8

CPM : Network Analysis

8.1. INTRODUCTION

The Critical Path Method, commonly abbreviated as CPM was discovered independently of PERT, by Du Pont and Spery Rand Corporation in 1957, for applications to industrial situations like construction, manufacturing, maintenance etc. Since then, it has found wide acceptance by construction industry with application to bridges, dams, tunnels, buildings, highways, power plants etc. Following are the examples from fairly diverse field where application of CPM can be made :

1. Building a new bridge across river Ganges.
2. Constructing a multi-storeyed building.
3. Extension of a factory building.
4. Shifting a manufacturing unit to another site.
5. Overhauling of a diesel engine.
6. Manufacture of a new car.

The above mentioned projects are more or less *unique* in nature. Either the projects are repetitive in nature such as 'overhaul of a diesel engine', or of such a nature that similar projects have been executed previously. However, take the example of building a new bridge across river Ganges : neither in the past nor in future, this project is likely to be undertaken in the same form. But activities required to complete the above project can be listed easily and earlier experience and knowledge can be used for gaining information regarding execution, duration, resources, cost etc. of these activities. Hence realistic and more accurate information can be obtained and the results so available will be more reliable than in PERT.

CPM networks are generally used for repetitive type projects, or for these projects for which fairly accurate estimate of time for completion of each activity can be made, and for which cost estimations can be made with fair degree of accuracy. However, it is not suitable for research and development projects.

PERT and CPM *differ* on the following points :

(*i*) In CPM, time estimates for completion of activities are with fair degree of accuracy, while in PERT system, time estimates are not so accurate and definite.

(*ii*) In CPM, cost optimisation is given prime importance. The time duration for completion depends on this cost optimisation. The cost is not directly proportional to time. The cost is minimum corresponding to a certain *optimum time duration* and the cost *increases* if the time duration is either *increased* or *decreased*. On the other hand, in PERT, it is assumed that cost varies directly with time. Attention is, therefore, paid to minimise the time so that minimum cost results. Thus, in CPM *cost* is the direct controlling factor while in PERT, time is controlling factor.

As the name suggests, the, *critical path* in the CPM method plays an important role in planning and scheduling. A *critical path* is the timewise *longest path* in a network. A similar term was used in PERT also. However, in PERT, a critical path is the path that joins the *critical events* ; this path was determined on the basis of *slack* at each event. In CPM, however, a *critical path* is the one which passes through *critical activities*. This critical path is determined on the basis of minimum *float* for each activity. In other words, *in PERT the critical path is determined on the event-oriented slack philosophy while in CPM, the critical path is determined on activity-oriented float philosophy*.

8.2. CPM : PROCESS

CPM has several levels of application in project management, namely : *planning, scheduling* and *controlling*.

Planning

Planning is the most important of project management, in which the logical sequence in which the jobs or activities must be performed, is formalised. The logic should be reviewed with correctness. It should be ascertained that all the activities are shown, and the scope of the project has been interpreted correctly, and also that the resources that are required for performing each job are applied. Resources can be time, money, manpower, equipment and facilities.

Scheduling

Scheduling is the determination of time required for execution of each operation and the time order in which each operation has to be carried out, to meet the plan objectives. Scheduling has to be done not only in respect of operations but also in respect of *resources*. The resources, in general, include *time, space, equipment, material* and *effort*. More specifically, scheduling is the mechanical process of formalising the planned functions, assigning the starting and completion dates to each part (or activity) of the project in such a manner that the whole project proceeds in a logical sequence and in an orderly and systematic manner.

Controlling

Controlling is the process in which difference or deviations between the plan and actual performances are reviewed after the project has started. The analysis and correction to these deviations form the basic aspect of control. Replanning and rescheduling is done to compensate for the deviations. In CPM, controlling is required not only in respect of physical progress of work, but also in respect of cost.

The complete *process* of CPM application can be summarised in the following major *steps* :

1. DESCRIBE	:	project in terms of dependencies among activities, *i.e.* plan the project.
2. DETERMINE	:	the schedule of the activities.
3. PREDICT	:	those activities which control significant target dates of the project.
4. ANALYSE	:	the schedule developed.
5. REPLAN	:	the project if analysis so indicates.
6. ALLOCATE	:	resources to project in an efficient manner for the schedule developed.
or MAKE	:	schedule changes required by resources limitations.
7. DEVELOP	:	time-cost relationship for activities and optimise total project cost by selecting suitable project duration.

Fig. 8.1 shows the CPM process diagrammatically.

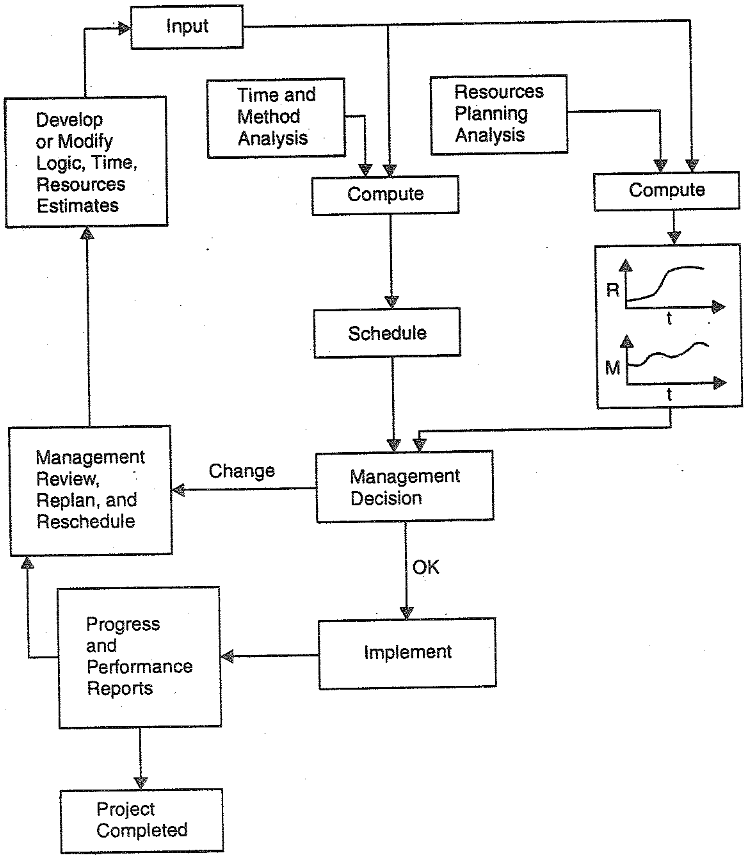

FIG. 8.1. CPM PROCESS.

8.3. CPM : NETWORKS

CPM networks are generally *activity-oriented* while the PERT networks are *event-oriented*. Essentially, this means that in activity oriented networks, the arrows representing activities or jobs are lebelled with some description of activity, while in event-oriented networks, events are the object of interest and are appropriately described. *Activities* are usually operations which take time to carry out, and on which resources are expended. The junctions between activities are termed as *events*. An event is a point in time, a milestone representing the beginning or the completion of some activity or group of activities. In CPM network, each activity is represented by an arrow, and the sequence in which the activities are performed is shown by the sequence of the arrows. The activities are suitably lebelled. However, both events and activities can be lebelled if so desired, though the tendency seems to be towards activity levelling, since it is the *activity* with which engineers are most directly concerned. Engineers design activities, not events. When a project falls behind schedule, it is the critical activities which receive managerial attention.

In CPM network planning, a tentative list of activities, that must be performed in order to accomplish the project objectives, is prepared. These activities are then interconnected to show how they must be performed with respect to one another. The placement of an activity arrow in the network, with respect to other activities of the project, is done by answering the following three basic questions :

1. What activity or activities must immediately precede this activity ?
2. What activity or activities cannot be started until after the completion of this activity ?
3. What activity or activities can be performed concurrently with this activity ?

The process of drawing a network consists of the following :

(*a*) listing the activities of a project

(*b*) answering the three questions with regard to each activity

(*c*) drawing the network that represents on paper answer to these three questions. The interdependencies can be represented by suitable use of dummies.

The elements of network have been discussed in detail in chapter 3, giving the network rules. However, for illustration purposes, let us take the project of purchasing a new heavy duty lathe and disposing the old lathe. The project consists of the following activities :

(*i*) Await delivery of lathe,

(*ii*) Remove existing lathe,

(*iii*) Install power supply,

(*iv*) Install lathe,

(*v*) Connect to power,

(*vi*) Test and

(*vii*) Dispose of existing lathe.

Fig. 8.2 shows the network for the project.

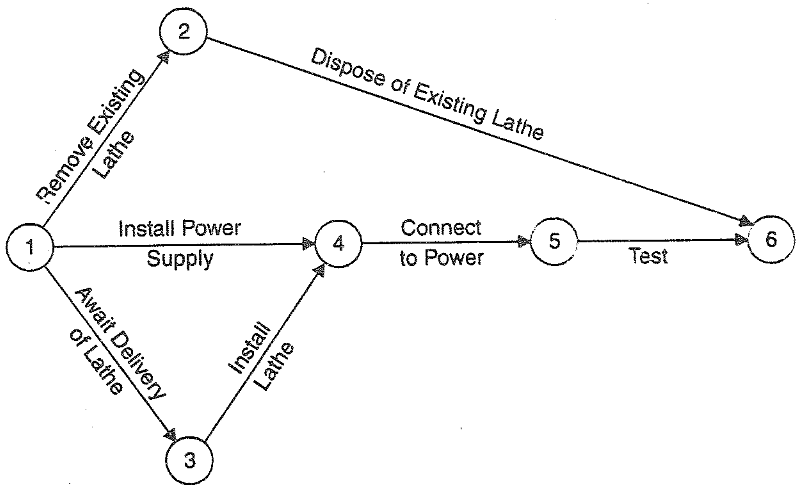

FIG. 8.2

Numbering the events

Though CPM networks are activity-oriented, the events constitute important control points. The events should, therefore, be so numbered that they reflect the logical sequence of the activities. This can be best done by following Fulkerson's rule, discussed in § 3.8.

8.4. ACTIVITY TIME ESTIMATE

After finalising the network, the next step is to estimate the time required for the completion of each activity of the network. Two approaches may be used for the assessment of duration of activity completion : (*a*) *probabilistic approach* and (*b*) *deterministic approach*. The first approach (*i.e.* the probabilistic approach) is followed by PERT planners, in which three time estimates are made for each activity : the optimistic time (t_O), the likely time (t_L) and the pessimistic time (t_P). In the second approach (deterministic approach), we may assume that we know enough about each job or operation, so that a single time estimate of their duration is sufficiently accurate to give reasonable results. This approach is followed in the CPM networks. No uncertainties are taken into consideration.

The two approaches are represented in Fig. 8.3. In Fig. 8.3 (*a*), the time estimates of the activity has greater range and hence greater uncertainty. In Fig. 8.3 (*b*), however, the range of variation is very narrow, and we approach towards a more deterministic model.

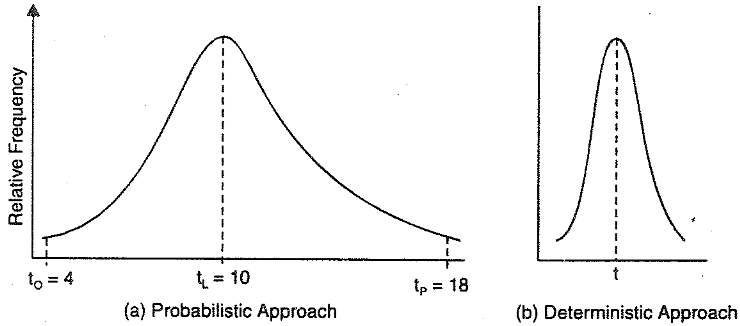

FIG. 8.3. ACTIVITY TIME ESTIMATE.

In PERT, we calculated the expected time t_E for each activity, from the three time estimates. This expected time had the same degree of uncertainty, reflected in the range of the time estimates. In CPM, however, no calculations are required. The estimated time, represented simply by t, is directly used for network analysis. The activity duration t is directly written near the arrow representing that activity.

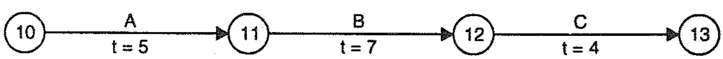

FIG. 8.4

For example, the time of completion (t) of activity A (or 10—11) in Fig. 8.4 is 5 units, while for those of activity B (or 11—12) and activity C (or 12—13), are 7 and 4 units respectively. The time of completion for any activity $i - j$ is denoted by symbol t^{ij}, in the place of symbol t_E^{ij}, used in PERT.

8.5. EARLIEST EVENT TIME

After having determined or estimated the time duration (t) for an activity, let us now determine the time of occurrence of an event at the head of the activity. The *earliest occurrence time* or *earliest event time* (T_E) is the earliest time at which an event can occur. It is the time by which all the activities discharging into the event under consideration are completed. The term is analogous to the *earliest expected time* used in the PERT analysis, except that the degree of uncertainty inherent in the word 'expected,' is not there.

In a CPM network, the time of completion of each activity (t^{ij}) is known. Hence the earliest occurrence time can be easily calculated. For example, event 10 for Fig. 8.4 is the beginning of the project, and hence it can be assumed to occur at zero time. Event 2 is at the end of activity *A* for which completion duration (t) is 5 days. Hence event 11 occurs at 0 + 5 = 5 days. Similarly, event 12 will occur at 5 + 7 = 12 days and end event 13 will occur at 12 + 4 = 16 days. These event times (t_E) are entered at the top of the node, as shown in Fig. 8.5.

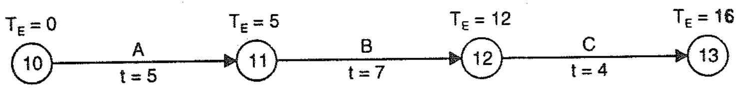

FIG. 8.5

From the above discussions, one can formulate the following expression for T_E of any event *j*

$$T_E^j = T_E^i + t^{ij} \quad ...(8.1)$$

where T_E^i = earliest occurrence time for the tail event

T_E^j = earliest occurrence time for the head event

ij = activity under consideration

t^{ij} = time of completion of activity ij

Let us now take the case of a network shown in Fig. 8.6.

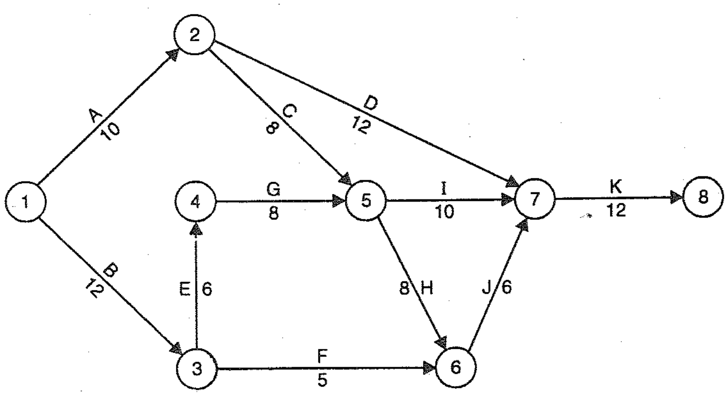

FIG. 8.6

In this network, any event, say No. 5 can be reached by two paths : Path A through 1—2—5 and path B through 1—3—4—5.

Along path 1—2—5, T_E for event 5 = 10 + 8 = 18

Along path 1—3—4—5, T_E for event 5 = 12 + 6 + 8 = 26.

From the above, we note that T_E for event 5 are different for the two activity paths. *However, no event can be considered to have reached until all activities leading to the event are completed.* Hence event 5 cannot be considered to have occurred until all the activities along both the paths are complete. Thus, T_E will be the greater of the two, and its value will be 26.

From the above discussions, the earliest occurrence time or earliest event time (T_E^j) for any event j can be found from the following expression.

$$T_E^j = (T_E^i + t^{ij})_{max} \qquad ...(8.2)$$

Based on Eqns. 8.1 and 8.2, the earliest occurrence times for various events of network shown in Fig. 8.6 will be as under.

Event 1 : Since it does not have any predecessor event,

$$T_E^1 = 0$$

Event 2 : $T_E^2 = T_E^1 + t^{1-2}$

$= 0 + 10 = 10$

Event 3 : $T_E^3 = T_E^1 + t^{1-3}$

$= 0 + 12 = 12$

Event 4 : $T_E^4 = T_E^3 + t^{3-4}$

$= 12 + 6 = 18$

Event 5 : It has two predecessor events, 2 and 4. For each of these,

$$T_E^5 = T_E^2 + t^{2-5} = 10 + 8 = 18$$

and $T_E^5 = T_E^4 + t^{4-5} = 18 + 8 = 26$

Hence $T_E^5 = 26$.

Event 6 : It has two predecessor events 5 and 3

$\therefore$ $T_E^6 = T_E^5 + t^{5-6} = 26 + 8 = 34$

and $T_E^6 = T_E^3 + t^{3-6} = 12 + 5 = 17$

$\therefore$ $T_E^6 = 34$

Event 7 **:** It has three predecessor events : 2, 5 and 6

$\therefore \quad T_E^7 = T_E^2 + t^{2-7} = 10 + 12 = 22$

$T_E^7 = T_E^5 + t^{5-7} = 26 + 10 = 36$

and $\quad T_E^7 = T_E^6 + t^{6-7} = 34 + 6 = 40$

Hence $\quad T_E^7 = 40$

Event 8 **:** It has only one predecessor event, 7.

$\because \quad T_E^8 = T_E^7 + t^{7-8}$

$= 40 + 12 = 52.$

These computations may also be conveniently done by entering T_E^j values for each node, obtained from different paths, in squares, and cancelling each except the one giving highest value of T_E. *Forward pass* is used, going from one node to the other, as illustrated in Fig. 8.7, for the network of Fig. 8.6. It should be noted

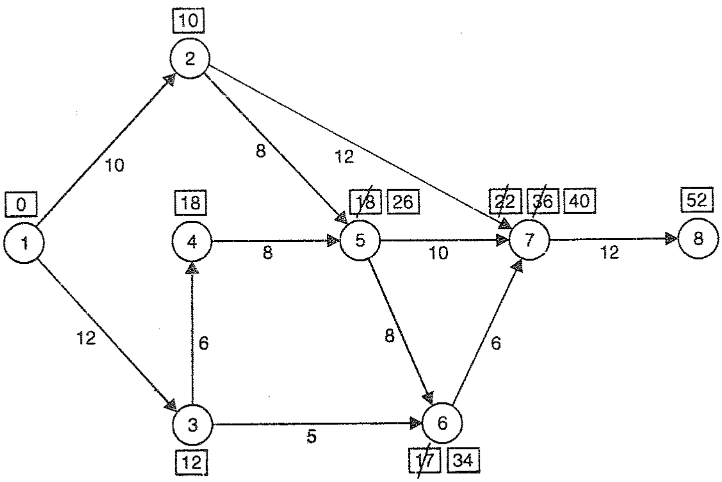

FIG. 8.7

that if the node has only one predecessor event, only one square will be there. If it has 3 predecessor events (such as for event 7), it will have three values of T_E entered in three squares made over the node ; only that square will be retained which will have highest value of T_E. The method is not very suitable for large networks, where no space is available for making so many squares.

For large networks, computations are usually done in a tabular form. This is illustrated in Table 8.1, in which events are tabulated, starting with the end event, for convenience in computations. Thus, event 8 is entered first in the successor event column. It's predecessor event (7) is entered in the predecessor event column. The next *lower event* 7 is then entered in the successor event column. It has three predecessor events (6, 5, 2) which are entered in the predecessor event column in the decreasing order of their numerical value. This procedure is followed till all the events are entered. The last successor event is 2 which has predecessor event 1. The initial event is not entered in the successor event column since it does not have any predecessor event.

Then the activity time (t^{ij}) is entered for each activity i—j joining the predecessor event i and successor event j. The column for T_E^j is then filled, starting from the bottom, and using the relations

$$T_E^j = T_E^i + t^{ij}$$

and

$$T_E^j = (T_E + t^{ij})_{max}.$$

When more than one value of T_E^j is obtained for an event, the highest value is underlined for forward use.

Table 8.1

Computations for Earliest Event Time (T_E^j)

Successor event j	*Predecessor event i*	*Activity i—j*	t^{ij}	T_E^j
8	7	7—8	12	<u>52</u>
7	6	6—7	6	<u>40</u>
	5	5—7	10	36
	2	2—7	12	22
6	5	5—6	8	<u>34</u>
	3	3—6	5	27
5	4	4—5	8	<u>26</u>
	2	2—5	8	18
4	3	3—4	6	<u>18</u>
3	1	1—3	12	<u>12</u>
2	1	1—2	10	

8.6. LATEST ALLOWABLE OCCURRENCE TIME

The *latest allowable occurrence time* or the *latest event time* is the latest time by which an event must occur to keep the project on schedule. It is denoted by symbol T_L. If the *scheduled completion time* (T_S) of the project is given, the latest *event time* of the end event will be equal to T_S. If the scheduled completion time is not specified, then T_L is taken equal to the earliest event time T_E.

The latest event time for an activity is computed by starting from the tail event and using the *backward pass*. For illustration, let us again consider the network of Fig. 8.4, for which calculated values of T_E for each event are shown in Fig. 8.5. The *latest event time* for end event $(T_L^{13}) = T_E^{13} = 16$.

The activity 12—13 takes 4 days. Hence event 12 cannot occur later than 16 – 4 = 12 days. Thus, $T_L^{12} = 12$.

Similarly, for event 11,

$$T_L^{11} = T_L^{12} - t^{11-12} = 12 - 7 = 5$$

and $$T_L^{10} = T_L^{11} - t^{10-11} = 5 - 5 = 0.$$

The values of T_L so obtained are entered either at the top or at the bottom of each node as shown in Fig. 8.8.

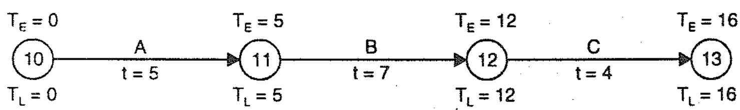

FIG. 8.8

From the above discussion, one can formulate the following expression for T_L^i for a predecessor event i, from the known value T_L^j of a successor event :

$$T_L^i = T_L^j - t^{ij} \quad ...(8.3)$$

Let us now take the case of another network of Fig. 8.6. The values of T_E for each event are known. The project completion time is taken equal to T_E, since T_S is not given. Hence $T_L = T_E = 52$, for the last event. For this network, some events have more than one successor events.

Event 7. $T_L^7 = T_L^8 - t^{7-8}$

$= 52 - 12 = 40$

Event 6. $T_L^6 = T_L^7 - t^{6-7}$

$= 40 - 6 = 34$

Event 5. It has two successor events : 7 and 6

$\therefore \qquad T_L^5 = T_L^7 - t^{6-7} = 40 - 10 = 30$

Also, $\qquad T_L^5 = T_L^6 - t^{5-6} = 34 - 8 = 26$

Out of these, the *minimum value* (*i.e.* 26) will be the appropriate value of T_L^5.

Event 4. $T_L^4 = T_L^5 - t^{4-5}$

$= 26 - 8 = 18$

Event 3. It has two successor events : 6 and 4

$\therefore \qquad T_L^3 = T_L^6 - t^{3-6} = 34 - 5 = 29$

and $\qquad T_L^3 = T_L^4 - t^{3-4} = 18 - 6 = 12$

$\therefore \qquad T_L^3 = 12$ (minimum)

Event 2. This has also two successor events : 7 and 5

$\therefore \qquad T_L^2 = T_L^7 - t^{2-7} = 40 - 12 = 28$

and $\qquad T_L^2 = T_L^5 - t^{2-5} = 26 - 8 = 18$

$\therefore \qquad T_L^2 = 18.$

Event 1. This has also two successor events : 3 and 2

$\therefore \qquad T_L^1 = T_L^3 - t^{1-3} = 12 - 12 = 0$

and $\qquad T_L^1 = T_L^2 - t^{1-2} = 18 - 10 = 8$

$\therefore \qquad T_L^1 = 0$

Thus, we have the following expression for T_L^i

$$T_L^i = (T_L^j - t^{ij})_{min}. \qquad \text{...(8.4)}$$

These computations may also be conveniently done by entering T_L^i values for each node, obtained from different paths, in triangles and cancelling each except the one giving the least value of T_L. *Backward pass* is used, going from one node to the other as illustrated in Fig. 8.9. It should be noted that if an event has only one successor event, only one triangle will be there. If it has 2 successor events it will have two values of T_L entered in two triangles made over or below the node ; only that triangle will be retained which will have the lowest value of T_L. The method is not very suitable for large networks ; space limitations are there for making so many triangles.

For large networks, computations are usually done in a tabular form. This is illustrated in Table 8.2. The first two columns

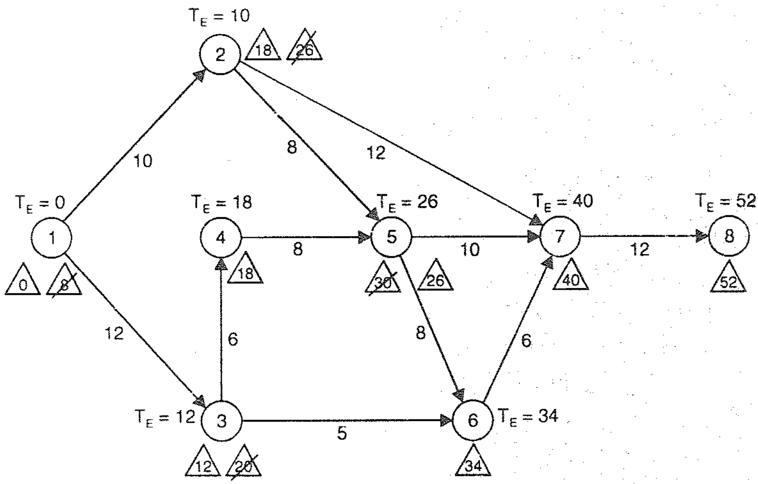

FIG. 8.9

Table 8.2
Computation of Latest Event Occurrence Time (T_L)

Predecessor event i	*Successor event j*	*Activity i—j*	t^{ij}	T_L^i
7	8	7—8	12	40
6	7	6—7	6	34
5	7	5—7	10	30
	6	5—6	8	26
4	5	4—5	8	18
3	6	3—6	5	29
	4	3—4	6	12
2	7	2—7	12	28
	5	2—5	8	18
1	3	1—3	12	0
	2	1—2	10	8

are for the predecessor events (i) and successor events (j) respectively. Here also, the tabulation is done with the highest numbered events, decreasing downwards in numerical order. Column 3 is for the activity i—j while column 4 is for the corresponding completion time t^{ij} of the activity. The computation T_L^i is started in the *backward direction*, starting with the highest number predecessor event. The following equations are used :

$$T_L^i = T_L^i - t^{ij} \text{ (for single path)}$$

and

$$T_L^i = (T_L^j - t^{ij})_{min} \text{ for multiple paths.}$$

The computations are continued till the first event is reached. For multiple paths, the minimum value of T_L^i is underlined, for further use, as the appropriate value of the latest event time.

8.7. COMBINED TABULAR COMPUTATIONS FOR T_E AND T_L

The *earliest event time* (T_E) is computed using *forward pass,* as illustrated in Table 8.1, while the *latest allowable event time* (T_L) is computed using backward pass, as illustrated in Table 8.2. Since both T_L and T_E are to be calculated for each event, their computations are done in a combined tabular form illustrated in Table 8.3.

For illustration, we will take the same network of Fig. 8.6. Column 1 of Table 8.3 gives the event number, starting with the initial event and proceeding in the direction of increasing numbers of the events. Column 2 gives the predecessor events while column 6 gives the successor events to the events of column 1. These columns are completed first, using the network. An event under consideration (column 1) may have one or more than one predecessor events (column 2), and one or more than one predecessor events (column 6). A horizontal line is drawn after entering all the predecessor events and successor events to every event of column 1.

Then, computations are done for the *earliest event time* (T_E) in columns 3, 4 and 5. Column 3 is for the activity time t^{ij} where j is the event under consideration (column 1) and i is the predecessor event (column 2). T_E^j is computed from the relation

$$T_E^j = T_E^i + t^{ij}$$

Where there are more than one predecessor events, several values of T_E^j are obtained, which are entered in column 4. The maximum value of T_E^j is underlined. This underlined value is the appropriate value of the earliest event time for the event under consideration (column 1), and is entered as T_E in column 5. For the computation of T_E, we thus use the *forward pass*, starting from the initial event and proceeding in the downward direction (↓) in the Table.

Then we compute the *latest event (occurrence) time* of the same events under consideration (column 1), in columns 7, 8 and 9. Column 7 is the activity time t^{ij} where i is the event under consideration and j is the successor event (column 6). T_L^i is computed from the relation

$$T_L^i = T_L^j - t^{ij}$$

Table 8.3
Computations of T_E and T_L

Event No.	*Earliest event time (↓)*				*Latest event time (↓)*			
	Predecessor event (i)	t^{ij}	T_E^j	T_E	*Successor event (j)*	t^{ij}	T_L^i	T_L
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
1	—	—	0	0	2 3	10 12	8 0	0
2	1	10	10	10	5 7	8 12	18 28	18
3	1	12	12	12	4 6	6 5	12 29	12
4	3	6	18	18	5	8	18	18
5	2 4	8 8	18 26	26	6 7	8 10	26 30	26
6	3 5	5 8	17 34	34	7	7	34	34
7	2 5 6	12 10 6	22 36 40	40	8	12	40	40
8	7	12	52	52	—		52	52

Computations are done by backward pass, starting with the end event and proceeding upwards (↑) in the Table. If T_S is not given, T_L of the last event is taken equal to its T_E. Where there are more than one successor events, several values of T_L^i are obtained, which are entered in column 8. The minimum value of T_L^i is underlined. This underlined value is the appropriate value of the *latest event time* for the event under consideration, and is entered as T_L in column 9.

Thus, for each of the activities of column 1, T_E is given in column 5 while T_L is given in column 9.

8.8. START AND FINISH TIMES OF ACTIVITY

So far we have discussed two event times : the *earliest event time* (T_E) and the *latest allowable occurrence time* (T_L). Since CPM networks are activity oriented, the following activity times are useful for network computations :

(*i*) Earliest start time

(*ii*) Earliest finish time

(*iii*) Latest start time

(*iv*) Latest finish time.

1. Earliest Start Time (EST)

The *earliest start time* for an activity is the earliest time by which it can commence. This is naturally equal to the earliest event time associated with the tail of the activity arrow. It is abbreviated by EST.

Thus, EST = Earliest event time at its tail.

If the activity is denoted by *i—j*, and if the earliest event time at its tail is T_E^i, we have

$$EST = T_E^i \qquad ...(8.5)$$

2. Earliest Finish Time (EFT)

If an activity proceeds from its early start time and takes the estimated duration for completion, then it will have an early finish. Hence *earliest finish time* (EFT) for an activity is defined as the earliest time by which it can be finished. This is evidently equal to the earliest start time plus estimated duration of the activity :

$$EFT = \text{earliest start time} + \text{activity duration}$$

or

$$EFT = T_E^i + t^{ij} \qquad ...(8.6)$$

3. Latest Start Time (LST)

Latest start time for an activity is the *latest time* by which an activity can be started without delaying the completion of the project. For 'no delay' condition to be fulfilled it should be naturally equal to the latest finish time (LFT) minus the activity duration.

$$\therefore \qquad \text{LST} = \text{LFT} - \text{Activity duration.}$$

Since the Latest Finish Time (LFT) is equal to T_L^j (See Eqn. 8.8), we have

$$\text{LST} = T_L^j - t^{ij} \qquad \text{...(8.7)}$$

4. Latest Finish Time (LFT)

The latest finish time for an activity is the latest time by which an activity can be finished without delaying the completion of the project. Naturally, the latest finish time for an activity will be equal to the latest allowable occurrence time for the event at the head of the arrow. Hence

$$\text{LFT} = \text{Latest event time at the head of activity arrow}$$

or $$\text{LFT} = T_L^j \qquad \text{...(8.8)}$$

From the above definitions, we note that out of the *four* times of an activity, the earliest start time and the latest finish time are equal to the earliest event time (T_E^i) at its tail and the latest event time (T_L^j) at its head respectively, the values of which are already available. From these, the earliest finish time and the latest start time can be found by simply adding activity time to T_E^i and by subtracting activity time from T_L^j respectively.

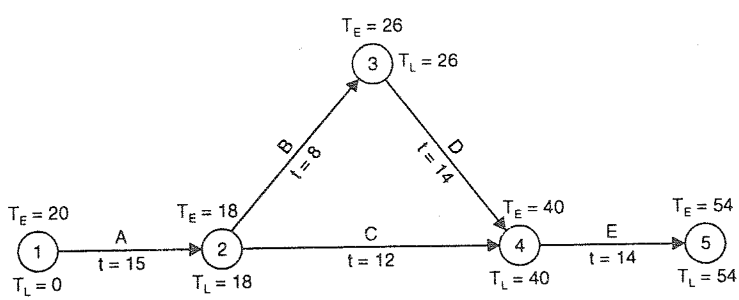

FIG. 8.10

For example, consider the network of Fig. 8.8. For activity 11-12, we have

$$EST = T_E^i = T_E^{11} = 5$$
$$EFT = EST + t^{11-12} = 5 + 7 = 12$$
$$LST = T_L^j - t^{ij} = T_L^{12} - t^{11-12} = 12 - 7 = 5$$
$$LFT = T_L^j = T_L^{12} = 12.$$

Here we find that EST and LST are the same ; also the EFT and LFT are also the same.

Again consider a *partial network* situation shown in Fig. 8.10.

Consider activity 2—4

$$EST = T_E^j = T_E^2 = 18$$
$$EFT = T_E^i + i^{ij} = 18 + 12 = 30$$
$$LST = T_L^j - t^{ij} = T_L^4 - t^{2-4} = 40 - 12 = 28$$
$$LFT = T_L^j = T_L^4 = 40.$$

The above *four activity times* are generally written on the activity arrow, along with activity *name* and *activity time* (t^{ij}), as illustrated in Fig. 8.11. The activity name [*i.e.* C etc.], EST and EFT are written on the top of the arrow, while the activity duration (t^{ij}), LST and LFT are written below the arrow.

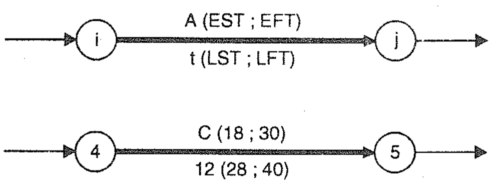

FIG. 8.11

8.9. FLOAT

The term *float* is associated with the activity times. It is analogous to the term *slack* which was associated with the event times. Just as the *slack* denotes the flexibility range within which an event can occur (*i.e.* it is the difference between the earliest event time and latest event occurrence time), *float* denotes the range within which an activity start time or its finish time may fluctuate without affecting the completion of the project.

Floats are of the following types :

(*i*) Total float

(*ii*) Free float

(*iii*) Independent float

(*iv*) Interfering float.

1. Total Float

Total float is the time span by which the starting (or finishing) of an activity can be delayed without delaying the completion of the project. In certain activities, it will be found that there is a difference between maximum time available and the actual time required to perform the activity. *This difference is known as the total float.*

Effectively, it is an inbuilt reserve of a resource (time) which is available for use in certain tail event and head event. Now, maximum time available to perform events can therefore be considered as setting the limits.

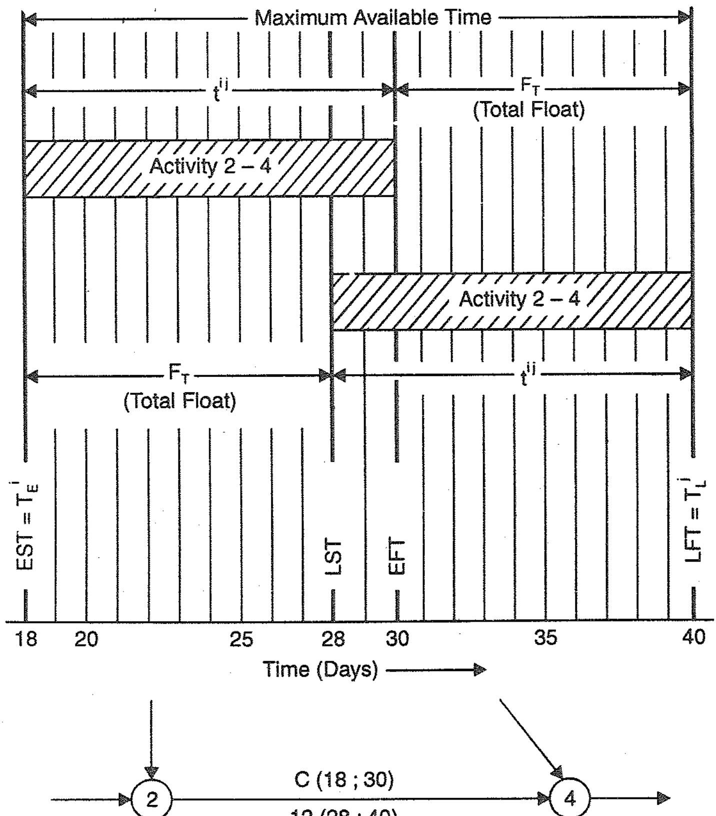

FIG. 8.12

Consider an activity i—j. The time duration available for this activity is equal to the difference between its earliest start time (T_E^i) and the latest finish time (T_L^j) :

$\therefore$ Maximum time available = $T_L^j - T_E^i$

activity time required = t^{ij}

$\therefore$ Total float (F_T) = time available – time required

or $F_T = (T_L^j - T_E^i) - t^{ij}$ ...(8.9)

or $F_T = T_L^j - (T_E^i + t^{ij})$ ...[8.9 (*a*)]

Also $F_T = (T_L^j - t^{ij}) - T_E^i$ ...[8.9 (*b*)]

Now from Eqn. 8.7, $T_E^j - t^{ij}$ = LST

and from Eqn. 8.5, T_E^i = EST

$\therefore$ F_T = LST – EST ...(8.10)

Similarly F_T = LFT – EFT ...[8.10 (*a*)]

Eqn. 8.10 is directly used for computing the total float of an activity i—j.

For example, for the activity 2—4 of Fig. 8.10, the *total float*

F_T = LST – EST

= 28 – 18 = 10

This means that even if the starting of this activity is delayed by 10 days, the project completion will not be delayed.

Total float of an activity *ij* (2—4) is denoted diagrammatically in Fig. 8.12.

It should be clearly noted that the total float for each activity is a measure of its particular relationship to all other activities in the project, since the earliest start time ties in all preceding activities and the latest finish time ties in all succeeding activities.

2. Free Float

Free float is that portion of positive total float that can be used by an activity without delaying any succeeding activity (or without affecting the total float of the succeeding activity). The concept of *free float* is based on the possibility that all the events occur at their earliest times (*i.e.* all activities start as early as possible).

To get a clear concept of the free float, consider activity *i-j* and its successor activity *j-k*. Events *i* and *j* has earliest occurrence times as T_E^i and T_E^j. Earliest start time for activity ij will be T_E^i, while EST for j-k will be T_E^j. However, if t^{ij} is the activity time, activity i-j will be complete by $(T_E^i + t^{ij})$ time, while activity j-k cannot start because its EST (T_E^j) is greater than $(T_E^i + t^{ij})$. The difference between the two is the free float for i-j.

$$\therefore \quad F_F \text{ for } i\text{-}j = T_E^j - (T_E^i + t^{ij}) \qquad \text{...(8.11)}$$

But $(T_E^i + t^{ij})$ is the early finish time (EFT) of the activity i-j, while T_E^j is the early start time for activity j-k.

$$\therefore \quad F_F \text{ for } ij = \text{EST for successor activity} - \text{EFT of present activity.}$$

or $$F_F \text{ for } ij = T_E^j - \text{EFT} \qquad \text{...(8.12)}$$

Hence free float for an activity i-j is the difference between its earliest finish time and the earliest start time of its successor activity. It can be found by taking the difference between the head event time (T_E^j) and early finish time of the activity i-j.

Again from Eqn. 8.11, F_F for i-j

$$= T_E^j - T_E^i - t^{ij} \qquad \text{...(8.11)}$$

In the above equation T_E^j is the EST for successor activity, T_E^i is the EST of present activity (i-j) and t^{ij} is the activity duration.

Hence, *free float is the excess of the available time over the required time when the activity, as well as its successor activity start as early as possible.*

Again, from Eqn. 8.9 (a),

$$F_T = (T_L^j - T_E^i) - t^{ij}$$

$$\therefore \quad T_E^i + t^{ij} = T_L^j - F_T$$

Substituting in Eqn. 8.11,

$$F_F = T_E^j - (T_L^j - F_T)$$

or $$F_F = F_T - (T_L^j - T_E^j) = F_T - S_j \qquad \text{...(8.13)}$$

Equation 8.13 gives another method of calculating the free float. *It is the difference of the total float and the head event slack.* If head slack is zero, free float will be equal to the total float.

All the three methods (*i.e.* Eqn. 8.11, 8.12 and 8.13) of calculating *free float* of an activity i-j is shown diagrammatically in Fig. 8.13. Activity 21-24 has activity duration of 12 days, with tail and head event times marked. Free float so marked satisfy both

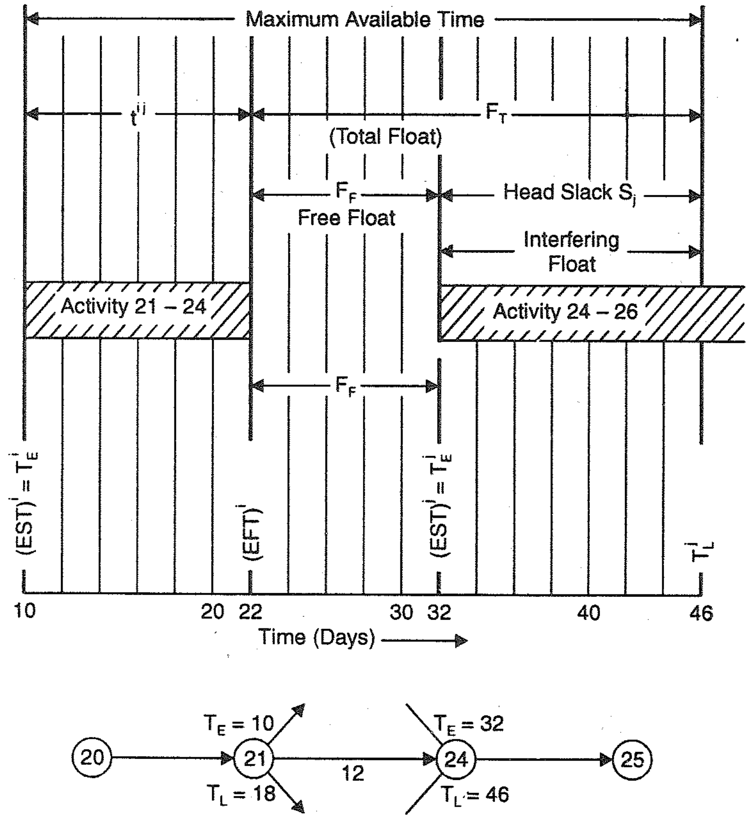

FIG. 8.13

Eqn. 8.12 as well as Eqn. 8.13. For the activity 21-35, free float is given by

$$F_F = T_E^j - T_E^i - t^{ij} = 32 - 10 - 12 = 10 \text{ days} \qquad ...(i)$$

$$F_F = T_E^j - \text{EFT} = 32 - 22 = 10 \text{ days} \qquad ...(ii)$$

Also, $F_F = F_T - S_j$

where $F_T = (T_L^j - T_E^i) - t^{ij} = (46 - 10) - 12 = 24$ days.

$S_j = T_L^j - T_E^j = 46 - 32 = 14$

$\therefore \qquad F_F = 24 - 14 = 10$ days. ...(iii)

3. Independent Float

Independent float gives us an idea about the *excess* time that exists if the preceding activity ends as *late* as possible and the succeeding activity starts as *early* as possible. *The independent float is, therefore, defined as the excess of minimum available time over the required activity duration.*

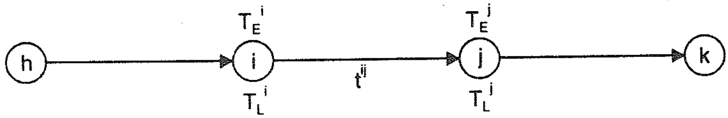

FIG. 8.14

To get the concept of independent float, consider activity *i-j* of Fig. 8.14, having a predecessor activity *h-i* and successor activity *j-k*. The *latest time* by which the predecessor activity *h-i* finishes is T_L^i, while the *earliest time* by which the successor activity *j-k* can start is T_E^j. Hence, under this condition, *minimum time available* for activity $i\text{-}j = T_E^j - T_L^i$. Thus, by definition, independent float (F_{ID}) is given by

$$F_{ID} = \text{minimum available time} - \text{activity time}$$

or $$F_{ID} = (T_E^j - T_L^i) - t^{ij} \qquad ...(8.14)$$

It should be noted that *independent float is a part of the free float*. To show this, consider Eqn. 8.11 for free float :

$$F_F = T_E^j - T_E^i - t^{ij} \qquad ...(\text{Eqn. } 8.11)$$

Substituting the value of $T_E^j - t^{ij}$ from Eqn. 8.14,

i.e. $$T_E^j - t^{ij} = F_{ID} + T_L^i, \text{ we get}$$

$$F_F = (F_{ID} + T_L^i) - T_E^i = F_{ID} + (T_L^i - T_E^i)$$

or $$F_{ID} = F_F - (T_L^i - T_E^i)$$

But $T_L^i - T_E^i$ = tail event slack = S_i

$\therefore$ $$F_{ID} = F_F - S_i \qquad ...(8.15)$$

Thus, independent float is equal to the free float minus tail event slack. If the tail slack is zero, free float and independent float are equal. It is to be noted that if a negative value of independent float is obtained, then independent float is taken as zero.

The two methods of calculating independent float (by Eqn. 8.14 and 8.15) are shown diagrammatically in Fig. 8.15.

In Fig. 8.15, activity 21-24 (*i-j* activity) has 12 days duration, with tail and head event times marked. Independent float so marked

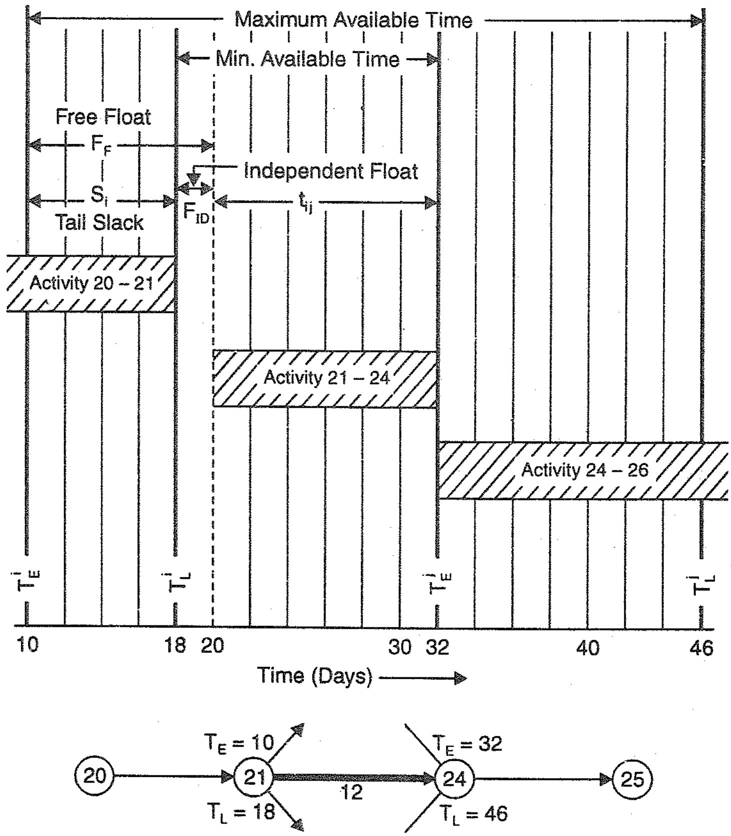

FIG. 8.15

satisfy both Eqn. 8.14 as well as Eqn. 8.15. For the activity 21-24, independent float is given by

$$F_{ID} = (T_E^j - T_L^i) - t^{ij} \quad \text{...(Eqn. 8.14)}$$

$$= (32 - 18) - 12 = 2 \text{ days} \quad \text{...}(i)$$

also, $F_{ID} = F_F - S_i$ ...(Eqn. 8.15)

where free float $= T_E^j - T_E^i - t^{ij} = 32 - 10 - 12 = 10$

and $S_i = T_L^i - T_E^i = 18 - 10 = 8$

$$\therefore \quad F_{ID} = 10 - 8 = 2 \text{ days.} \quad \text{...}(ii)$$

4. Interfering Float

Interfering float (F_{IT}) is just another name given to the head event slack (S_j), specially in CPM networks which are activity oriented. Interfering float is the potential downstream interference of any activity, and is equal to the *difference between total float and the free float.*

Thus, $F_{IT} = F_T - F_F$ ...(8.16)

But $F_T = (T_L^j - T_E^i) - t^{ij}$

$F_F = (T_E^j - T_E^i) - t^{ij}$

$\therefore$ $F_{IT} = (T_L^j - T_E^i - t^{ij}) - (T_E^j - T_E^i - t^{ij})$

or $F_{IT} = (T_L^j - T_E^j) = S_j$ ...(8.17)

Thus, interfering float is equal to the head event slack. This definition of interfering float has been incorporated in Fig. 8.13.

Summary of Floats

Let us summarise by defining the various types of floats as under :

1. ***Total float*** of an activity is the *excess* of the *'minimum available time'* over *activity time*. Thus

$$F_T = (T_L^j - T_E^i) - t^{ij} \quad ...(8.9)$$

2. ***Free float*** of an activity is the *excess* of *'available time'* over the *activity time, when all jobs start as early as possible.* Thus

$$F_T = (T_E^j - T_E^i) - t^{ij} \quad ...(8.11)$$

3. ***Independent float*** of an activity is the *excess* of *'minimum available time'* over the *activity time*. Thus

$$F_{ID} = (T_E^j - T_L^i) - t^{ij}$$

4. ***Interfering float*** of an activity is the *difference* between *total float* and *free float*. It is thus equal to the head event slack. Thus,

$$F_{IN} = F_T - F_F = S_j.$$

Example. *Let us now take the example of network shown in Fig. 8.6, for which the event times have been already computed in Table 8.4. The network is reproduced in Fig. 8.16, with the event times marked.*

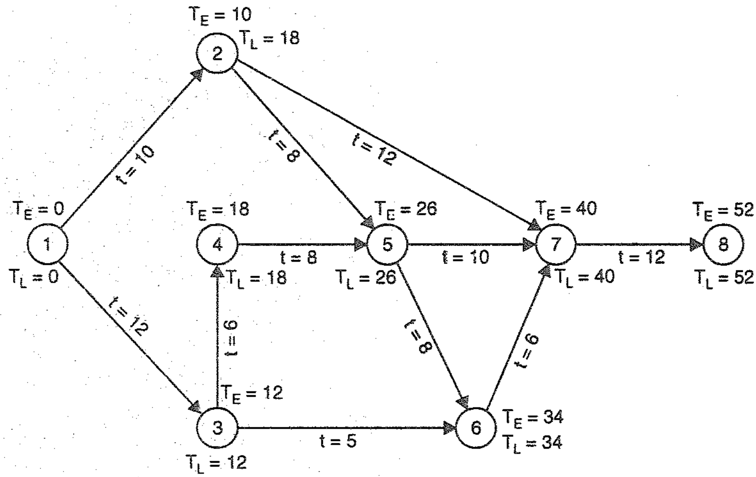

FIG. 8.16

Solution. The values of the activity times, total float, free float and independent float are tabulated in Table 8.4.

Table 8.4

Activity (i—j)	*Duration* t^{ij}	*Earliest*		*Latest*		*Total float* F_T	*Free float* F_F	*Independent float* F_{ID}
		Start time (EST)	*Finish time (EFT)*	*Start time (LST)*	*Finish time (LFT)*			
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
1—2	10	0	10	8	18	8	0	0
1—3	12	0	12	0	12	0	0	0
2—5	8	10	18	18	26	8	8	0
2—7	12	10	22	28	40	18	18	10
3—4	6	12	18	12	18	0	0	0
3—6	5	12	17	29	34	17	17	17
4—5	8	18	26	18	26	0	0	0
5—6	8	26	34	26	34	0	0	0
5—7	10	26	36	30	40	4	4	4
6—7	6	34	40	34	40	0	0	0
7—8	12	40	52	40	52	0	0	0

Table 8.4 has 9 columns, out of which column 1 gives the activity (*i-j*) while column 2 gives the activity duration (t^{ij}). While filling column 1, it is advisable to start with the lowest number event (initial event) and proceed in the next high numerical order completing all the activities originating from that event before going to the next higher event. This will facilitate in filling column 3 for early start time (EST) which is equal to T_E^i. Column 4 for early finish time (EFT) is the sum of column (2) and column (3). Then column (6) for latest finish time (LFT) should be filled, which is equal to T_L^j. Column 5 for latest start time (LST) is then obtained by subtracting column (2) from column (6).

After having entered all the four activity times, float calculations are done. Though various equations for the three types of floats have been suggested in this article, the one relating to the activity times and activity duration should be used, so that data from columns (2) to (6) can be directly used.

Column 7 for *total float* is obtained *either* by finding difference between columns (5) and (3), *or* by finding the difference between columns (6) and (4) ; both ways indentical results should be obtained.

Column (8) is for *free float*, which is obtained by finding the difference between EST of succeeding activity and the EFT of the activity under consideration. For example, free float for 1-2 = EST of successor activity (2-5) or (2-7) minus EFT of (1-2) = 10 – 10 = 0. Similarly, F_F for 1-3 = EST of (3-4) or (3-5) minus EFT of (1-3) = 12 – 12 = 0. It should be noted that F_F cannot be greater than F_T.

Column (9) is for *independent float*, which is obtained by subtracting the tail event slack from the free float. Tail event slack can be easily taken from Fig. 8.16, while free float is already available in column (8). For example, for activity 2-5, tail event slack is 8 while free float is also 8 ; hence independent float for activity 2-5 will be zero. However, for activity 2-7, free float is 18 while tail event slack = (18 – 10) = 8. Hence independent float = 18 – 8 = 10.

8.10. CRITICAL ACTIVITIES AND CRITICAL PATH

Out of the various types of floats, *total float* is the most useful. Since the total float is the difference between maximum available time and the activity duration, there are three possibilities :

(*i*) It may have a *negative value,* if the time availability is less than activity duration.

(*ii*) It may have a *zero value* of the time availability is equal to the activity duration.

(*iii*) It may have a *positive value* if *time availability* exceeds the activity duration.

This information about the degree of the total float is very useful as regards the *criticality of the activity*. There may be various sequences of activities, leading the initial event to the final event. These activities can be *classed* on the basis of the degree of the float as under :

(*a*) ***Super Critical Activity***	:	When the float of the activity is negative ; such activity demands very special attention and action.
(*b*) ***Critical Activity***	:	When float is zero : such activity demands above normal attention with no freedom of action.
(*c*) ***Sub-Critical Activity***	:	When the float is positive, demanding normal attention, but allowing some freedom of action.

Negative float results when the activity completion time is more than the available time. Such negative float, though possible, indicates an abnormal situation requiring a decision on how to compress the network, *i.e.* an attempt is to be made by employing more resources so as to make total float zero or positive from the original negative value. Naturally, compression of a network would mean additional cost. This concept of time-cost analysis will be discussed in the next chapter.

The *critical path*, as already defined, is the longest path through the network and time along this path gives the project duration. Critical path joins those activities which are *critical.* Critical path can be easily determined with the help of *total float calculations*. The activities on the critical path are those activities that have total float equal to zero. The activities that control the project duration are the ones that have zero total float and form continuous chain (or path) starting at the first node and ending with the last node.

The critical path for the network of Fig. 8.6 is shown by thick lines in Fig. 8.16, along with the activities start and finish times marked.

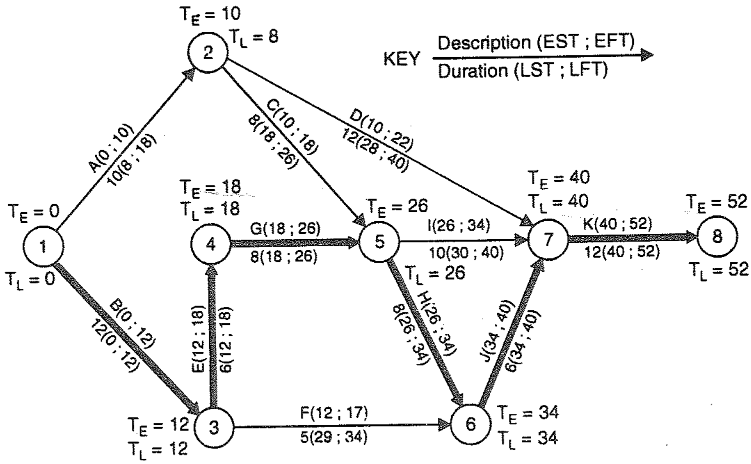

FIG. 8.17. CRITICAL PATH.

From the above, the following points are noteworthy.

1. Critical path starts from the initial event and ends at the end event of the network. All events and activities lying along the critical path are *critical* for the completion of the activities. Thus events 1, 3, 4, 5, 6, 7 and 8 are critical, and activities 1-3, 3-4, 4-5, 5-6, 6-7 and 7-8 are critical.

2. Critical path passes through those events where *slack is* zero. Although, this is *necessary condition* but it is not *sufficient.* This is evident from the fact that though events 3 and 6 are critical but activity 3-6 (connecting events 3 and 6) is *not* critical. In order to identify the critical path, the *float concept* is more useful since it provides both the necessary and sufficient condition for the activity to be critical. *This explains the basic difference in approach for determining critical path via event-oriented slack philosophy (used in PERT) and activity oriented float philosophy.*

3. There can be more than one critical path in a network, and depending upon the total float value, *degree of criticality* can be assigned to a particular path.

4. Non-critical activities have flexibilities in their start time or finish time.

8.11. ILLUSTRATIVE EXAMPLES

Example 8.1. *A building project consists of 10 activities, represented by the network shown below in Fig. 8.18.*

The normal durations required to perform various activities of the above project are given in Table 8.5 below. Compute : (a) event times, (b) activity times and (c) total float. Also, determine the critical path.

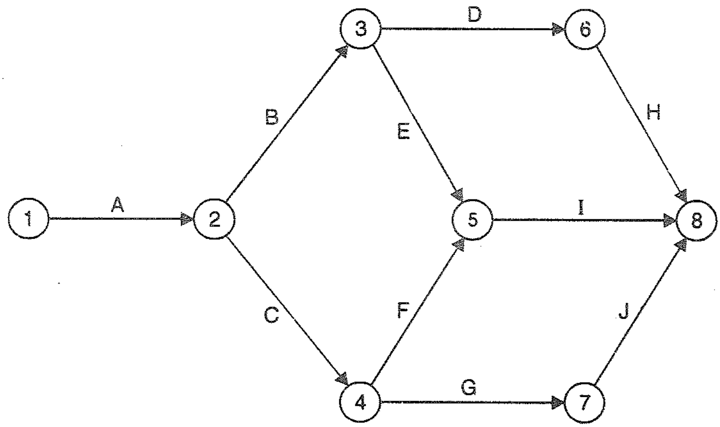

FIG. 8.18

Table 8.5

Activity	*Estimated duration*	*Activity*	*Estimated duration*
A	5	*F*	2
B	2	*G*	3
C	6	*H*	8
D	4	*I*	7
E	4	*J*	2

Solution.

1. Computation of event times

The *earliest event times* (T_E) and latest event occurrence times (T_L) are computed in a tabular form, shown below in Table 8.6.

Table 8.6
Computation of Event Times

Event No.	*Earliest event time (↓)* Predecessor event (i)	t^{ij}	T_E^j	T_E	*Latest event time (↑)* Successor event (j)	t^{ij}	T_L^i	T_L
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
1	—	—	—	0	2	5	0	0
2	1	5	5	5	3 4	2 6	6 5	5
3	2	2	7	7	5 6	4 4	9 8	8
4	2	6	11	11	5 7	2 3	11 15	11
5	3 4	4 2	11 13	13	8	7	13	13
6	3	4	11	11	8	8	12	12
7	4	3	14	14	8	2	18	18
8	5 6 7	7 8 2	20 19 16	20	—	—		20

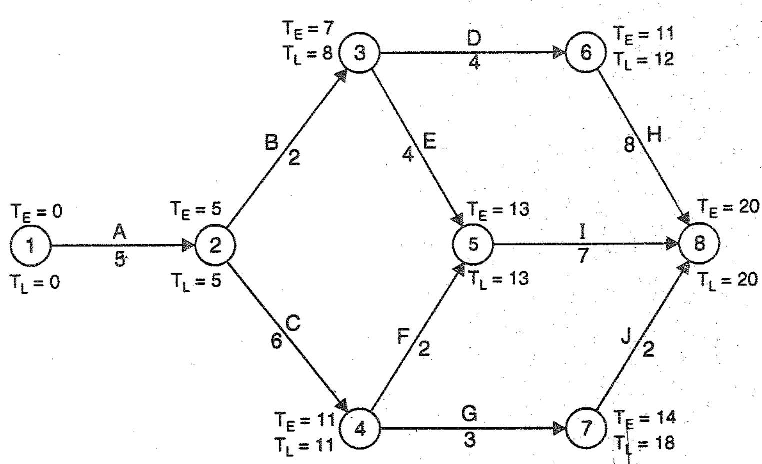

FIG. 8.19

The network, marked with the event times is shown in Fig. 8.19.

2. Computation of activity times and the total float

The earliest start time (EST), earliest finish time (EFT), latest start time (LST) and latest finish time (LFT) of each activity, along with the total float are computed in Table 8.7. The EST of each activity is equal T_E^i, while EFT is equal to $T_E^i + t^{ij}$. Similarly, LFT is equal to T_L^j while LST is equal to $T_L^j - t^{ij}$, as explained earlier. Finally, the total float of each activity is equal to either LST—EST or LFT—EFT.

Table 8.7
Computation of Activity Time and Total Float

Activity (i—j)	*Duration t^{ij}*	*Earliest*		*Latest*		*Total Float F_T*
		Start time (EST)	*Finish time EFT*	*Start time (LST)*	*Finish time LFT*	
(1)	(2)	(3)	(4)	(5)	(6)	(7)
1—2	5	0	5	0	5	0
2—3	2	5	7	6	8	1
2—4	6	5	11	5	11	0
3—5	4	7	11	9	13	2
3—6	4	7	11	8	12	1
4—5	2	11	13	11	13	0
4—7	3	11	14	15	18	4
5—8	7	13	20	13	20	0
6—8	8	11	19	12	20	1
7—8	2	14	16	18	20	4

3. Location of critical path

The activities for which total float is zero, are the critical activities, and these are 1-2, 2-4, 4-5 and 5-8. The critical path is therefore along these activities, starting from event 1 and ending at event 8. The critical path is shown with thick lines in Fig. 8.20, along with the activity times marked on each activity.

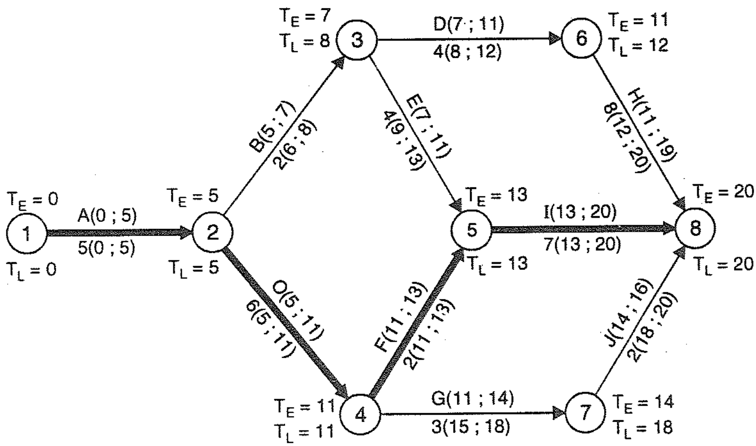

FIG. 8.20. CRITICAL PATH.

Example 8.2. *The network for a certain project is shown in Fig. 8.21, along with the estimated time of completion of each activity marked. Compute the activity times, and total float, free float and independent float for each activity. Locate the critical path on the network.*

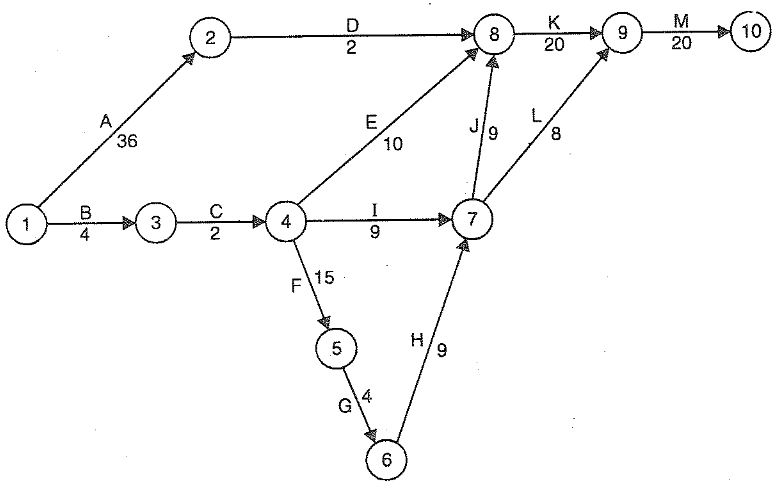

FIG. 8.21

Solution.

1. Computation of event times

The earliest event time (T_E) and latest event occurrence times (T_L) for each activity are shown tabulated in Table 8.8.

Table 8.8
Computation of Event Times

Event No.	Earliest event time (↓)				Latest event time (↑)			
	Predecessor event (i)	t^{ij}	T_E^j	T_E	Successor event (j)	t^{ij}	T_L^i	T_L
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
1	—	—	—	0	2	36	5	0
					3	4	0	
2	1	36	36	36	8	2	41	41
3	1	4	4	4	4	2	4	4
4	3	2	6	6	5	15	6	6
					7	9	25	
					8	10	33	
5	4	15	21	21	6	4	21	21
6	5	4	25	25	7	9	25	25
7	4	9	15	34	8	9	34	34
	6	9	34		9	8	55	
8	2	2	38	43	9	20	43	43
	4	10	16					
	7	9	43					
9	7	8	42	63	10	20	63	63
	8	20	63					
10	9	20	83	83	—			83

2. Activity Times

The earliest start time (EST), earliest finish time (EFT), latest start time (LST) and latest finish time (LFT) for each activity are computed in tabular form as shown in Table 8.9. EST for each activity is equal to T_E^i, while EFT is equal to $T_E^i + t^{ij}$. Similarly, LFT of each activity is equal to T_L^j while LST is equal to $T_L^j - t^{ij}$.

Table 8.9
Computation of Activity Times and Floats

Activity *(i-j)*	*Duration* (t^{ij})	*Earliest*		*Latest*		*Float*		
		Start time EST	*Finish time EFT*	*Start time LST*	*Finish time LFT*	*Total* F_T	*Free* F_F	*Independent* F_{ID}
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
1—2	36	0	36	5	41	5	0	0
1—3	4	0	4	0	4	0	0	0
2—8	2	36	38	41	43	5	5	0
3—4	2	4	6	4	6	0	0	0
4—5	15	6	21	6	21	0	0	0
4—7	9	6	15	25	34	19	19	19
4—8	10	6	16	33	43	27	27	27
5—6	4	21	25	21	25	0	0	0
6—7	9	25	34	25	34	0	0	0
7—8	9	34	43	34	43	0	0	0
7—9	8	34	42	55	63	21	21	21
8—9	20	43	63	43	63	0	0	0
9—10	20	63	83	63	83	0	0	0

3. Calculation of Floats

The total float for each activity is computed taking the difference of LST and EST ; it is shown computed in column 7. The free float is computed by taking the difference of EST of successor activity and EFT of the activity under consideration. It is shown computed in column 8. The independent float is computed by subtracting tail event slack from the free float ; it is shown tabulated in column 9.

4. Critical Path

Activities 1—3, 3—4, 4—5, 5—6, 6—7, 7—8, 8—9 and 9—10, having total float equal to zero, are critical. Hence the path 1—3—4—5—6—7—8—9—10 is the critical path, as marked by thick lines in Fig. 8.22.

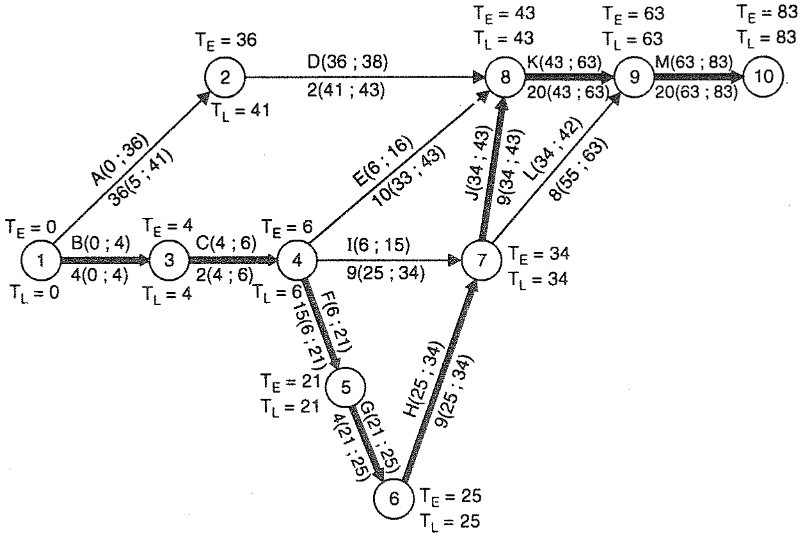

FIG. 8.22

PROBLEMS

1. Define 'earliest event time' and 'latest occurrence event time'. How are these determined ? Explain the tabular form for determining these.
2. What do you understand by 'earliest start time' and 'latest start time' of an activity ? How are these determined ?
3. Define 'latest start time' and 'latest finish time'. How are these determined ?
4. What do you understand by total float ? How is it determined ? What is its importance in network planning ?
5. Differentiate clearly between 'total float', 'free float' and 'independent float'.
6. Define 'free float'. What is its importance ? How is it determined ?
7. What do you understand by independent float ? Show that it can be determined by subtracting the tail event slack from the free float ?
8. Explain the tabular form of doing computations for CPM network elements.
9. What do you understand by critical path ? How is it determined ?
10. The network of a certain project is shown in Fig. 8.23, with the estimated durations of various activities. Determine the following :

    (*a*) Earliest event time and latest event time.

    (*b*) Earliest and latest start and finish times of each activity.

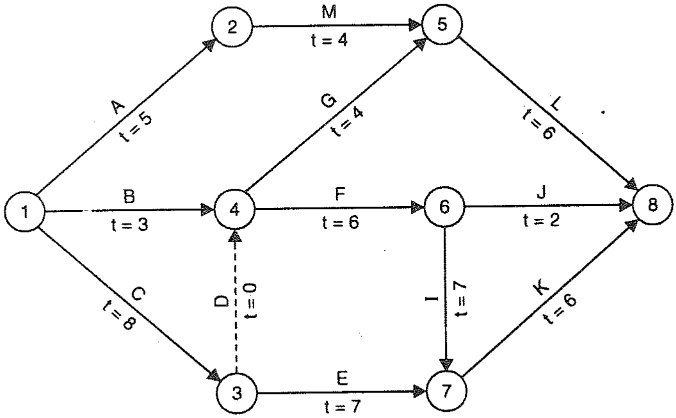

FIG. 8.23

(*c*) Total and free floats for each activity.

(*d*) Critical path for the network.

11. The network shown in Fig. 8.24 has the estimated duration for each activity marked. Determine the total float for each activity and establish the critical path. Also determine free float and independent float for each activity.

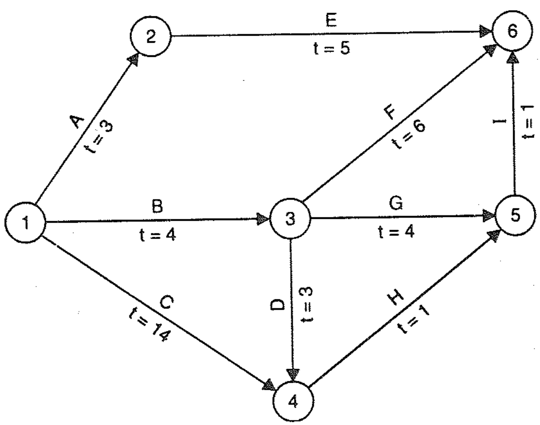

FIG. 8.24