UNITED ARAB REPUBLIC

THE INSTITUTE OF NATIONAL PLANNING

Memo. No. 544

PROGRAM EVALUATION AND REVIEW TECHNIQUE (PERT) WITH COMPUTER APPLICATION

I. PERT / TIME

by

Dr. Hamdy A. Taha

Eng. Mohamed A. Mongy

Operation Research Center.

February 1965

CAIRO

PROGRAM EVALUATION AND REVIEW TECHNIQUE (PERT) WITH COMPUTER APPLICATION

I. PERT / TIME

Contents

Introduction:

Program Evaluation and Review Technique (PERT) is a modern planning Tool which was developed by the Americans to help managers better control their projects. It gives management the clue as to what should be done in order to accomplish program objectives on time. It pinpoints the areas of the project that require attention so that management can prepare for it in advance. It also indicates the project requirements of man, equipment and capital and gives management a broader view of what should be expected from the project. The main factor in carrying out PERT computations is the time factor which is used as the common denominator to reflect planned resource application and performance specifications.

Arrow Diagram:

The first step in constructing a PERT model is to draw the arrow diagram of the project. This is affected by dividing the project into a definite number of jobs. Each job consists of an activity and two end events and requires a certain time to complete it. This time can be estimated in advance. In terms of the arrow diagram, an activity is represented by an arrow whose two end points represent the events of the activity. These events represent points in time which indicate the termination of one or more activities and the beginning of new ones.

Figure (1) shows an activity A with two end events i, and j respectively. It means that activity A can't start until event i is completed while event j indicates the completion of A. The arrow indicates that the direction of progress goes from event i to event j. It should be noted that the length

Figure (1)

of the arrow may not be drawn to scale to represent the actual time of the activity.

With the above concept in mind, the whole project can be divided into a number of activities and events which will then give an arrow network showing the precedence relationship between the different activities. In constructing such a network, it will help, at each event, to ask the following questions in order to ensure the proper precedency relationship:-

1. What immediately precedes this activity?
2. What immediately follows this activity?
3. What can be accomplished concurrently?

Example:

Consider a project consisting of the following activities:

Activity	Time units
A	10
B	5
C	4
D	3
E	1

The network defining this project is given in Figure (2).

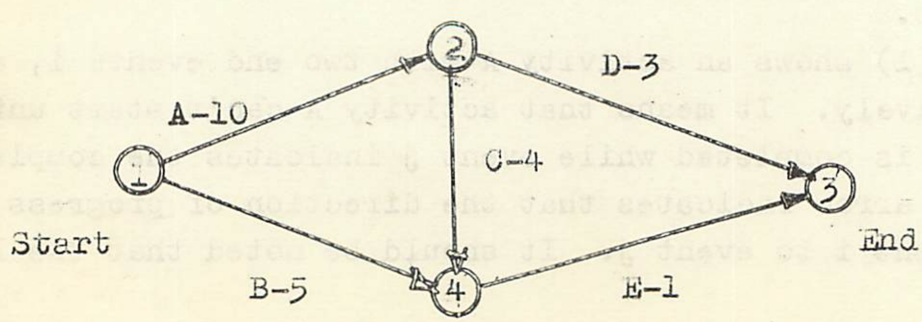

Figure (2)

This diagram shows that activities C ad D can't start until activity A is completed and that activity E can't start until activities C and B are completed. Activity D can start immediately after activity A is completed. Note that the numbering of the events is such that the number of tail event is <u>less</u> than the number of the head event for <u>any</u> activity. This is mandatory especially when computers are used to solve the problem.

<u>Dummy Activities</u>:

It is necessary for the arrow diagram to be correct that each activity be identified by two end events uniquely. This means that if two activities are started and are completed together, then the arrow diagram must still identify each uniquely. This is usually accomplished by the use of a dummy activity which is represented by a dotted line with the time associated with it equal to zero. Figure (3) shows the case where the two activities A and B having the same start and end can be represented in the arrow diagram. In this case activity

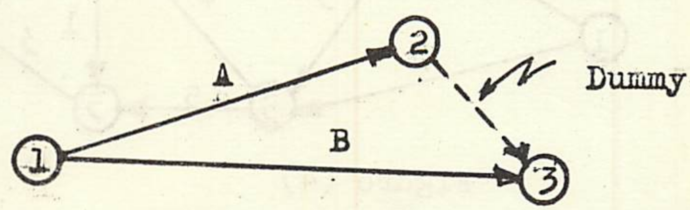

Figure (3)

A is defined by events (1,2) while activity B is defined by events (1,3). The dummy activity (2,3) is introduced, at zero time, only for convenience.

<u>Critical Path Concept</u>:

After the arrow diagram is completed the next step is to compute the longest path in the network. This longest path is known as the <u>critical path</u>.

The jobs lying on the critical path (known as critical jobs) are the jobs that should be completed in time if the project is to be completed as scheduled. Other jobs will have free slack time which allows the delay in their start within limits to be determined from the computations of the network.

The critical path indicates the jobs that require special attention since, as mentioned before they must be completed on time.

One method for determining the critical path consists of establishing the earliest and latest times that each event can occur and then applying simple tests to see if an activity lies on the critical path.

Determination of Event Occurrence Time

A project consisting of ten activities (including lead time) is shown below. Included is the duration for each activity.

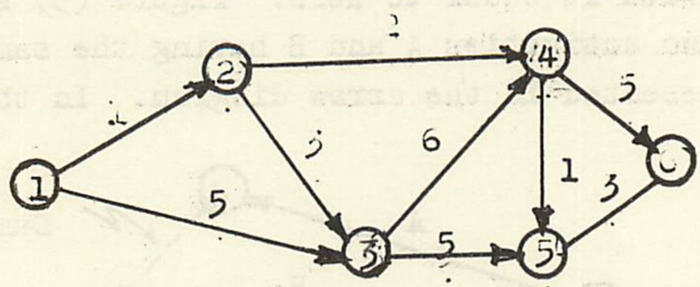

Figure (4)

To determine the earliest possible occurrence of each event, work from project start to completion, considering each event in turn, and comparing parallel paths to the event. For example:

1. Activity (0,1) is a lead time activity and for this network is assumed to be zero. (not indicated for simplicity)

2. The time from event ① to event ② is one day. The only path from ① to ② is that one activity. Thus, the

earliest that event (2) can occur is one day after project start.

3. The time from (1) to (3) is five days. Also, (3) may not occur until activities (1,2) and (2,3) which require four days, are completed. The earliest that (3) can occur, then, is five days.

4. The time from (2) to (4) is two days. However, (4) may not occur until activity (3,4) which requires six days, is completed. Thus, the earliest that event (4) can occur is eleven days (5 + 6) after project start.

5. The time from (3) to (5) is five days. However, (5) may not occur until activity (4,5), which takes one day, is completed. Thus, the earliest that event (5) can occur is twelve days (11 + 1) after project start.

6. The time from (5) to (6) is three days. However, (6) may not occur until activity (4,6) is completed. Thus, the earliest that event (6) can occur is sixteen days (11 + 5) after project start.

These earliest possible event occurrences are shown in the diagram below in the squares beside each event.

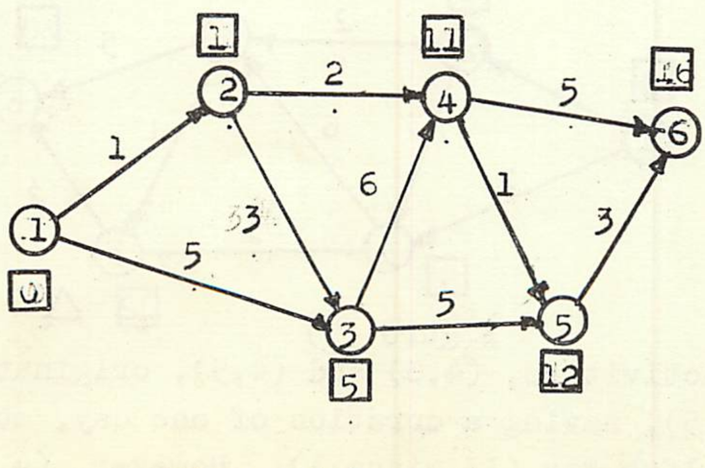

Figure (5)

The latest permissible completion for each activity is determined by finding the latest occurrence for each event, starting with project completion and working back to project start. The latest occurrence for each event is found by comparing the start time requirements for all activities starting at that event. Latest event occurrence times are shown on the network model in a triangle.

For example, continuing the above project:

1. The latest occurrence for event ⑥ is predetermined to be sixteen days.

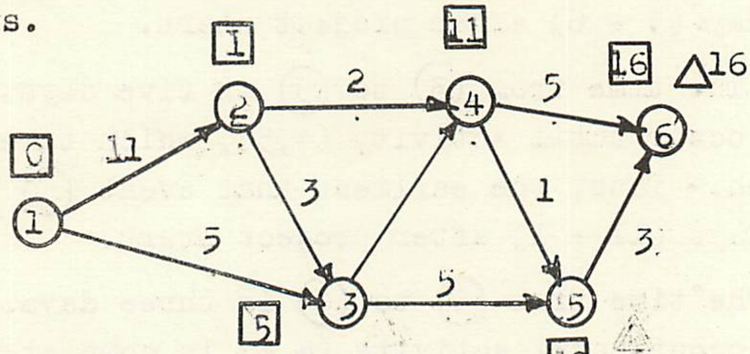

Figure (6)

2. The only activity dependent upon completion of event ⑤ is (5,6) with a duration of three days. Thus, the latest occurrence for event ⑤ is thirteen days (16 minus 3) after project start.

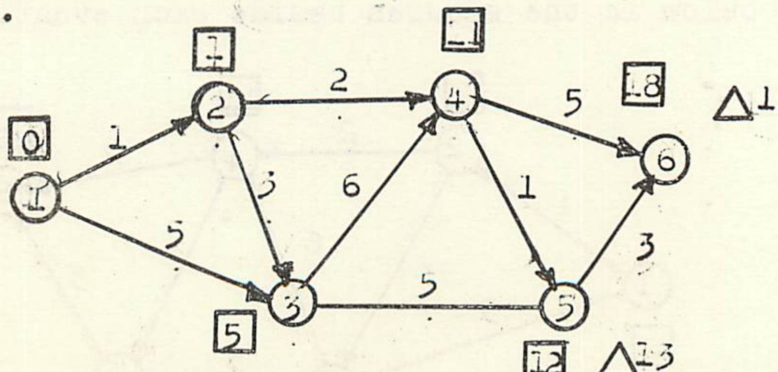

Figure (7)

3. Two activities, (4,6) and (4,5), originate at event ④ . Activity (4,5), having a duration of one day, must start no later than the twelfth day (13 minus 1). However, (4,6), with a duration of five days, must start no later than the eleventh day

(16 minus 5). Thus, the latest permissible occurrence for event (4) is the eleventh day.

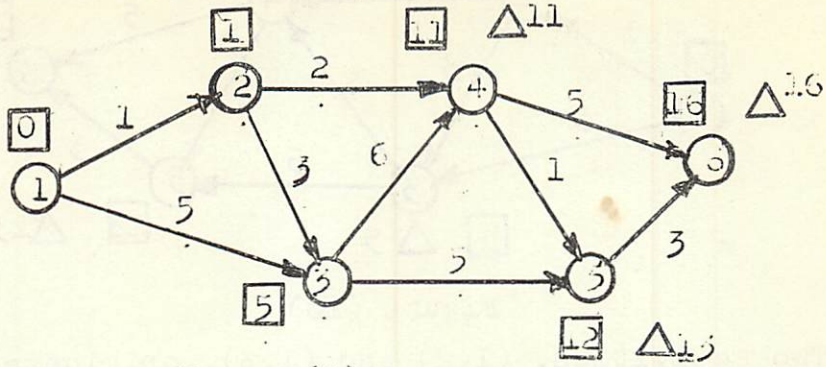

Figure (8)

4. Two activities, (3,4) and (3,5), originate at event (3). Activity (3,5), having a duration of five days, must start no later than the eighth day (13 minus 5). However, (3,4), with a duration of six days, must start no later than the fifth day (11 minus 6). Thus, the latest occurrence for event (3) is the fifth day.

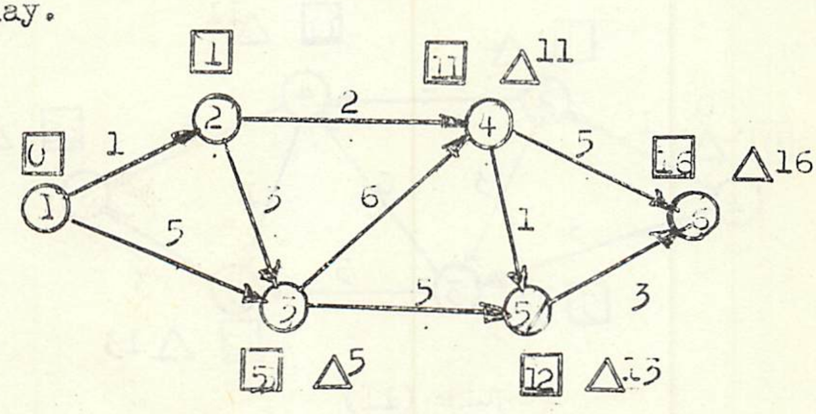

Figure (9)

5. Two activities, (2,4) and (2,3), originate at event (2). Activity (2,4), with a duration of two days, must start no later than the ninth day (11 minus 2); activity (2,3), with a duration of three days, must start no later than the second day (5 minus 3). Thus, the latest permissible occurrence for event (2) is the second day.

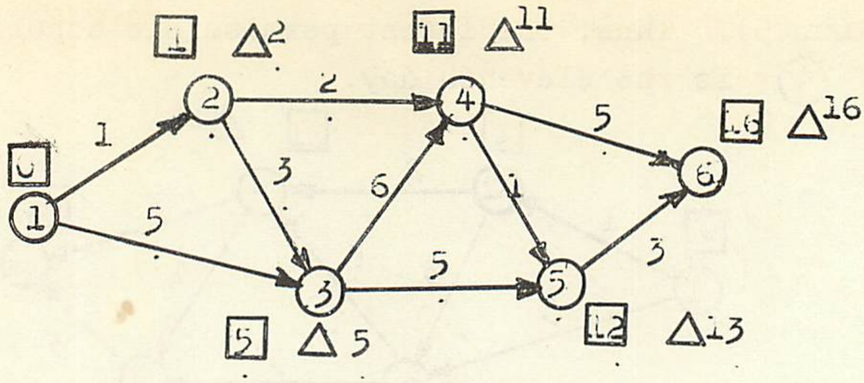

Figure (10)

6. Two activities, (1,2) and (1,3), originate in event ① . Activity (1,2), with a duration of one day, must start no later than the end of the first day (2 minus 1); activity (1,3), with a duration of five days, must start no later than the beginning of the first day (5 minus 5). Thus, the latest permissible occurrence for event ① is the start of the first day (or day 0).

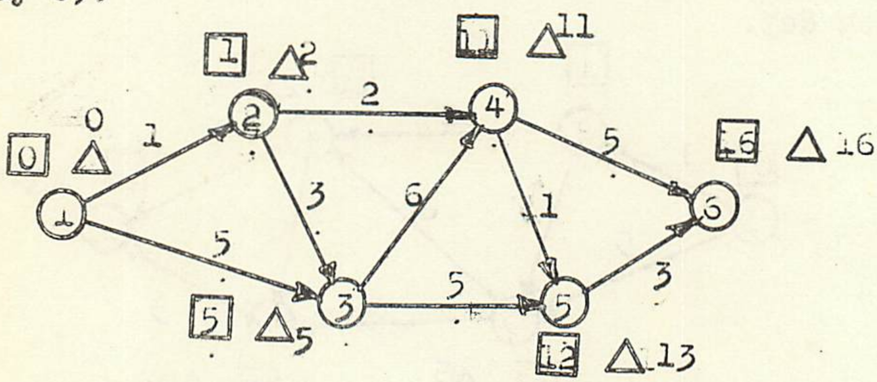

Figure (11)

Determining the Critical Path

Once the earliest possible and latest permissible occurrence times are found, activities lying on the critical path can be readily determined by two simple tests.

1. If the earliest occurrence time (□) equals the latest occurrence time (△) at the head of the arrow and the earliest occurrence time (□) equals the latest occurrence time (△) at the tail of the arrow, the activity possibly lies on the critical path. This is a necessary condition for a critical activity.

2. If the above condition is met, <u>and</u> the difference between the time at the head of the arrow and the time at the tail of the arrow equals the activity duration, then that activity is critical and lies on the critical path. This is a sufficient condition for a critical activity.

Using these criteria, it is now possible to determine the critical path for the sample project. Activities may be tested in any convenient sequence.

1. Activity (1,2) does not satisfy the first condition because, at the head of the arrow, the earliest time does not equal the latest time, [1] ≠ △2 .

2. Activity (1,3) satisfies the first condition, [0] = △0 and [5] = △5 , and the second condition, [5] - △0 = 5, (the activity duration). Thus, (1,3) lies on the critical path.

3. Activity (2,4) does not satisfy the first condition because, at the tail of the arrow, the earliest time does not equal the latest completion time, [1] ≠ △2 .

4. Activity (2,3) is excluded for the same reason as in 3, above.

5. Activity (3,4) satisfies the first condition, [5] = △5 and [11] = △11 , and the second, [11] - [5] = 6 (the activity duration). Thus, (3,4) lies on the critical path.

6. Activity (3,5) cannot meet the first test because, at the head of the arrow, the earliest time does not equal the latest time, [12] ≠ △13 .

7. Activity (4,5) is excluded for the same reason as in 6, above.

8. Activity (4,6) meets the first test, [11] = △11 and [16] = △16 , and the second test; [16] - △11 = 5 (the activity duration). Thus, (4,6) lies on the critical path.

9. Activity (5,6) cannot meet the first test because, at the tail of the arrow, the earliest time does not equal the latest time, [12] ≠ △13 .

With the critical activities defined, the critical path for the project may be depicted, using heavier lines for the critical activities:

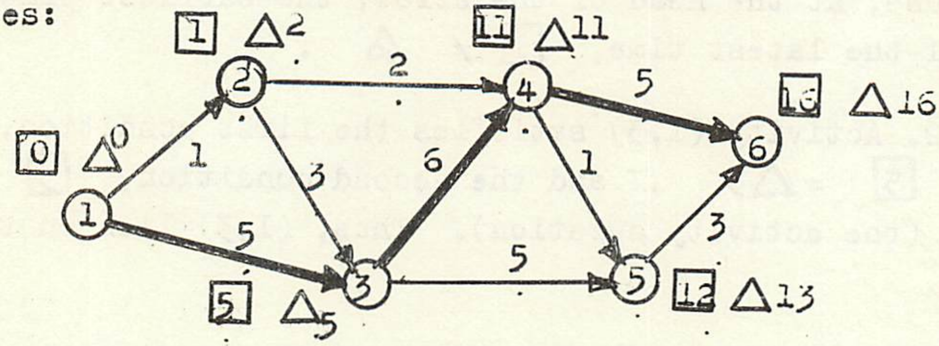

Figure (12)

The sample project has only one critical path; actual projects may have several.

The arrow diagram above shows the critical path with a fixed set of parameters, that is, the activity durations are considered to be unchanging. If one or more durations were to change, the critical path could change.

FLOAT :

Critical activities have been defined as those which, if delayed, will cause a corresponding delay in project completion. Conversely, non-critical activities may experience limited amounts of delay or acceleration in start and completion time without affecting project completion. The amount of time latitude permissible is referred to as float. If float time limits are exceeded, some non-critical activities can immediately become critical. For analysis purposes there are three kinds of float.

Total Float - is the maximum time available to complete an activity less the duration of that activity.

Free Float - is the time available to complete an activity assuming all activities in a project start, and are completed, as early as possible.

Scheduled Float - is the weighted float available to an activity. If used it will not affect any other activity and will permit the project to be completed on schedule.

Each of these concepts has particular significance in the control of projects. Total float provides an upper limit which cannot be exceeded if the project is to be completed on schedule.

Free float has the effect of pushing all available float as far into the future as possible. The logic behind this concept is that unavoidable delays undoubtedly occur in any long term project and it would be desirable to have float allocated to the future to counteract these delays.

Scheduled float is determined by allocating the float available in the network to the non-critical activities based on a priority weighing factor determined by the user. Thus, if an activity is non-critical, it will have more float assigned to it for each increase in its priority weighting factor. Priority weighting factors can vary from 1 to 9.

Use of the Information Obtained from Critical Path Computations:

After the previous computations are performed on the network, one can use the obtained information to set up the time schedule of the project. This is usually in done by the use of a "bar chart" where the lengths represent the actual time of the activities (to scale). Example of this bar chart is given in Figure (13)

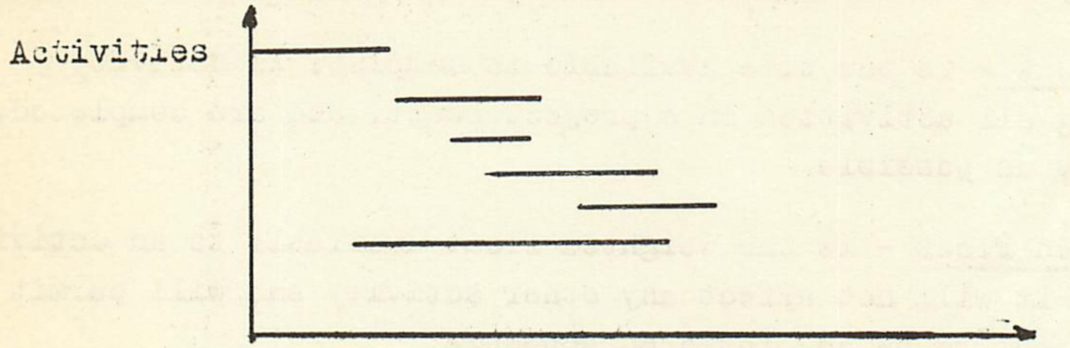

Figure (13)

The critical jobs will be shown on the bar Chart tight on schedule while the non-critical jobs have some slack so that they can be completed any time between the earliest start and the latest completion of the job.

Input data:

The input of the program are ordered as follows:

1. Number of activities (FORMAT I4).
2. events i and j and duration for each activity (FORMAT 2 I4, F4.2).

The card number 2 should be repeated with one card for each activity.

It should be noted that for any activity it is necessary that $i < j$.

Flow Diagram Pert-Time Problem:

A Fortran program is given for computing the critical path of a given network (see page No. 16). The flow diagram is shown in Figure (14)

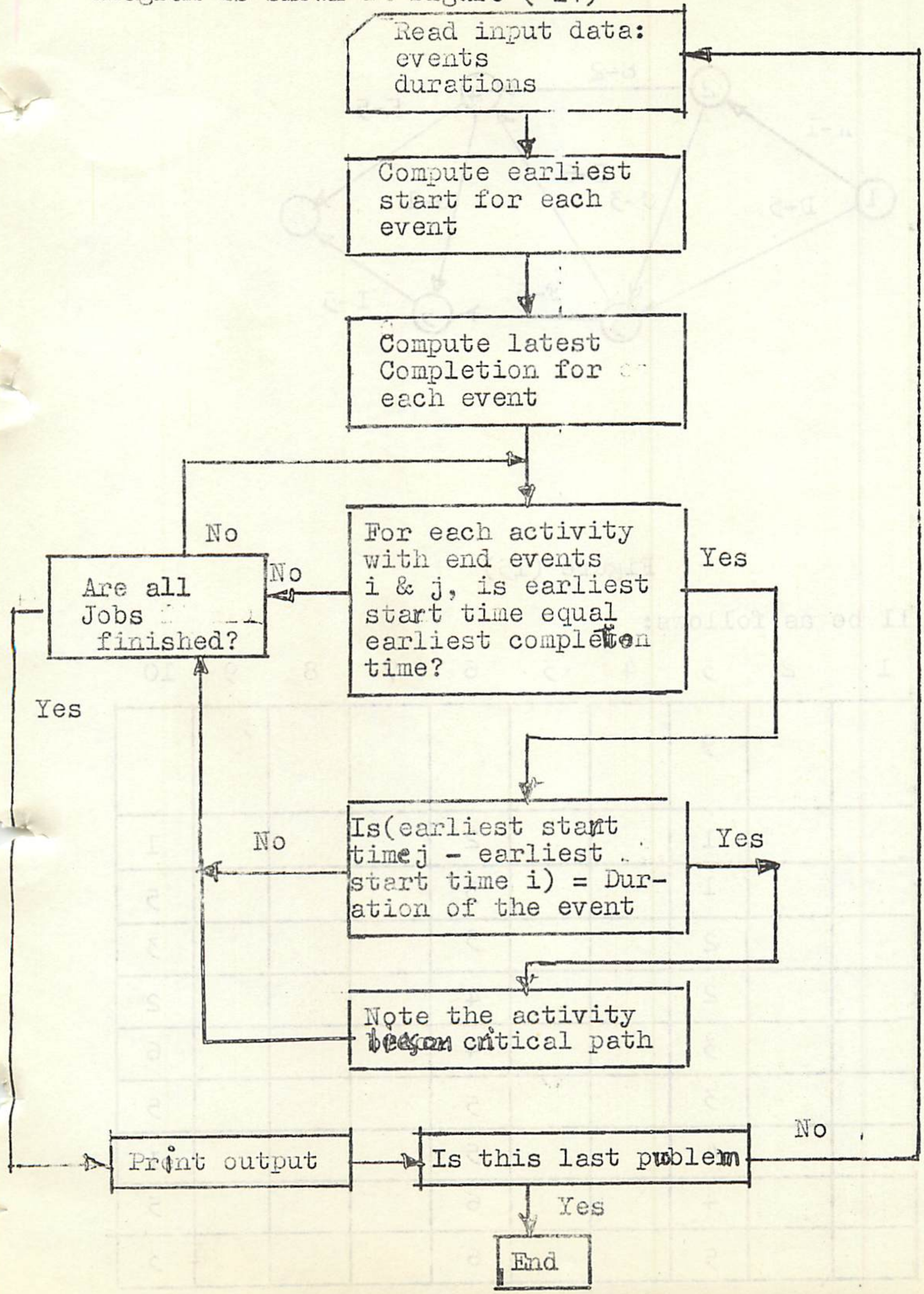

Figure (14)

Example

Refering to the above example:

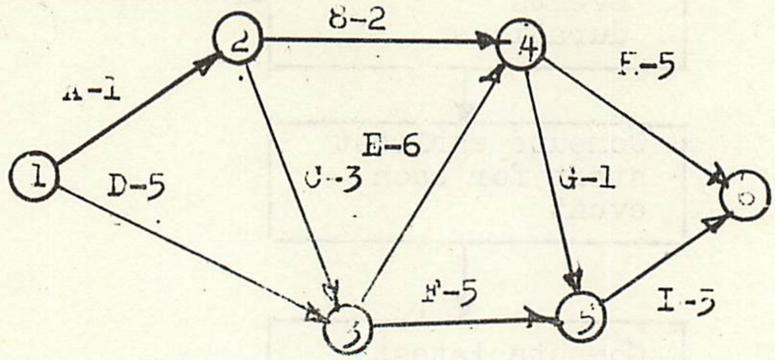

Figure (15)

Input data will be as follows:

		CC: 1	2	3	4	(5)	6	7	8	9	10
Card 1				9							
Card 2	A			1			2				1
	D			1			3				5
	C			2			3				3
	B			2			4				2
	E			3			4				6
	F			3			5				5
	G			4			5				1
	H			4			6				5
	I			5			6				5

The limitation on the number of activities to be handled by the program is given by the DIMENSION Statement of the program and should be changed accordingly if necessary.

Output of the Program:

The output of the program gives a listing of each activity defined by its end events i & j together with its duration. The program gives the following information about each activity

ES = Earliest start of the activity.

EC = Earliest completion of the activity
 = ES + DURATION

LC = latest completion of the activity.

LS = lastest start of the activity.
 = LC - DURATION.

TF = total float = LC - ES - DUR.

FF = free float which is the time available to complete an activity assuming all activities in a project start, and are completed as early as possible.

The program also gives the total project duration in time units. The critical path activites will be indicated in the output by corresponding asterisks.

FORTRAN PROGRAM FOR THE BROBLEM.

```
C       A PROGRAM FOR PERT/TIME SCHEDULING BY CPM
        DIMENSION J(100),I(100),SE(100),SL(100),D(100)
   100  FORMAT(2I3,F4.0)
   102  FORMAT(20HPERT-TIME SCHEDULING)
   103  FORMAT(///18HPPOJECT DURATION =,E15.3)
   106  FORMAT(2I4,2XF5.1,2X2F5.1,2X2F5.1,2X F5.1)
   107  FORMAT(2I4,2XF5.1,2X2F5.1,2X2F5.1,2X F5.1, 2H *)
   108  FORMAT (//2X 8HACTIVITY,2X 3HDUR,4X 2HES,3X 2HEC,5X 2HLS,3X 2HLC
,
       1 5X 2HTF)
C       INITIALIZATION
      7 DO 2 K=1,100
        J(K)=0
        I(K)=0
        SE(K)=0
        SL(K)=0
      2 D(K)=0
        K2=2
        K3=1
C       READ INPUT
        READ 100,M
        DO 5 K=1,M
      5 READ 100,I(K),J(K),D(K)
C       COMPUTE EARLIEST-START TIMES
      9 DO 10 N=1,M
        K1=N
        IF(J(N)-K2)10,50,10
     50 K3=1
        K4=I(K1)
        T=SE(K4)+D(K1)
        IF(SE(K2)-T)51,10,10
     51 SE(K2)=T
     10 CONTINUE
        IF(K3)11,20,11
     11 K2=K2+1
        K3=0
        SE(K2)=0
        GO TO 9
C       COMPUTE LATEST-START TIMES
     20 K2=K2-2
        NUM=K2+1
        SL(NUM)=SE(NUM)
     14 SL(K2)=SL(NUM)
        DO 15 K1=1,M
        IF(I(K1)-K2) 15,30,15
```

```
30 K4=J(K1)
   T =SL(K4)-D(K1)
   IF(T-SL(K2)) 31,15,15
31 SL(K2)=T
15 CONTINUE
   IF(K2-1) 60,60,16
16 K2=K2-1
   GO TO 14
C     FINAL RESULTS
60 PRINT 102
   PRINT 103,SE(NUM)
   PRINT 108
   DO65 N2=1,M
   K1=J(N2)
   K2=I(N2)
   CE=SE(K2)+D(N2)
   CL=SL(K1)-D(N2)
   TF=SL(K1)-CE
   IF (SL(K1) - SE(K1))  70, 71, 70
71 IF (SL(K2) - SE(K2))  70, 72, 70
72 AR=SL(K1)-SL(K2)-D(N2)
   IF (AR)  70, 73, 70
73 PRINT 107,K2,K1,D(N2),SE(K2),CE,CL,SL(K1),TF
   GO TO 65
70 PRINT 106,K2,K1,D(N2),SE(K2),CE,CL,SL(K1),TF
65 CONTINUE
C     START NEW PROBLEM
   GO TO 7
   END
TURN SW 1 ON FOR SYMBOL TABLE, PRESS START
END OF PASS I
```

Illustrative Solved Example:

To illustrate how the computer gives the final results of the problem, we give hereunder the computer output for the above example:

PERT-TIME SCHEDULING

PPOJECT DURATION = 16000.000E-03

ACTIVITY		DUR	ES	EC	LS	LC	TF	
1	2	1.0	0.0	1.0	1.0	2.0	1.0	
1	3	5.0	0.0	5.0	0.0	5.0	0.0	*
2	3	3.0	1.0	4.0	2.0	5.0	1.0	
2	4	2.0	1.0	3.0	9.0	11.0	8.0	
3	4	6.0	5.0	11.0	5.0	11.0	0.0	*
3	5	5.0	5.0	10.0	8.0	13.0	3.0	
4	5	1.0	11.0	12.0	12.0	13.0	1.0	
4	6	5.0	11.0	16.0	11.0	16.0	0.0	*
5	6	3.0	12.0	15.0	13.0	16.0	1.0	

Note : Asterisks indicate critical activities.

References

1. Gabriel N. Stilian and others,
PERT A new management planning and control technique.

2. General Electric, Computer Department,
GE-225 Application - Critical Path Method Program, 1962

3. IBM,
General Information Manual , PERT .... a dynamic project planning & control method.
E20 - 8067 - 1

4. IBM,
Application Program Bulletin.
IBM 7090 PERT COST Program.
H20 - 6297.