEAD TIME	16	9	9	9	9	9	4	4	225
2	3	MEASURE AND SKETCH	16	1	7	7	7	7	4	4	175
3	4	DEVELOP MAT LIST	12	12	12	12	12	12	3	3	239
4	6	PROCURE PIPE	13	12	11	10	9	8	4	3	195
4	8	PROCURE VALVES	10	10	10	10	10	10	3	3	173
4	5	ERECT SCAFFOLD	10	2	11	12	2	7	4	3	87
4	7	DEACTIVATE LINES	16	9	9	9	9	9	4	4	225
5	7	PEFABRICATE SECT	14	8	8	8	8	8	4	4	180
6	9	REMOVE OLD PIPE	13	7	7	7	7	7	4	4	148
7	8	PLACE NEW PIPE	11	11	11	11	11	11	3	3	205
7	9	PLACE NEW VALVES	13	9	8	9	8	9	3	4	174
9	10	WELD PIPE	13	9	8	9	8	9	3	4	174
8	11	FIT UP PIPE SEC	14	8	9	8	9	8	4	4	197
10	13	INSULATE	16	9	9	9	9	9	4	4	225
11	12	PRESSURE TEST	11	7	7	8	4	9	3	4	105
12	14	REPAINT	16	9	9	9	9	9	4	4	225
13	999	REMOVE SCAFFOLD	14	8	8	8	8	8	4	4	180
14	999	CLEAN UP	13	12	11	10	9	8	4	3	195

THE OPTIMUM TOTAL COST IS $ 3432

COMPLETE LIST OF ALL JOBS BY PRIORITY LEVEL

	LEVEL OF PRIORITY	JOB NAME	SECTION	SPAN	DURATION	START EARLY	START LATE	FINISH EARLY	FINISH LATE	FLOAT TOTAL	FLOAT FREE
****	1	LEAD TIME	PIPE REPLACEMENT JOB	1 TO 2	4	0	0	4	4	0	0
	1	LEAD TIME	PIPE REPLACEMENT JOB	1 TO 5	4	0	10	4	14	10	10
****	2	MEASURE AND SKETCH	PIPE REPLACEMENT JOB	2 TO 3	4	4	4	8	8	0	0
****	3	DEVELOP MAT LIST	PIPE REPLACEMENT JOB	3 TO 4	3	8	8	11	11	0	0
	4	PROCURE PIPE	PIPE REPLACEMENT JOB	4 TO 6	3	11	17	14	20	6	0
	4	PROCURE VALVES	PIPE REPLACEMENT JOB	4 TO 8	3	11	18	14	21	7	7
****	4	ERECT SCAFFOLD	PIPE REPLACEMENT JOB	4 TO 5	3	11	11	14	14	0	0
	4	DEACTIVATE LINES	PIPE REPLACEMENT JOB	4 TO 7	4	11	14	15	18	3	3
****	5	PEFABRICATE SECT	PIPE REPLACEMENT JOB	5 TO 7	4	14	14	18	18	0	0
	5	REMOVE OLD PIPE	PIPE REPLACEMENT JOB	6 TO 9	4	14	20	18	24	6	4
****	6	PLACE NEW PIPE	PIPE REPLACEMENT JOB	7 TO 8	3	18	18	21	21	0	0
	6	PLACE NEW VALVES	PIPE REPLACEMENT JOB	7 TO 9	4	18	20	22	24	2	0
	7	WELD PIPE	PIPE REPLACEMENT JOB	9 TO 10	4	22	24	26	28	2	0
****	7	FIT UP PIPE SEC	PIPE REPLACEMENT JOB	8 TO 11	4	21	21	25	25	0	0
	8	INSULATE	PIPE REPLACEMENT JOB	10 TO 13	4	26	28	30	32	2	0
****	8	PRESSURE TEST	PIPE REPLACEMENT JOB	11 TO 12	4	25	25	29	29	0	0
****	9	REPAINT	PIPE REPLACEMENT JOB	12 TO 14	4	29	29	33	33	0	0
	9	REMOVE SCAFFOLD	PIPE REPLACEMENT JOB	13 TO 999	4	30	32	34	36	2	2
****	10	CLEAN UP	PIPE REPLACEMENT JOB	14 TO 999	3	33	33	36	36	0	0

THE ACTIVITY(S) TO BE COMPRESSED AT THIS STAGE

NODE PRE	SUC	ACTIVITY NAME	DURATION	CHANGE IN COST
8	11	FIT UP PIPE SEC	4	0

COMPRESSION GIVES THE FOLLOWING RESULTS

THE OPTIMUM TOTAL COST IS $ 3432

THE TOTAL COST AT THIS STAGE IS $ 3432

THE COST CONSTRAINT IS $ 3437

THE DURATION AT THIS STAGE IS 35

THIS DURATION IS 1 TIME UNITS BELOW THE OPTIMUM TIME

THE ACTIVITY(S) TO BE COMPRESSED AT THIS STAGE

NODE PRE	SUC	ACTIVITY NAME	DURATION	CHANGE IN COST
11	12	PRESSURE TEST	4	1

COMPRESSION GIVES THE FOLLOWING RESULTS

THE OPTIMUM TOTAL COST IS $ 3432

THE TOTAL COST AT THIS STAGE IS $ 3436

THE COST CONSTRAINT IS $ 3437

THE DURATION AT THIS STAGE IS 31

THIS DURATION IS 5 TIME UNITS BELOW THE OPTIMUM TIME

COMPLETE LIST OF ALL JOBS BY PRIORITY LEVEL

LEVEL OF PRIORITY		JOB NAME	SECTION	SPAN	DURATION	START EARLY	START LATE	FINISH EARLY	FINISH LATE	FLOAT TOTAL	FLOAT FREE
****	1	LEAD TIME	PIPE REPLACEMENT JOB	1 TO 2	3	0	0	3	3	0	0
	1	LEAD TIME	PIPE REPLACEMENT JOB	1 TO 5	4	0	8	4	12	8	8
****	2	MEASURE AND SKETCH	PIPE REPLACEMENT JOB	2 TO 3	4	3	3	7	7	0	0
****	3	DEVELOP MAT LIST	PIPE REPLACEMENT JOB	3 TO 4	3	7	7	10	10	0	0
	4	PROCURE PIPE	PIPE REPLACEMENT JOB	4 TO 6	3	10	12	13	15	2	0
	4	PROCURE VALVES	PIPE REPLACEMENT JOB	4 TO 8	3	10	15	13	18	5	5
****	4	ERECT SCAFFOLD	PIPE REPLACEMENT JOB	4 TO 5	2	10	10	12	12	0	0
	4	DEACTIVATE LINES	PIPE REPLACEMENT JOB	4 TO 7	4	10	11	14	15	1	1
****	5	PEFABRICATE SECT	PIPE REPLACEMENT JOB	5 TO 7	3	12	12	15	15	0	0
	5	REMOVE OLD PIPE	PIPE REPLACEMENT JOB	6 TO 9	4	13	15	17	19	2	2
****	6	PLACE NEW PIPE	PIPE REPLACEMENT JOB	7 TO 8	3	15	15	18	18	0	0
****	6	PLACE NEW VALVES	PIPE REPLACEMENT JOB	7 TO 9	4	15	15	19	19	0	0
****	7	WELD PIPE	PIPE REPLACEMENT JOB	9 TO 10	4	19	19	23	23	0	0
****	7	FIT UP PIPE SEC	PIPE REPLACEMENT JOB	8 TO 11	3	18	18	21	21	0	0
****	8	INSULATE	PIPE REPLACEMENT JOB	10 TO 13	4	23	23	27	27	0	0
****	8	PRESSURE TEST	PIPE REPLACEMENT JOB	11 TO 12	3	21	21	24	24	0	0
****	9	REPAINT	PIPE REPLACEMENT JOB	12 TO 14	4	24	24	28	28	0	0
****	9	REMOVE SCAFFOLD	PIPE REPLACEMENT JOB	13 TO 999	4	27	27	31	31	0	0
****	10	CLEAN UP	PIPE REPLACEMENT JOB	14 TO 999	3	28	28	31	31	0	0

REFERENCES

1. Moshman, J., J. Johnson, and M. Larsen. 1963. *RAMPS - A Technique for Resource Allocation and Multi - Project Scheduling*. C-E-I-R, Inc., 1200 Jefferson Davis Highway, Arlington 2, Virginia.

2. Wiest, J. D. 1963. *The Scheduling of Large Projects with Limited Resources*. Doctoral Dissertation, Research Memorandum No. 113, Graduate School of Industrial Administration, Carnegie Institute of Technology, Pittsburgh.

3. Lambourn, S. 1963. "Resource Allocation and Multi - Project Scheduling (RAMPS) - A New Tool in Planning and Control." *The Computer Journal*. Vol. V (4); pp. 300-304.

4. Martino, R. L. 1964. "Project Management and Control." *The American Management Association*. New York, N. Y.

5. Moder, J. J. and C. R. Phillips. 1964. *Project Management and Control with CPM and PERT*. Reinhold Publishing Company, New York, N. Y.

6. Evarts, H. F. 1964. *Introduction to PERT*. Allyn & Bacon, Inc. Boston, Mass.

7. *Bibliography PERT and Other Management Systems and Techniques*. 1963. PERT Orientation and Training Center, Bolling Air Force Base, Washington, 25, D. C.

8. Kelly, J. E. and M. R. Walker. 1959. "Critical Path Planning and Scheduling." *Proceedings of the 1959 EJCC*, National Joint Computer Committee; pp. 160-173.

9. *IBM 7090/7094 IBSYS Operating System Version 13 FORTRAN IV Language*. International Bysiness Machines Corporation, Data Processing Division, White Plains,New York, 1965.