Introduction to Operational Research

NETWORK ANALYSIS

Monika Hidayanti

Department of Mathematics
Universitas Padjadjaran

Chapter 4 Network Analysis (Project Management)

4.1 Introduction (S12)

4.2 Project network (S12)

4.3 Drawing Project Network (S12)

4.4 Finding Critical Path (S12)

4.5 Scheduling (S13)

4.1 Introduction

- Many important optimization problems can best be analysed by means of a graphical or network representation.
- Such as, Transportation, electrical, and communication networks pervade our daily lives.
- Network representations also are widely used for problems in such diverse areas as production, distribution, project planning, facilities location, resource management, and financial planning

- A network representation provides such a powerful visual and conceptual aid for portraying the relationships between the components of systems that it is used in virtually every field of scientific, social, and economic endeavour.

- Many network optimization models actually are special types of linear programming problems. For example, both the transportation problem and the assignment problem discussed in the preceding chapter

The four important kinds of network problems and some basic ideas of how to solve them (without delving into issues of data structures that are so vital to successful large-scale implementations).

Each of the first three problem types:

1) The shortest-path problem
2) The minimum spanning tree problem
3) The maximum flow problem

has a very specific structure that arises frequently in applications.

The fourth type is the **minimum cost flow problem**. It provides a unified approach to many other applications because of its far more general structure.

Terminology of Networks

- A network consists of a set of points and a set of lines connecting certain pairs of the points.
  - The points are called nodes (or vertices);
  - The lines are called arcs (or links or edges or branches)
- The arcs of a network may have a flow of some type through them,
  - If flow through an arc is allowed in only one direction (e.g., a one way street), the arc is said to be a **directed arc**.

- If flow through an arc is allowed in either direction (e.g., a pipeline that can be used to pump fluid in either direction), the arc is said to be an **undirected arc**.

- Network Definitions:

A network consists of a set of nodes (vertices) linked by arcs (edges or branches). The notation for describing a network is (N, A), where N is the set of nodes and A is the set of arcs.

- As an illustration, the network consist of

$N = \{1,2,3,4,5\}$

$A = \{(1,2), (1,3), (2,3), (2,5), (3,4), (3,5), (4,2), (4,5)\}$

The network is,

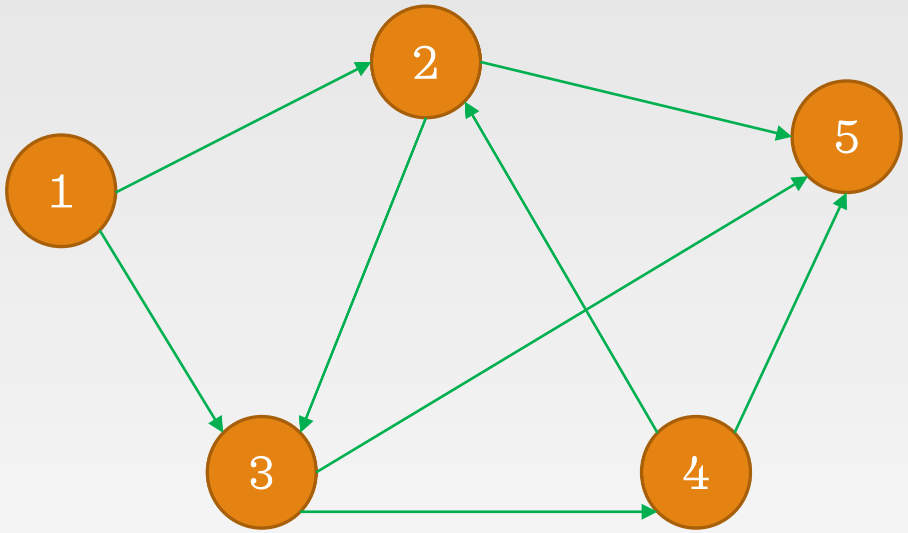


Figure 4.1 Example of *(N,A)* Network

Type of networks:

- **A connected network** is such the previous network which every two distinct nodes are linked by at least one path.
- **A tree** is a *cycle-free* connected network comprised of a *subset* of all the nodes, and
- **A spanning tree** is a tree that links *all* the nodes of the network.

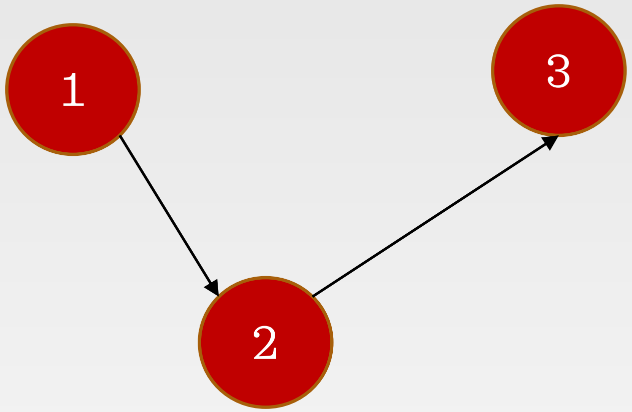


FIGURE 4.2 (A) A TREE

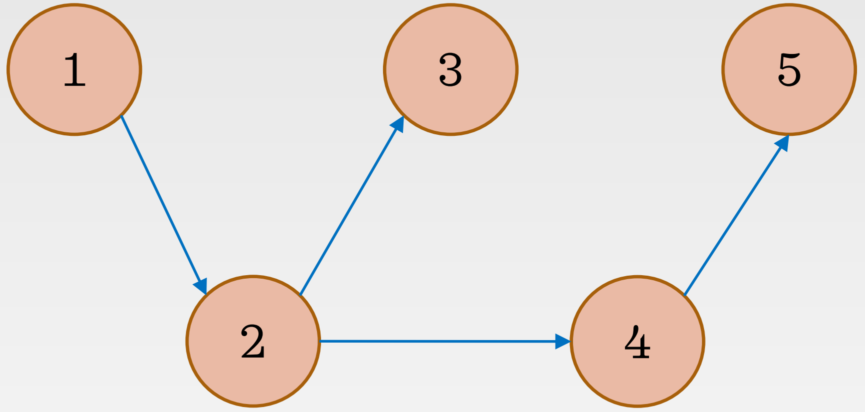


FIGURE 4.2 (B) A SPANNING TREE

In this chapter, we will focus to study critical path analysis/ method (CPA/M) and PERT (Program Evaluation and Review Technique) that emphasise on how to provide an appropriate schedule for a project.

4.2 Project networks

- A project is defined as a collection of interrelated activities with each activity consuming time and resources.
- CPM/A (Critical Path Method/Analysis) and PERT (Program Evaluation and Review Technique) are network-based methods designed to assist in the planning, scheduling, and control of projects.

Figure 4.3 summarises the steps of the techniques:

1) To define the activities of the project, their precedence relationships, and their time requirements.
2) The precedence relationships among the activities are represented by a network.
3) Specific computations to develop the time schedule for the project.

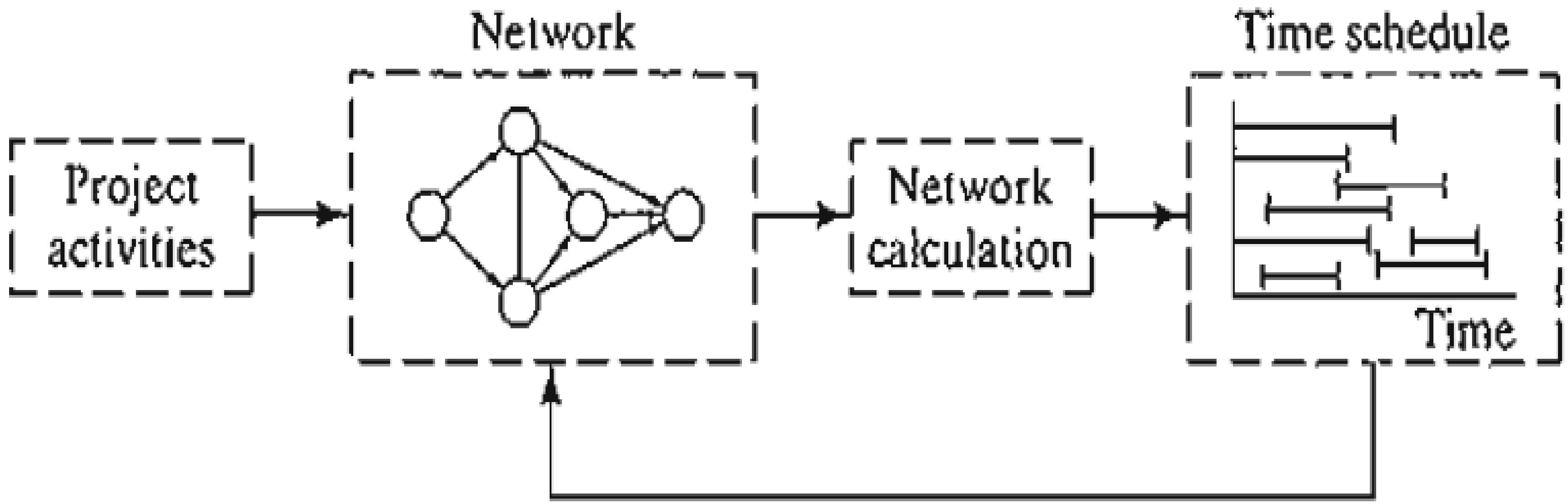

Network
Time schedule
Project activities
Network calculation
Time

Figure 4.3

- The schedule must be revised to reflect the realities on the ground when some of the activities may be expedited or delayed.
- The two techniques, CPM and PERT, which were developed independently, differ in that CPM/A assumes deterministic activity durations and PERT assumes probabilistic durations.

This presentation will start with CPM and then proceed with the details of PERT.

4.2.1Network Representation

- Each activity of the project is represented by an arc pointing in the direction of progress in the project.
- The nodes of the network establish the precedence relationships among the different activities.
- Three rules are available for constructing the network.

1) *Each activity is represented by one, and only one, arc.*
2) *Each activity must be identified by two distinct end nodes.*

3) To maintain the correct precedence relationships, the following questions must be answered as each activity is added to the network:

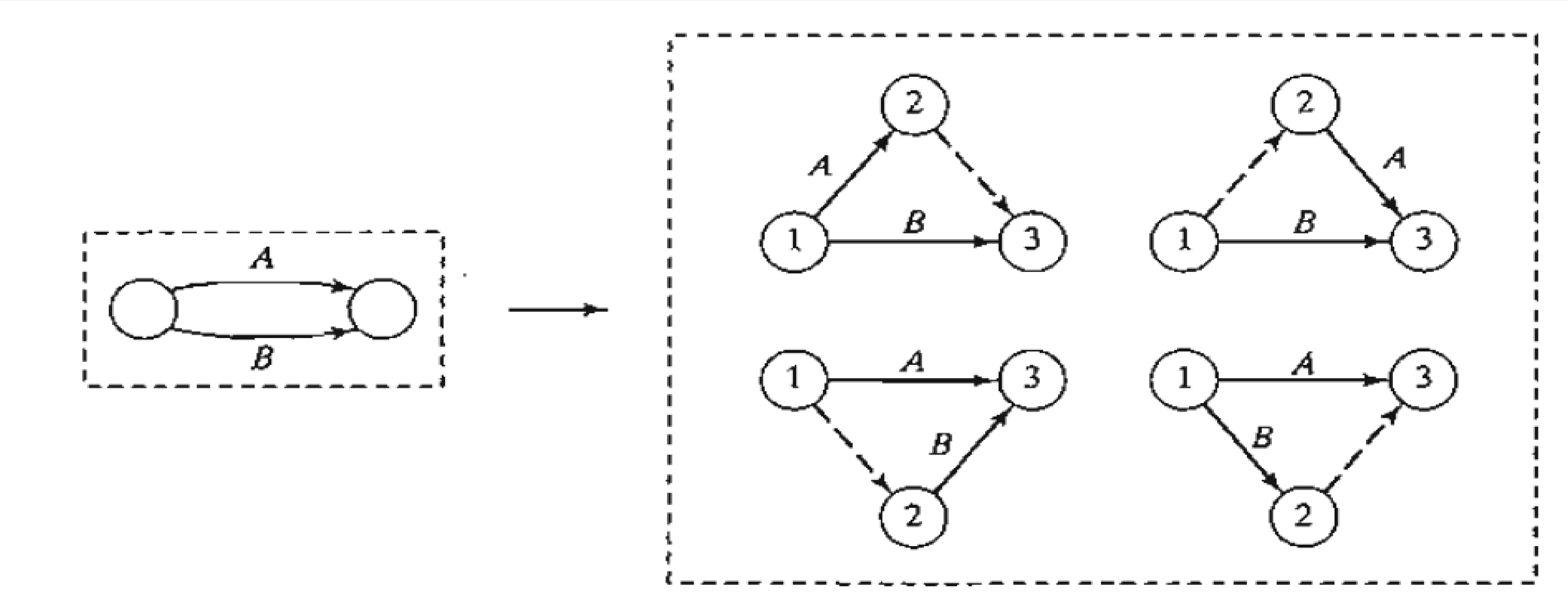
(a) What activities must immediately precede the current activity?

(b) What activities must follow the current activity?

(c) What activities must occur concurrently with the current activity?

- A dummy activity can be used to represent two concurrent activities, *A* and *B*.
- By definition, a dummy activity, which normally is depicted by a dashed are, consumes no time or resources.
- Inserting a dummy activity in one of the four ways shown in Figure 4.4, the concurrence of *A* and *B are* maintained, and provide unique end nodes for the two activities (to satisfy rule 2).

Use of dummy activity to produce unique representation of concurrent activities (Figure 4.4)

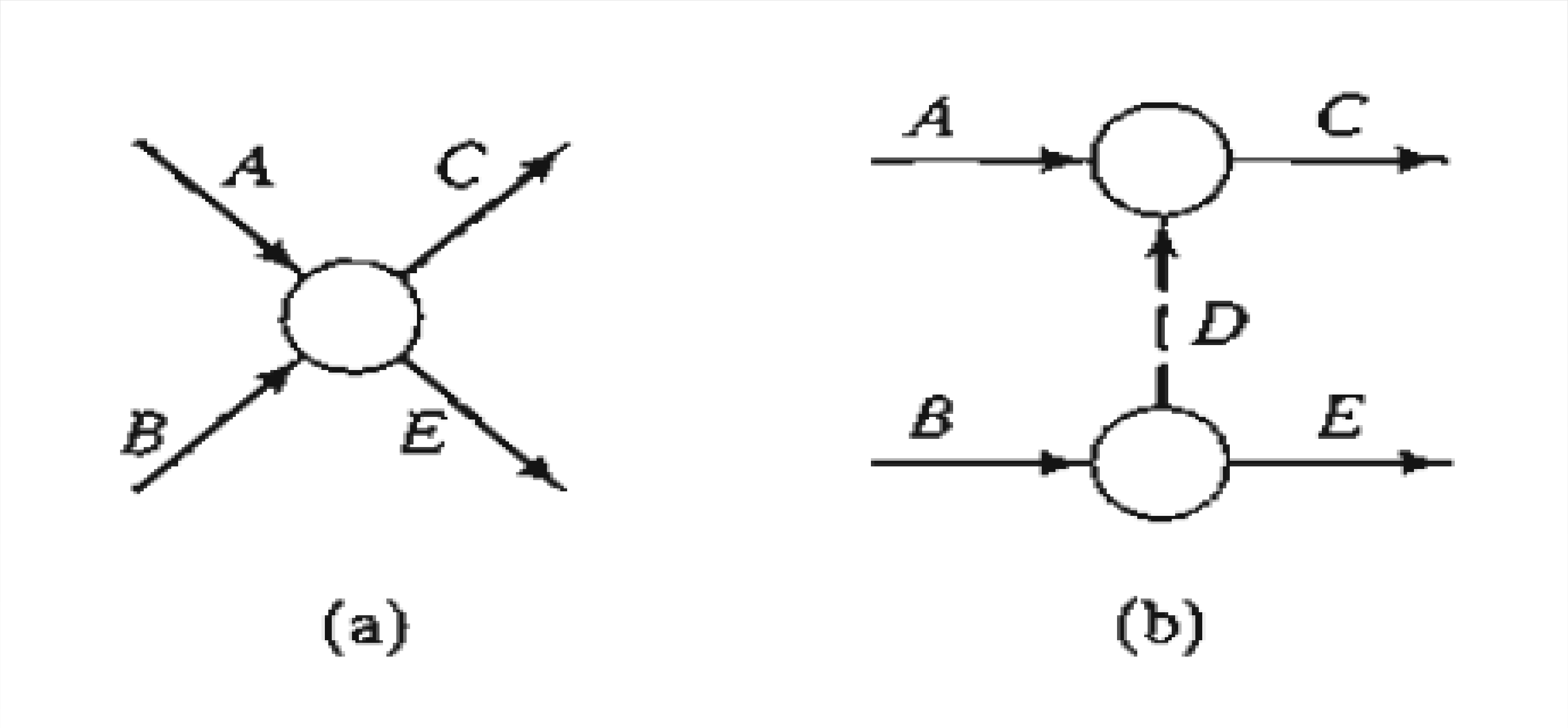


- The answers to Rule 3 may require the use of dummy activities to ensure correct precedence among the activities. For example, consider the following segment of a project:

1. Activity C starts immediately after *A* and *B* have been completed.

2. Activity *E* starts only after *B* has been completed.

Use of dummy activity to ensure correct precedence relationship (Figure 4.5)



- Part (a) of Figure 4.5 shows the incorrect representation of the precedence relationship because it requires both *A* and *B* to be completed before *E* can start.
- In part (b), the use of a dummy activity rectifies the situation.

4.3 Drawing Project Network

- Note that a project network :

1) Possesses a unique initial event, from which all activities which depend on no others start;
2) Possesses a unique terminal event, at which all activities upon which no other depends finish;
3) Has at least one activities starting from each event except terminal event;
4) Has at least one activities finishing from each event except initial event;
5) Has no closed loop

Example 1 : Project network building a House

Activity	Description	Duration (Days)	Activities that must be completed before current activity commences
A	Dig and pour foundation	21	-
B	Make window and door frames	7	-
C	Build walls	10	A,B
D	Build roof	6	C
E	Install plumbing	3	C
F	Install electrical wiring	2	D
G	Plaster walls	4	E,F
H	Decorate	6	G
I	Landscape garden	8	D

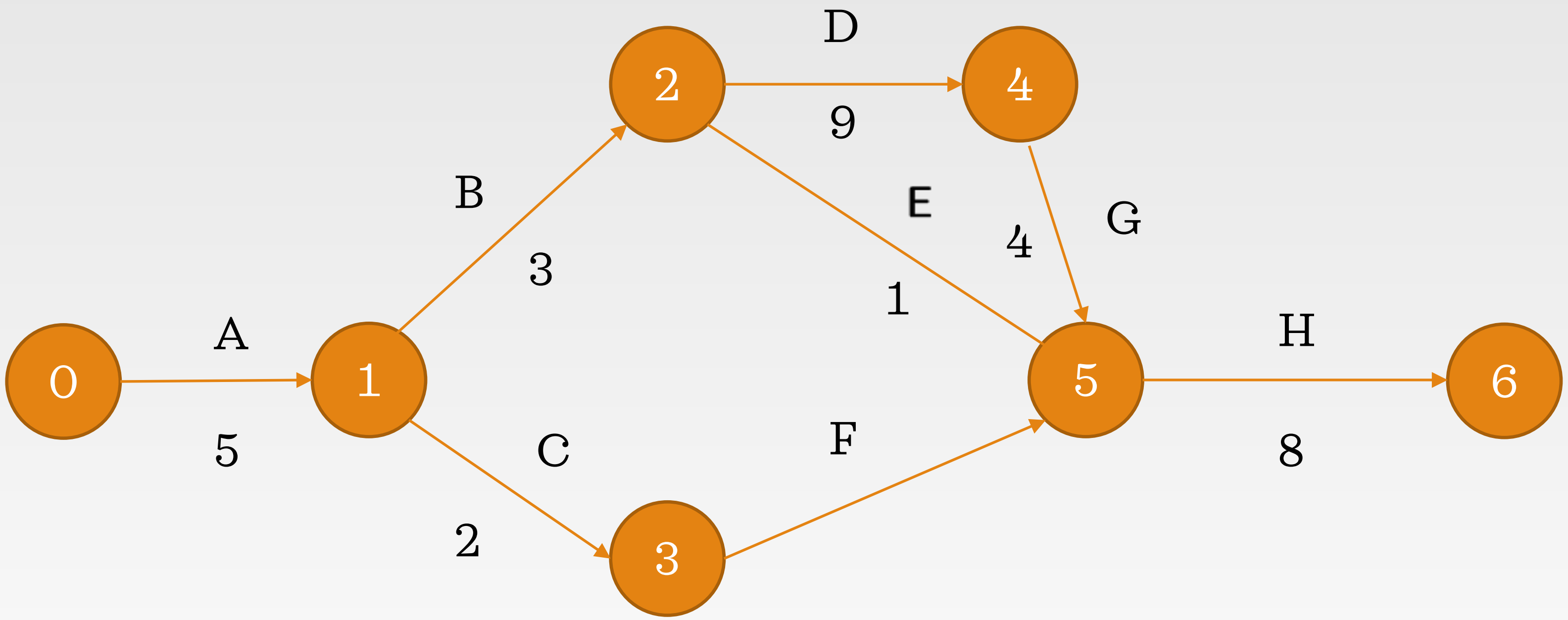
- In general there are two problems:

1) How do we draw the project network given the (duration and) dependencies activities?
2) How do we determine the critical path(s)?

Example 2

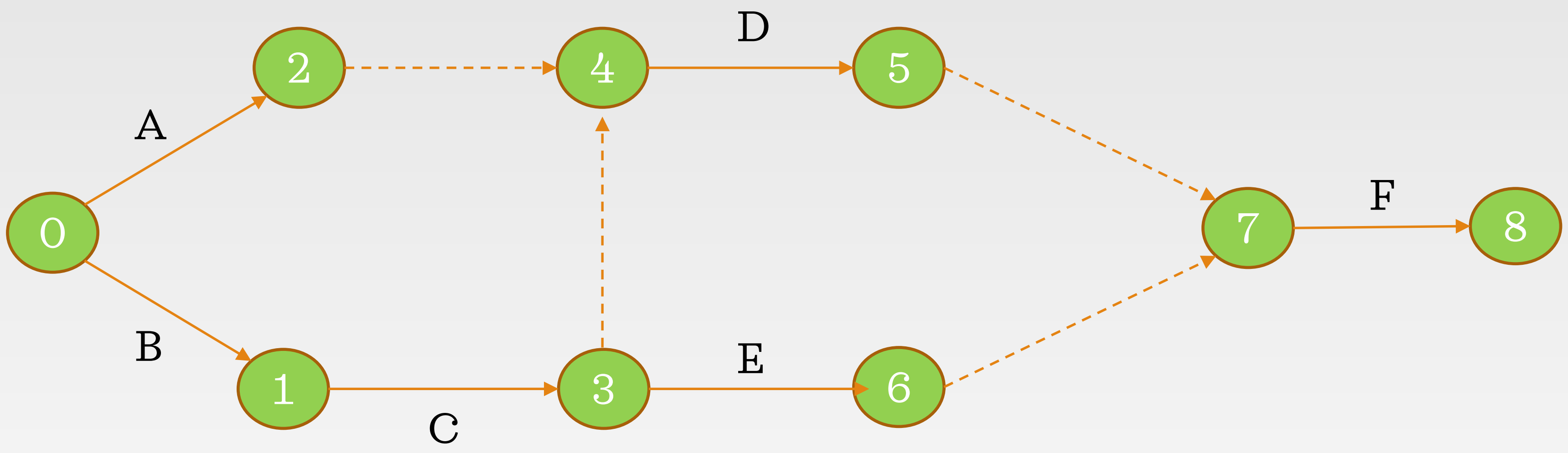
Activity	A	B	C	D	E	F	G	H
Duration	5	3	2	9	1	6	4	8
Depend on	-	A	A	B	B	C	D	E,F,G

Start with the 'from *i*' and 'to *j*' rows blank from the table above. Clearly A can start immediately.
Start event of A is 0, the initial event for the project.


D
2
4
9
B
E
G
3
4
1
A
H
0
1
5
6
5
C
F
8
2
3

Example 3

Activity	A	B	C	D	E	F	Dummies
Duration	-	-	B	A,C	C	D,E	
From *i* to *j*	0 2	0 1	1 3	4 5	3 6	7 8	2 3 5 6 4 4 7 7


A
B
C
D
E
F
0
1
2
3
4
5
6
7
8

4.4 Finding Critical Path

- The end result in CPM is the construction of the time schedule for the project. To achieve this objective conveniently, we carry out special computations that produce the following information:

1) Total duration needed to complete the project.
2) Classification of the activities of the project as *critical* and *noncritical*.

- Critical Path Method (CPM) is a technique for solving a class of scheduling problem. The objective of CPA is to schedule the component activities of a complex project, so that it may completes as soon as possible.

- An activity is said to be critical if there is no "leeway" in determining its start and finish times.
- A noncritical activity allows some scheduling slack, so that the start time of the activity can be advanced or delayed within limits without affecting the completion date of the entire project.
- To carry out the necessary computations, we define an event as a point in time at which activities are terminated and others are started.

In terms of the network, an event corresponds to a node. Define

$\square_j$ = Earliest occurrence time of event j

Δ_j = Latest occurrence time of event j

D_{ij} = Duration of activity (i, j)

The definitions of the *earliest* and *latest* occurrences of event j are specified relative to the start and completion dates of the entire project.

The critical path calculations involve two passes:

1) The forward pass determines the *earliest* occurrence times of the events, and
2) The backward pass calculates their *latest* occurrence times.

Forward Pass (Earliest Occurrence times)

The computations start at node 1 and advance recursively to end node n.

Initial Step

Set $\square_1 = 0$ to indicate that the project starts at time 0.

General Step *j*

Given that nodes *p, q, ...,* and *v* are linked *directly* to node j by incoming activities *(p, n, (q, j), ...,* and *(v,* j) and that the earliest occurrence times of events (nodes) *p, q, ... ,* and *v* have already been computed, then the earliest occurrence time of event *j*

is computed as $$\square_j = \max\{\square_p + D_{pj}, \square_q + D_{qj}, \ldots, \square_v + D_{vj}\}$$

The forward pass is complete when $\square_n$ at node n has been computed. By the definition, q represents the longest path (duration) to node j.

Backward Pass (Latest Occurrence Times, Δ). Following the completion of the forward pass, the backward pass computations start at node n and end at node 1.

Initial Step. Set $\Delta_n = \square_n$ to indicate that the earliest and latest occurrences of the last node of the project are the same.

General Step *j*. Given that nodes $p, q, \ldots,$ and v are linked *directly* to node j by *outgoing* activities $(j, p), (j, q), \ldots,$ and (j, v) and that the latest occurrence times of nodes $p, q, \ldots,$ and v have already been computed, the latest occurrence time of node j is computed as

$$\Delta_j = \min\{\Delta_p - D_{jp}, \Delta_q - D_{jq}, \ldots, \Delta_v - D_{jv}\}$$

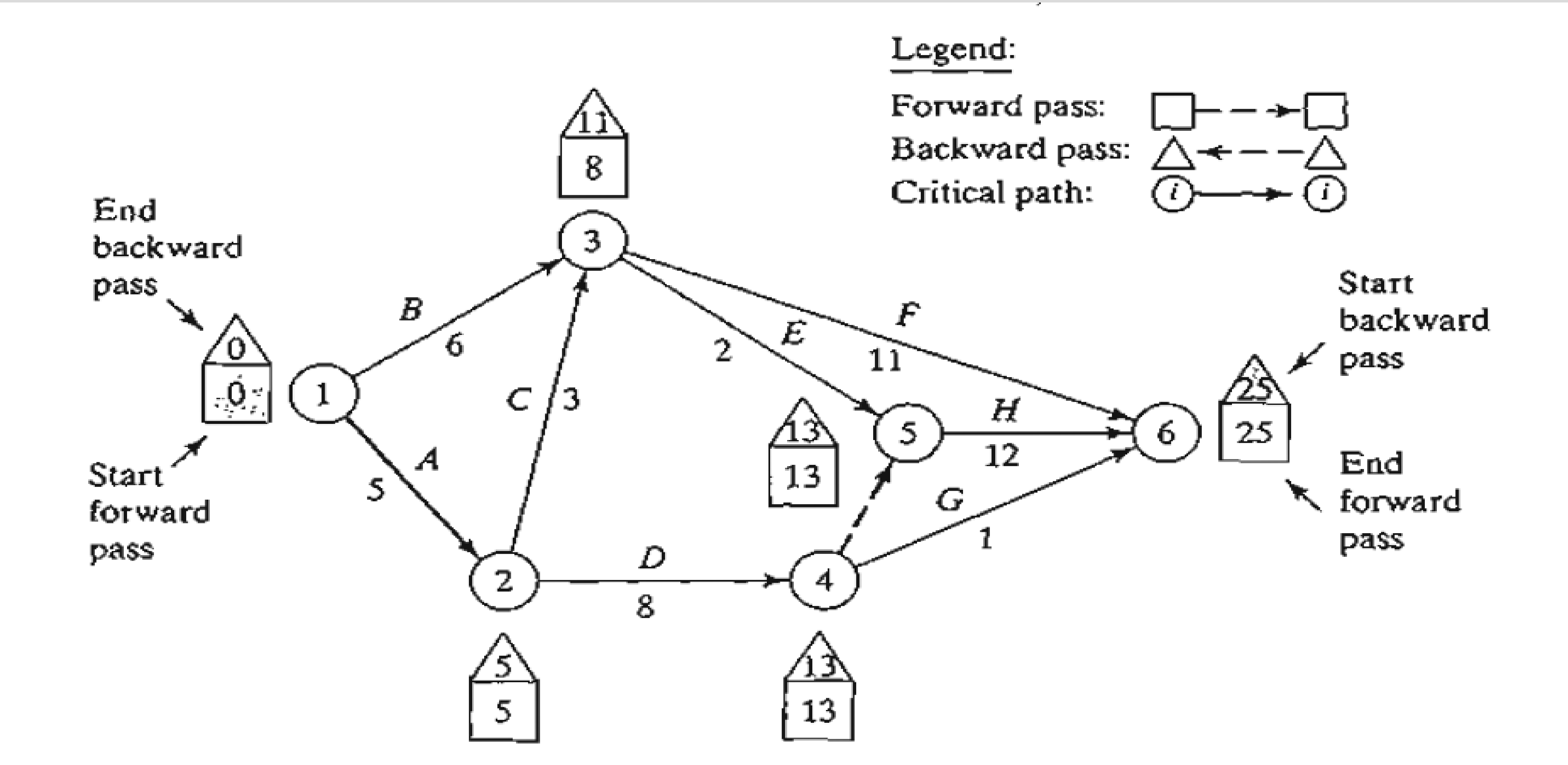
The backward pass is complete when Δ_1 at node 1 is computed. At this point, $\Delta_1 = \square_1 (= 0)$.

Based on the preceding calculations, an activity (i, j) will be *critical* if it satisfies three conditions.

1. $\Delta_i = \square_i$
2. $\Delta_j = \square_j$
3. $\Delta_j - \Delta_i = \square_j - \square_i = D_{ij}$

- The three conditions state that the earliest and latest occurrence times of end nodes i and j are equal and the duration D_{ij} fits "tightly" in the specified time span.
- An activity that does not satisfy all three conditions is thus *noncritical.*
- By the definition, the critical activities of a network must constitute an uninterrupted path that spans the entire network from start to finish.

Example 4 : consider the network below



Determine the critical path for the project network above where all the durations are in days.

Forward Pass

Node 1. Set $\square_1 = 0$

Node 2. $\square_2 = \square_1 + D_{12} = 0 + 5 = 5$

Node 3. $\square_3 = \max\{\square_1 + D_{13}, \square_2 + D_{23}\} = \max\{0 + 6, 5 + 3\} = 8$

Node 4. $\square_4 = \square_2 + D_{24} = 5 + 8 = 13$

Node 5. $\square_5 = \max\{\square_3 + D_{35}, \square_4 + D_{45}\} = \max\{8 + 2, 13 + 0\} = 13$

Node 6. $\square_6 = \max\{\square_3 + D_{36}, \square_4 + D_{46}, \square_5 + D_{56}\}$

$$= \max\{8 + 11, 13 + 1, 13 + 12\} = 25$$

The computations show that the project can be completed in 25 days.

Backward Pass

Node 6. Set $\Delta_6 = \square_6 = 25$

Node 5. $\Delta_5 = \Delta_6 - D_{56} = 25 - 12 = 13$

Node 4. $\Delta_4 = \min\{\Delta_6 - D_{46}, \Delta_5 - D_{45}\} = \min\{25 - 1, 13 - 0\} = 13$

Node 3. $\Delta_3 = \min\{\Delta_6 - D_{36}, \Delta_5 - D_{35}\} = \min\{25 - 11, 13 - 2\} = 11$

Node 2. $\Delta_2 = \min\{\Delta_4 - D_{24}, \Delta_3 - D_{23}\} = \min\{13 - 8, 11 - 3\} = 5$

Node 1. $\Delta_1 = \min\{\Delta_3 - D_{13}, \Delta_2 - D_2\} = \min\{11 - 6, 5 - 5\} = 0$

Correct computations will always end with $\Delta_1 = 0$.

The forward and backward pass computations can be made directly on the network as shown in Figure at example 4.

Applying the rules for determining the critical activities, the critical path is 1 → 2 → 3 & 4 → 5, which, as should be expected, spans the network from start (node 1) to finish (node 6).

The sum of the durations of the critical activities [(1,2), (2, 4), (4, 5), and (5, 6)] equals the duration of the project (= 25 days).

Observe that $(\square_6 - \square_4 \neq D_{46})$ activity (4,6) satisfies the first two conditions for a critical activity $(\Delta_4 = \square_4 = 13 \text{ and } \Delta_5 = \square_5 = 25)$ but not the third. Hence, the activity is noncritical.

4.5 Scheduling

We recognize that for an activity (i, j), $\square_i$ *represents* the *earliest start time*, and Δ_j represents the *latest completion*. This means that the interval $(\square_i, \Delta_j)$ delineates the (maximum) span during which activity (i, j) may be scheduled without delaying the entire project.

Construction of Preliminary Schedule.

The method for constructing a preliminary schedule is illustrated by an example.

We can get a preliminary time schedule for the different activities of the project by delineating their respective time spans as shown in Figure 4.9.

Two observations are in order.

1) The critical activities (shown by solid lines) must be stacked one right after the other to ensure that the project is completed within its specified 25-day duration.

2) The noncritical activities (shown by dashed lines) have time spans that are larger than their respective durations, thus allowing slack (or "leeway") in scheduling them within their allotted time intervals.

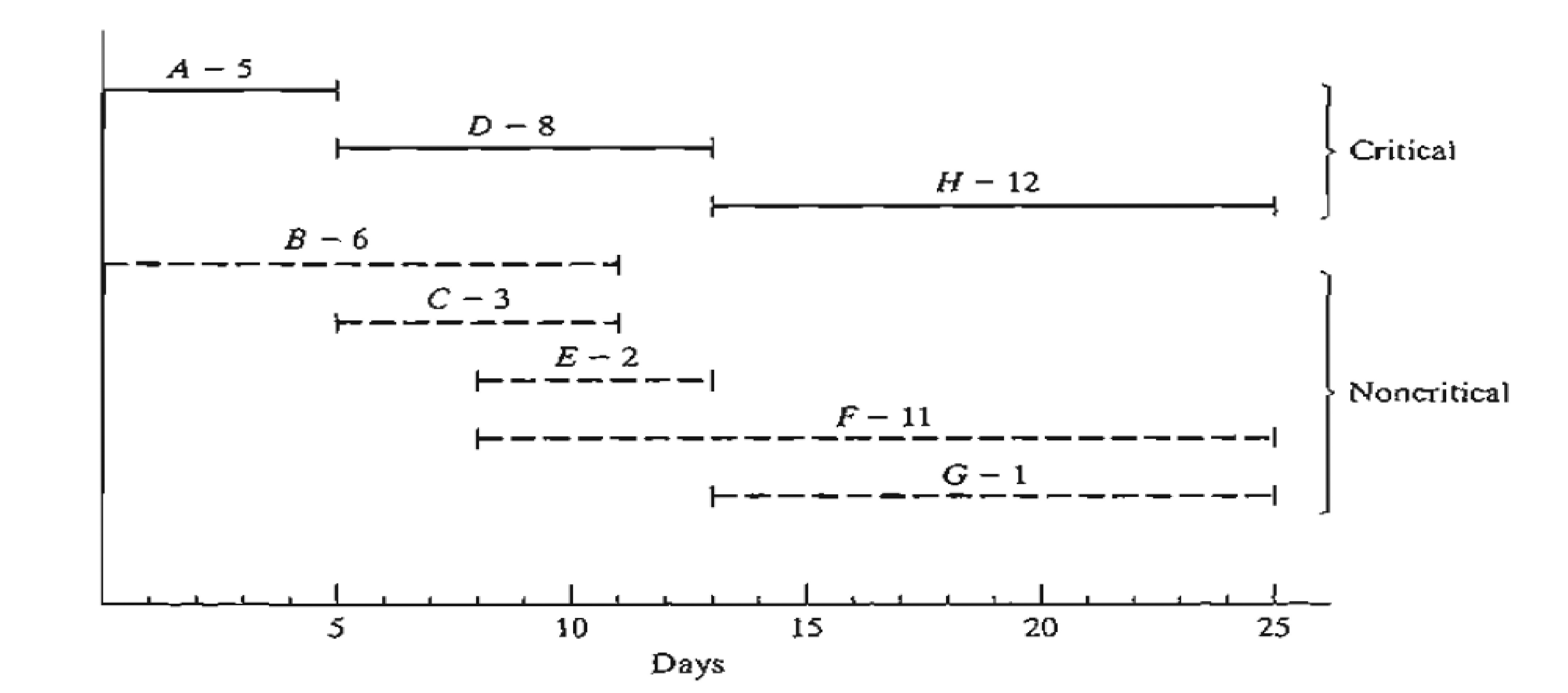


Figure 4.9 Preliminary schedule for the project of Example 4

Determination of the Floats

- Floats are the slack times available within the allotted span of the noncritical activity. The most common are the total float and the free float

- Figure 4.10 gives a convenient summary for computing the **total float** $(TF_{ij})$and the **free float** (FF_{ij}) for an activity (i, j). The total float is the excess of the time span defined from the *earliest* occurrence of event i to the *latest* occurrence of event j over the duration (i, j) . That is,

$$TF_{ij} = \Delta_j - \square_i - D_{ij}$$

The free float is the excess of the time span defined from the *earliest* occurrence of event i to the *earliest* occurrence of event j over the duration of (i, j) that is,

$$FF_{ij} = \square_j - \square_i - D_{ij}$$

By definition, $FF_{ij} \leq TF_{ij}$.

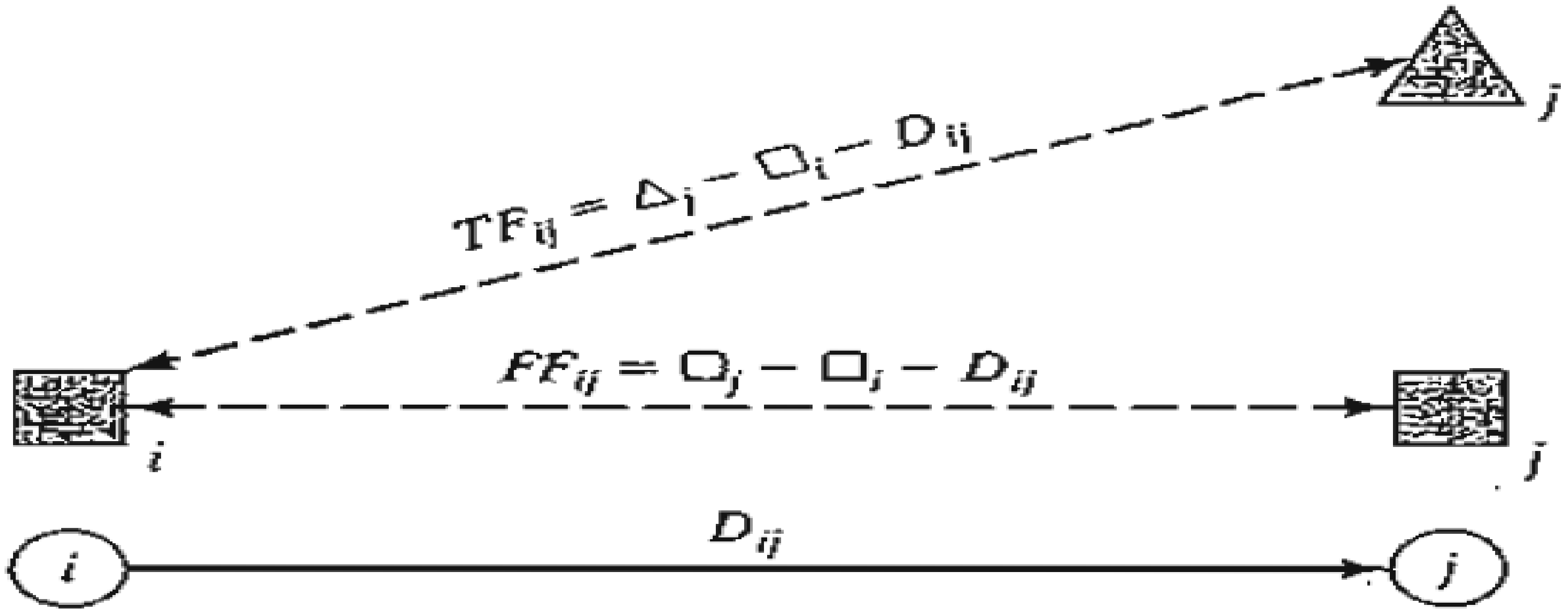


Figure 4.10 Computation of total and free floats

Red-Flagging Rule. *For a noncritical activity (i, j)*

(a) *If $FF_{ij} = TF_{ij}$, then the activity can be scheduled anywhere within its $(\square_i, \Delta_j)$ span without causing schedule conflict.*

(b) *If $FF_{ij} < TF_{ij}$, then the start of the activity can be delayed by at most FF_{ij} relative to its earliest start time $(\square_i)$ without causing schedule conflict. Any delay larger than FF_{ij} (but not more than TF_{ij}) must be coupled with an equal delay relative to $\square_j$ in the start time of all the activities leaving node j.*

The implication of the rule is that a noncritical activity (i, j) will be red-flagged if its $FF_{ij} < TF_{ij}$. This red flag is important only if we decide to delay the start of the activity past its earliest start time, $\square_i$, in which case we must pay attention to the start times of the activities leaving node j to avoid schedule conflicts.

Example 5

Compute the floats for the noncritical activities of the network in Example 4 and discuss their use in finalising a schedule for the project.

The following table summarizes the computations of the total and free floats. It is more convenient to do the calculations directly on the network using the procedure in example 4.

Noncritical activity	Duration	Total float (TF)	Free float (FF)
$B(1,3)$	6	$11 - 0 - 6 = 5$	$8 - 0 - 6 = 2$
$C(2,3)$	3	$11 - 5 - 3 = 3$	$8 - 5 - 3 = 0$
$E(3,5)$	2	$13 - 8 - 2 = 3$	$13 - 8 - 2 = 3$
$F(3,6)$	11	$25 - 8 - 11 = 6$	$25 - 8 - 11 = 6$
$G(4,6)$	1	$25 - 13 - 1 = 11$	$25 - 13 - 1 = 11$

- The computations red-flag activities B and C because their $FF < TF$. The remaining activities *(E, F,* and G) have $FF = TF$, and hence may be scheduled anywhere between their earliest start and latest completion times.
- To investigate the significance of the red-flagged activities consider activity *B*. Because its TF =5 days, this activity can start as early as time 0 or as late as time 5.

- Because its FF = 2 days, starting B anywhere between time 0 and time 2 will have no effect on the succeeding activities E and F.

- Activity B must start at time $2 + d (\leq 5)$, then the start times of the immediately succeeding activities E and F must be pushed forward past their earliest start time (= 8) by at least d. In this manner, the precedence relationship between B and its successors E and F is preserved.

- Turning to red-flagged activity C, we note that its $FF = 0$. This means that *any* delay in starting C past its earliest start time (= 5) must be coupled with at least an equal delay in the start of its successor activities E and F.