Time Planning and Control

Activity on Node Network and Precedence Diagramming

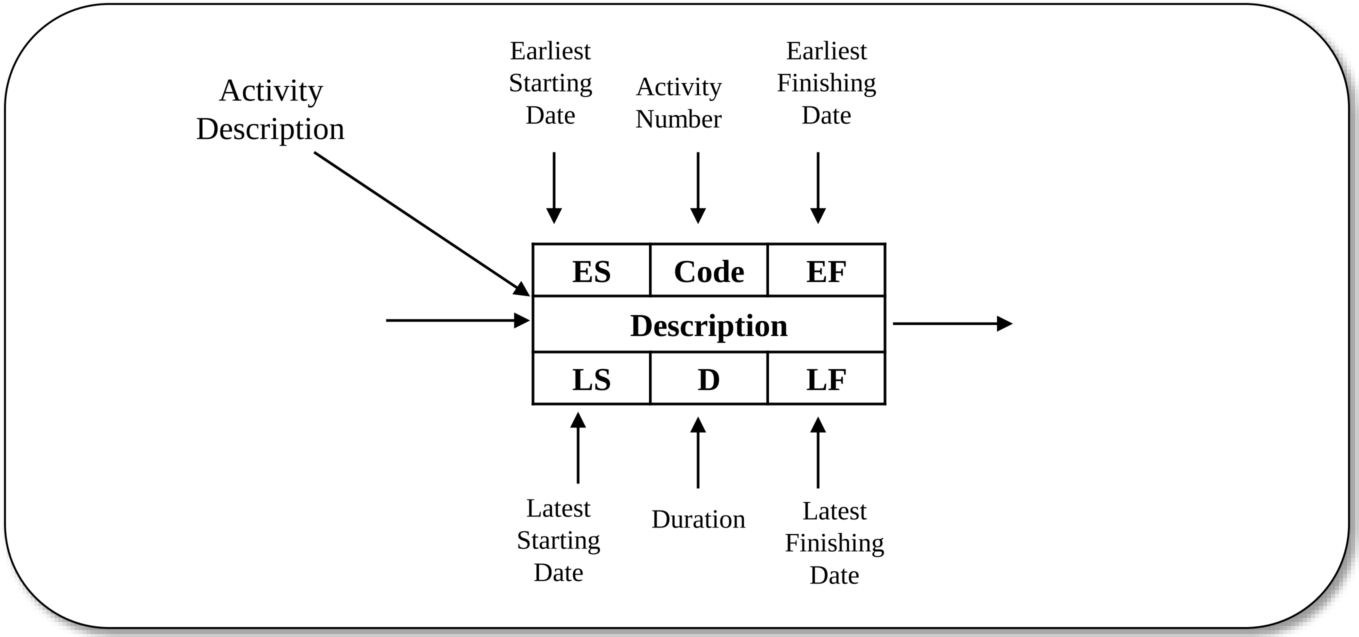
Processes of Time Planning and Control

1. Visualize and define the **<u>activities</u>**.
2. Sequence the activities (**<u>Job Logic</u>**).
3. Estimate the **<u>activity duration</u>**.
4. **<u>Schedule</u>** of the project or phase.
5. Allocate and balance **<u>resources</u>**.
6. Compare target, planned and actual dates and **<u>update</u>** as necessary.
7. **<u>Control</u>** the time schedule with respect to **<u>changes</u>**.

Activity on Node Notation

- Each time-consuming activity is portrayed by a **rectangular figure**.
- The dependencies between activities are indicated by **dependency lines (arrows)** going form one activity to another.
- Each **activity duration** in terms of working days is shown in the lower, central part of the activity box.
- The principal advantage of the activity on node network is that it **eliminates the need for dummies**.

Activity Box



The left side of the activity box (node) is the start side, while the right side is the finish (end) side.

Activity on Node Network

- Each activity in the network must be preceded either by the start of the project or by the completion of a previous activity.

- Each path through the network must be **continuous** with no gaps, discontinuities, or dangling activities.

- All activities must have at least one activity following, except the activity that terminates the project.

- Each activity should have a unique numerical designation (activity code). Activity code is shown in the upper, central part of the activity box, with the numbering proceeding generally from project start to finish.

Network Format

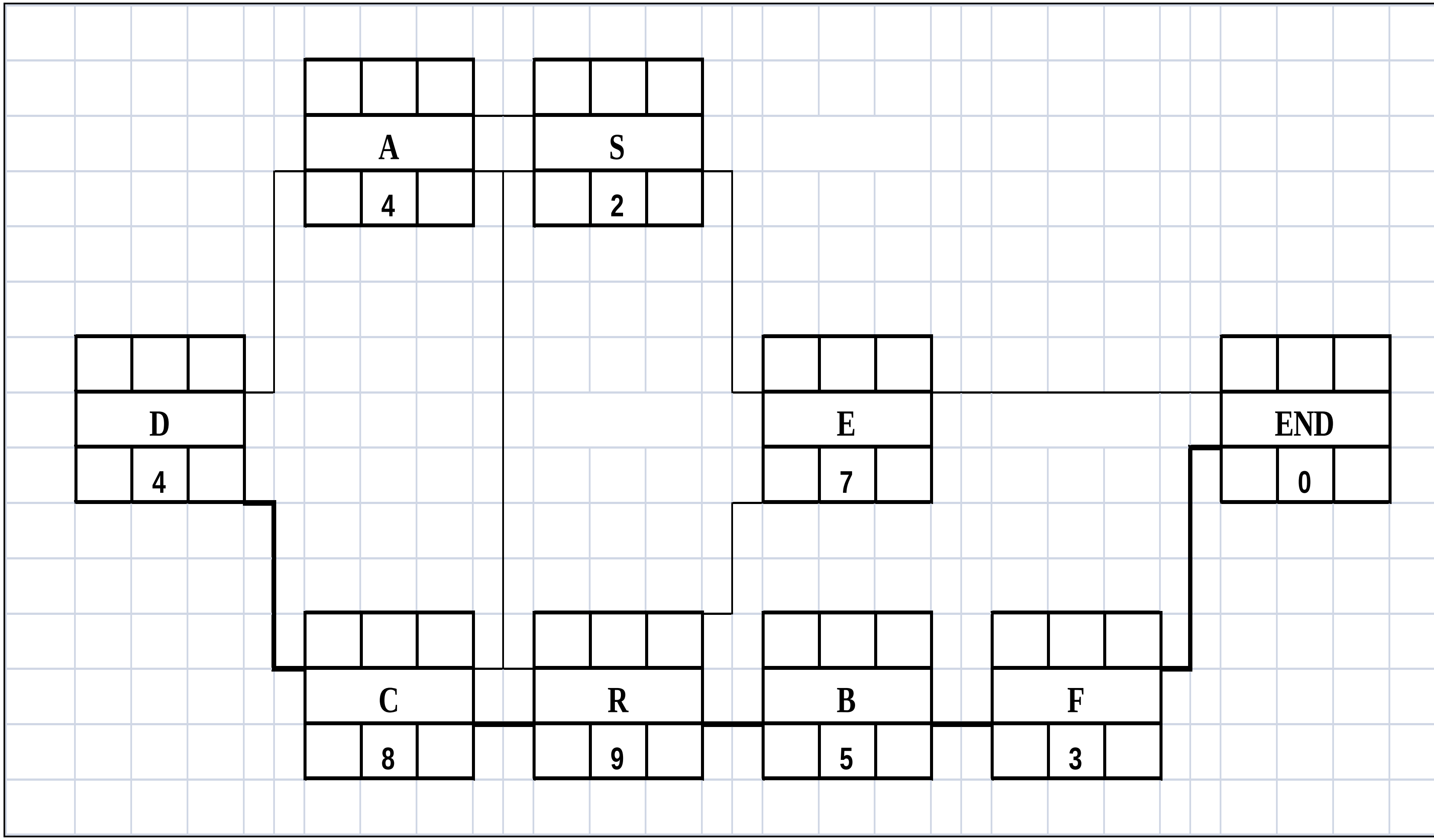
- A **horizontal diagram** format is the standard format.
- The general developing of a network is from start to finish, from project beginning on the left to project completion on the right.
- The sequential relationship of one activity to another is shown by the dependency lines between them.
- The length of the lines between activities has no significance.
- Arrowheads are not always shown on the dependency lines because of the obvious left to right flow of time.
- Dependency lines that go backward from one activity to another (**looping**) should not be used.
- **Crossovers** occur when one dependency line must cross over another to satisfy job logic.

Example

The activity list shown below represents the activities, the job logic and the activities' durations of a small project. Draw an activity on node network to represent the project.

Activity	Depends on	Duration (days)
A	D	4
B	R	5
C	D	8
E	R, S, A	7
F	B, C	3
D	None	4
S	A, C	2
R	A, C	9

Example



3/9/2023 10:31 PM

Network Computations

The purpose of network computations is to determine:

- The overall project completion time and
- The time brackets within which each activity must be accomplished (Activity Times).

In activity on node network, all of the numbers associated with an activity are incorporated in the one node symbol for the activity, whereas the arrow symbols contain each activity's data in the predecessor and successor nodes, as well as on the arrow itself or in a table.

ES	Activity code	EF
Activity Description		
LS	Duration	LF

EARLY ACTIVITY TIMES

1. The "Early Start" (ES) or "Earliest Start" of an activity is the earliest time that the activity can possibly start allowing for the time required to complete the preceding activities.

2. The "Early Finish" (EF) or "Earliest Finish" of an activity is the earliest possible time that it can be completed and is determined by adding that activity's duration to its early start time.

COMPUTATIONS OF EARLY ACTIVITY TIMES

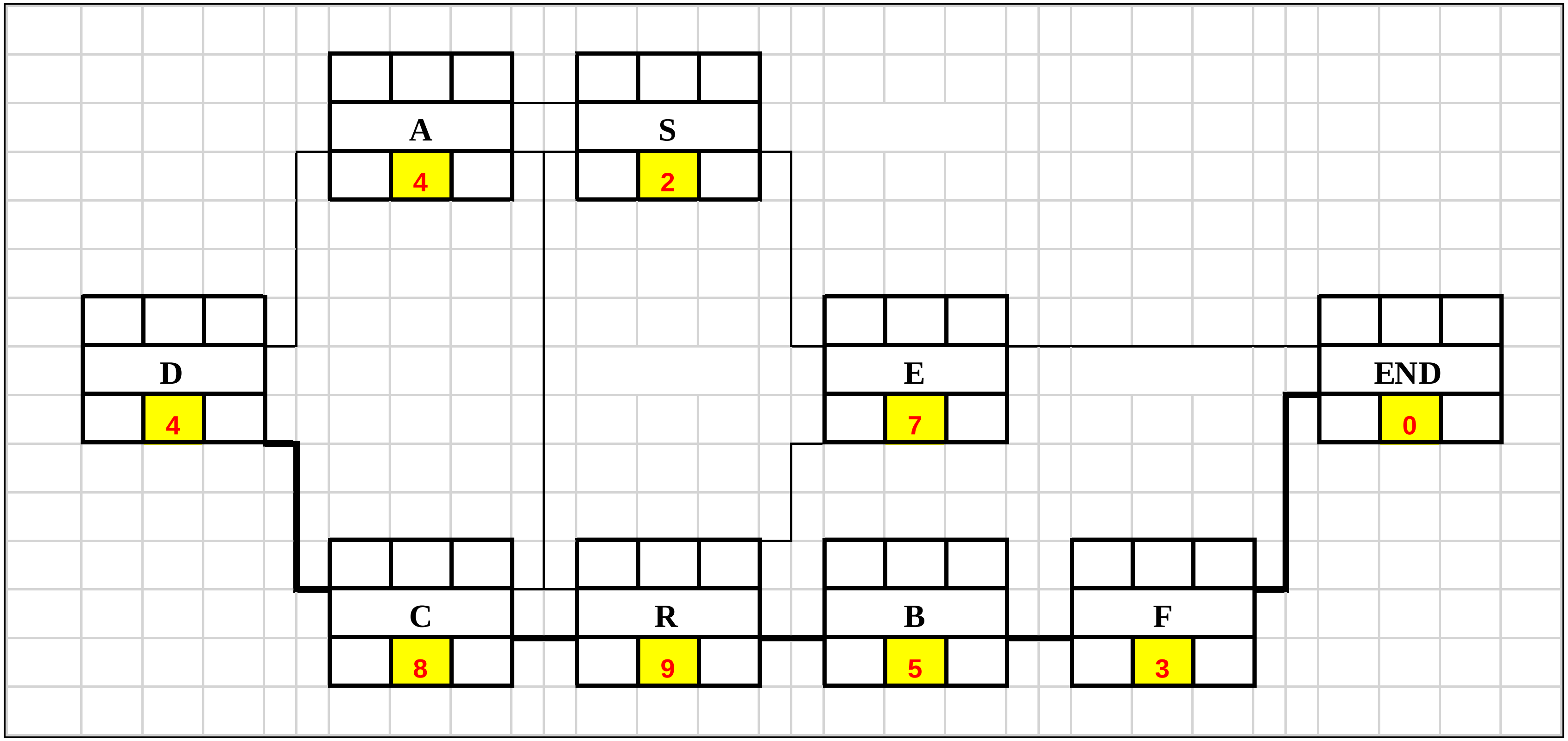
- **Direction**: Proceed from project start to project finish, from **left to right**.
- **Name**: This process is called the "**forward pass**".
- **Assumption**: every activity will start as early as possible. That is to say, each activity will start just as soon as the last of its predecessors is finished.
- The **ES** value of each activity is determined first.
- The **EF** time is obtained by adding the activity duration to the ES time.

$$EF = ES + D$$

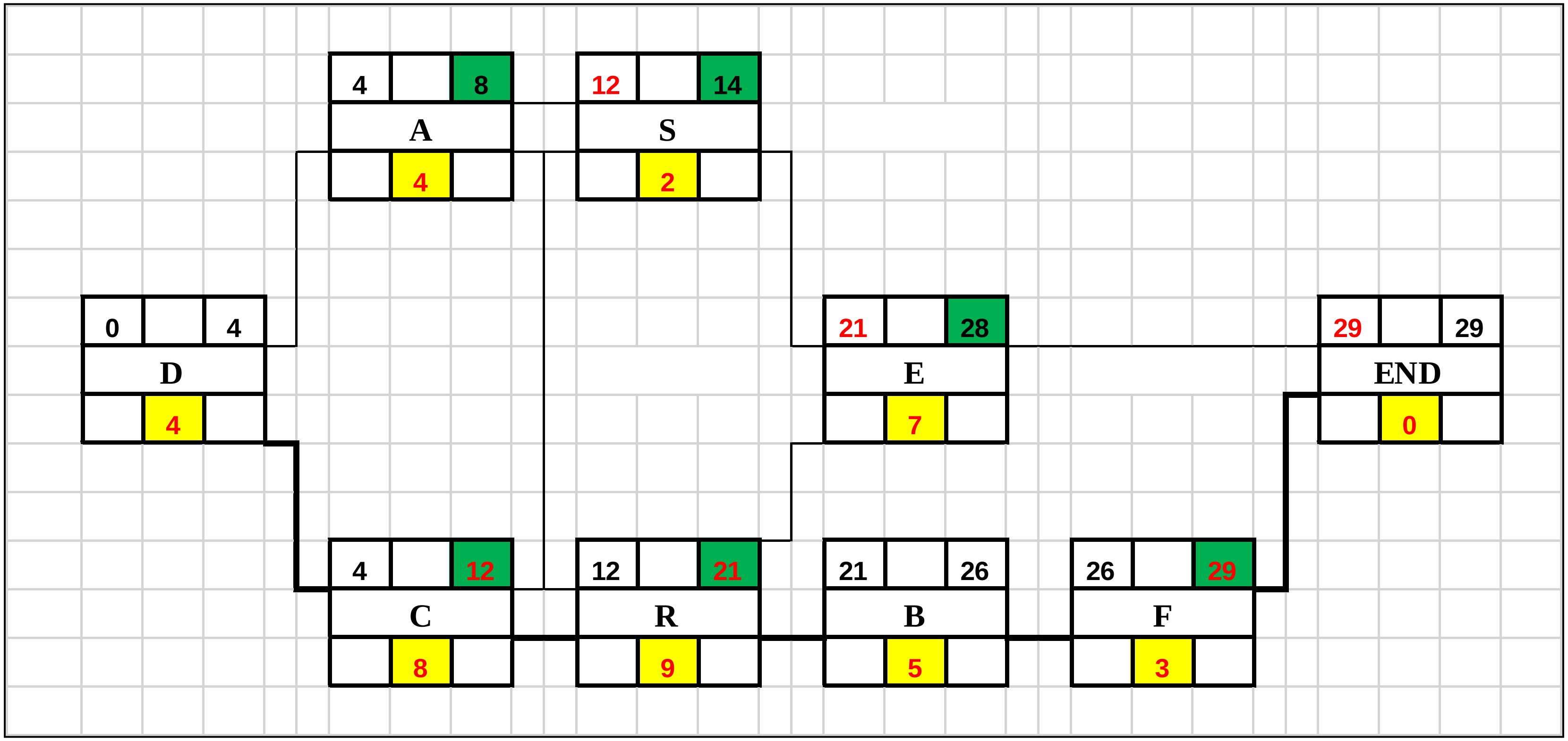
- In case of merge activities the earliest possible start time is equal to the latest (or **largest**) of the EF values of the immediately preceding activities.

Example

Calculate the early activity times (ES and EF) and determine project time.



Example



4
8
A
4
12
14
S
2
0
4
D
4
21
28
E
7
29
29
END
0
4
12
C
8
12
21
R
9
21
26
B
5
26
29
F
3

■ LATE ACTIVITY TIMES

3. The "late finish" (LF) or "Latest Finish" of an activity is the very latest that it can finish and allow the entire project to be completed by a designated time or date.

4. The “late start” (LS) or "Latest Start" of an activity is the latest possible time that it can be started if the project target completion date is to be met and is obtained by subtracting the activity's duration from its latest finish time.

COMPUTATIONS OF LATE ACTIVITY TIMES

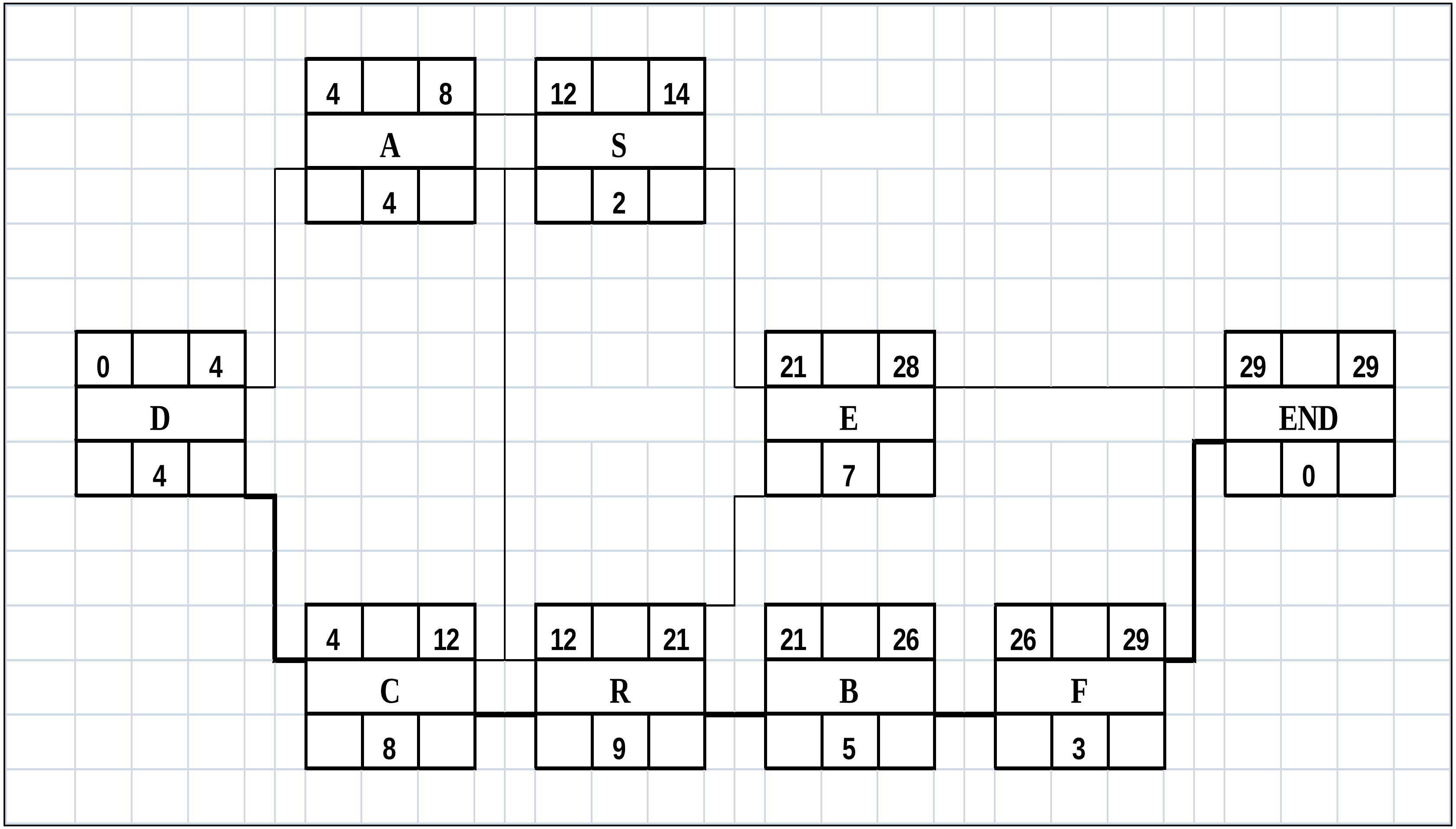
- **Direction**: Proceed from project end to project start, from **right to left**.
- **Name**: This process is called the "**backward pass**".
- **Assumption**: Each activity finishes as late as possible without delaying project completion.
- The **LF** value of each activity is obtained first and is entered into the lower right portion of the activity box.
- The **LS** is obtained by subtracting the activity duration from the LF value.

$$LS = LF - D$$

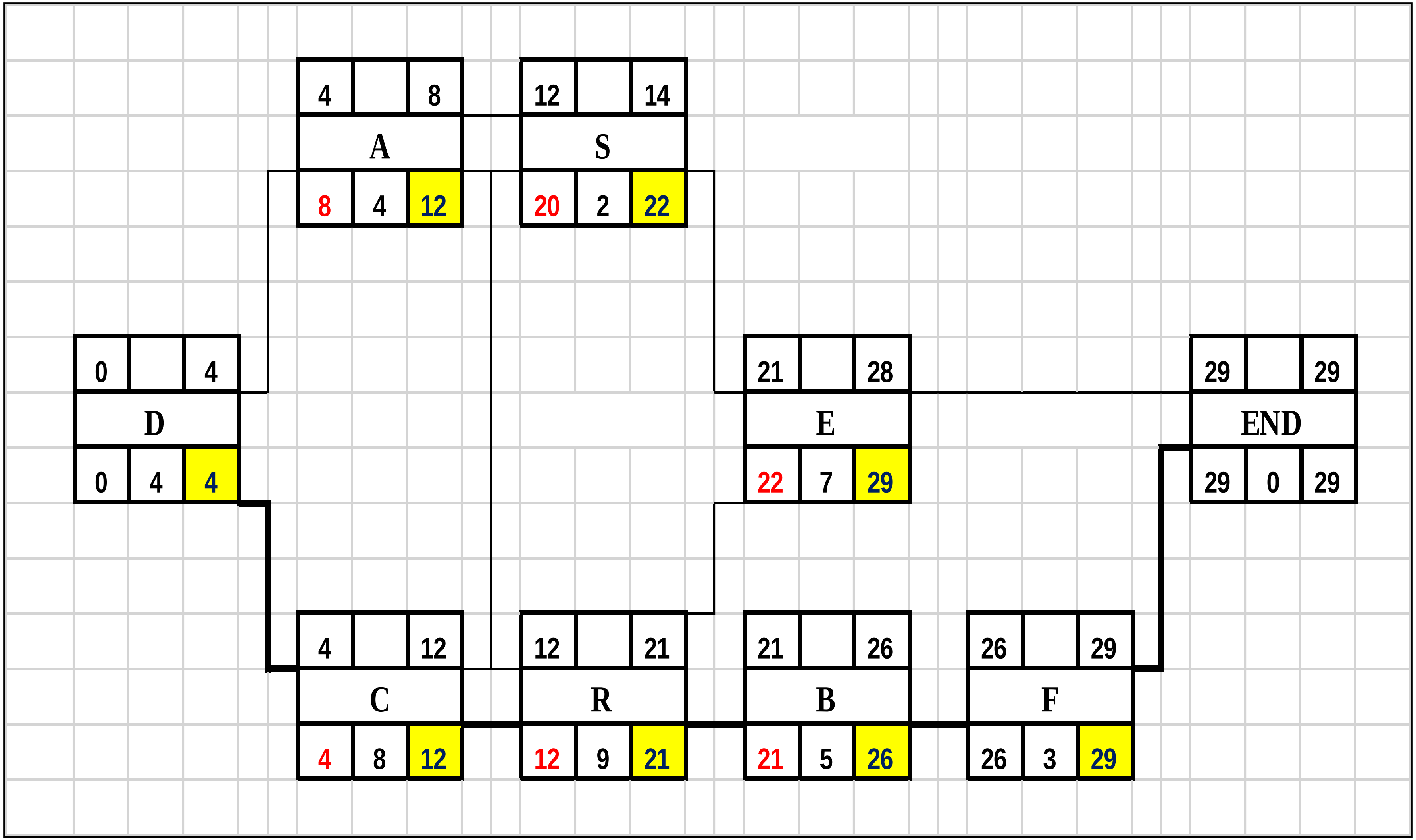
- In case of burst activities LF value is equal to the earliest (or **smallest**) of the LS times of the activities following.

Example

Calculate the late activity times (LS and LF).



■ Example


4
8
A
8
4
12
12
14
S
20
2
22
0
4
D
0
4
4
21
28
E
22
7
29
29
29
END
29
0
29
4
12
C
4
8
12
12
21
R
12
9
21
21
26
B
21
5
26
26
29
F
26
3
29

FLOAT Time

- Float or leeway is a measure of the time available for a given activity above and beyond its estimated duration.
- Two classifications of which are in general usage: **total float and free float**.

■ TOTAL FLOAT

- The total float of an activity is obtained by subtracting its ES time from its LS time. Subtracting the EF from the LF gives the same result.

Total float (TF) = LS - ES = LF - EF

- An activity with **zero** total float has **no spare time** and is, therefore, one of the operations that controls project completion time.
- Activities with zero total float are called "**critical activities**".

CRITICAL PATH

- Critical activity is quickly identified as one whose two start times at the left of the activity box are equal. Also equal are the two finish times at the right of the activity box.

- The critical activities must form a continuous path from project beginning to project end, this chain of critical activities is called the "critical path".

- The critical path is normally indicated on the diagram in some distinctive way such as with colors, heavy lines, or double lines.

- The critical path is the longest path in the network.

- Any delay in the finish date of a critical activity, for whatever reason, automatically prolongs project completion by the same amount.

FREE FLOAT

- The free float of an activity is the amount of time by which the completion of that activity can be deferred without delaying the early start of the following activities.

- The free float of an activity is found by subtracting its earliest finish time from the earliest start time of the activities directly following.

- **FF** = The smallest of the ES value of those activities immediately following - EF of the activity.

  = the smallest of the earliest start time of the successor activities minus the earliest finish time of the activity in question.

$$\mathbf{FF_i = Min.\ (ES_j) - EF_i}$$

CALENDAR-DATE SCHEDULE

- Activity times (ES, EF, LS, LF) obtained from previous calculations are expressed in terms of expired **working days**.

- For purposes of project directing, monitoring and control, it is necessary to convert these times to **calendar dates** on which each activity is expected to start and finish.

- This is done with the aid of a **calendar** on which the working days are numbered consecutively, starting with number 1 on the anticipated start date and skipping weekends and holidays.

Precedence Diagramming

- An important extension to the original activity-on-node concept appeared around 1964.
- The sole relationship used in PERT/CPM network is finish to start type of dependency, with $Fs_{ij} = 0$.
- Precedence diagramming includes precedence relationships among the activities. In Addition, one may specify a “lag time” associated with any of the precedence relationships, which can be used to account for overlapping times among activities.
- The computation of activity times (published in 1973) is more complex than AON.

Lag / Lead Times

- In many cases, there is a delay between the completion of one activity and the start of another following or there is a need to show that one activity will **overlap** another in some fashion.

- A successor "**lags**" a predecessor, but a predecessor "**leads**" a successor.

- Lag time can be designated on a dependency line with a positive, negative, or zero value.

- **Limitations and Disadvantages of Lag**:
  - Lag would complicate the scheduling process.
  - Lags are not extensively used except where the time effects are substantial for special project type.

Precedence Diagramming Relationships

1. **Start-to-Start (SS_{ij})**
   SS_{ij} is equal to the minimum number of time units that must be complete on the preceding activity (i) prior to the start of the successor (j). "Lag" is always applied to SS relation.

2. **Finish-to-Finish (FF_{ij})**
   FF_{ij} is equal to the minimum number of time units that must remain to be completed on the successor (j) after the completion of the predecessor (i).

3. **Finish-to-Start (FS_{ij})**
   FS_{ij} is equal to the minimum number of time units that must transpire from the completion of the predecessor (i) prior to the start of the successor (j).

Precedence Diagramming Relationships

4. Start-to-Finish (SF_{ij})

SF_{ij} is equal to the minimum number of time units that must transpire from the start of the predecessor (i) to the completion of the successor (j).

5. Start-to-Start and Finish-to-Finish (ZZ_{ij}):

ZZ_{ij} is a combination of two constraints, i.e., a start-to-start and finish-to-finish relationship. It is written with the SS_{ij} time units first, followed by the FF_{ij} time units.

■ Precedence Diagram Computation

Forward Pass Computations

$$[1]\ ES_j = \text{Max. all}_{\ i} \left\{ \begin{array}{l} \text{Initial Time} \\ EF_i + FS_{ij} \\ ES_i + SS_{ij} \\ EF_i + FF_{ij} - D_j \\ ES_i + SF_{ij} - D_j \end{array} \right\}$$

$$[2]\ EF_j = ES_j + D_j$$

Start-to-Start (SS_{ij})
Start-to-Finish (SF_{ij})
Finish-to-Start (FSij)
Finish-to-Finish (FF_{ij})
Start-to-Start and Finish-to-Finish (ZZ_{ij})

Precedence Diagram Computation

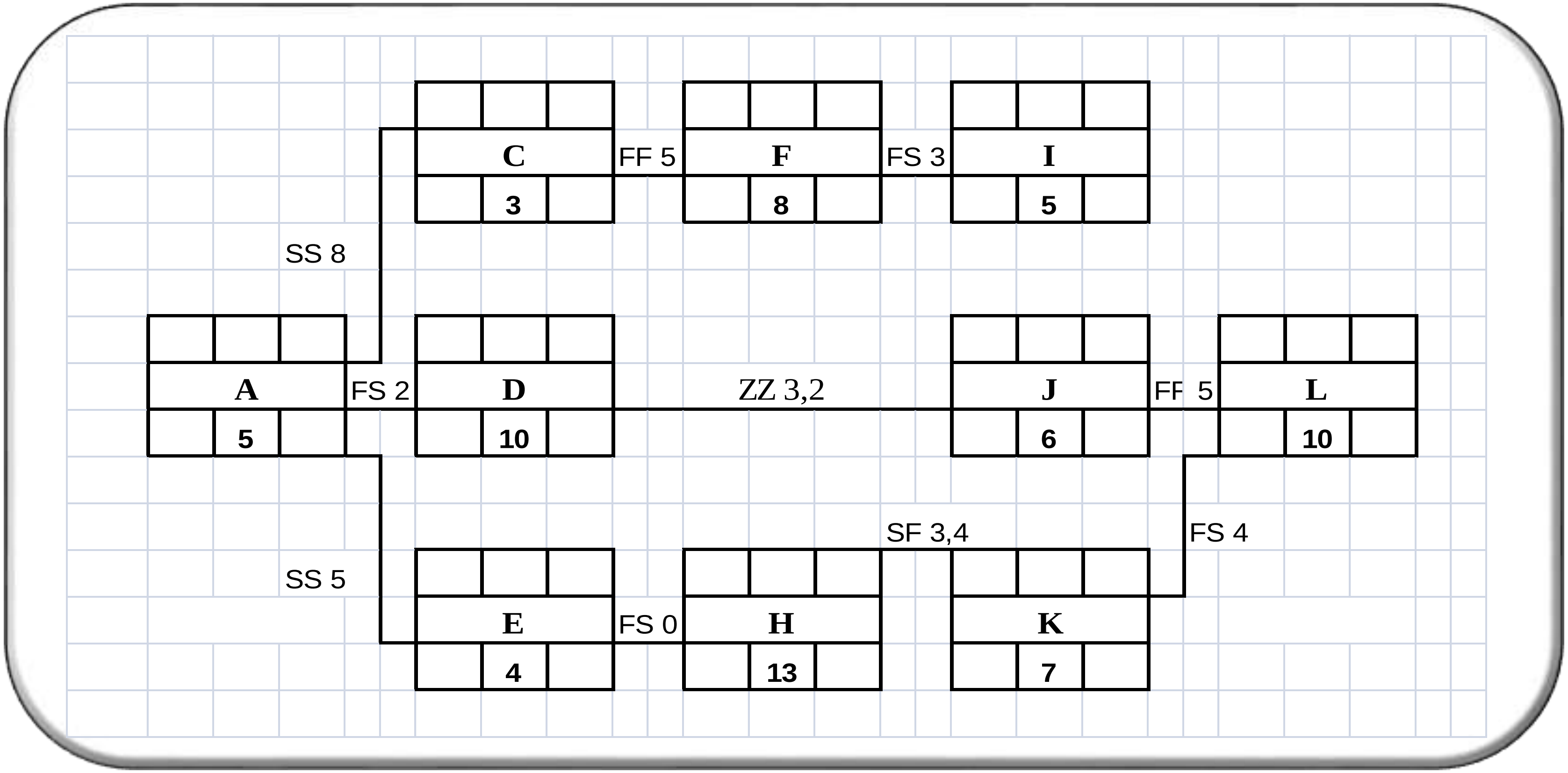
Backward Pass Computations

$$[3]\ LF_i = \text{Min. all}_{\ j} \left\{ \begin{array}{l} \text{Terminal Time} \\ LS_j - FS_{ij} \\ LF_j - FF_{ij} \\ LS_j - SS_{ij} + D_i \\ LF_j - SF_{ij} + D_i \end{array} \right\}$$

$$[4]\ LS_i = LF_i - D_i$$

Example

Given the precedence network for a small engineering project with activity durations in working days, it is required to compute the activity times (ES, EF, LS, and LF) and total floats (TF) and then indicate the critical activities.



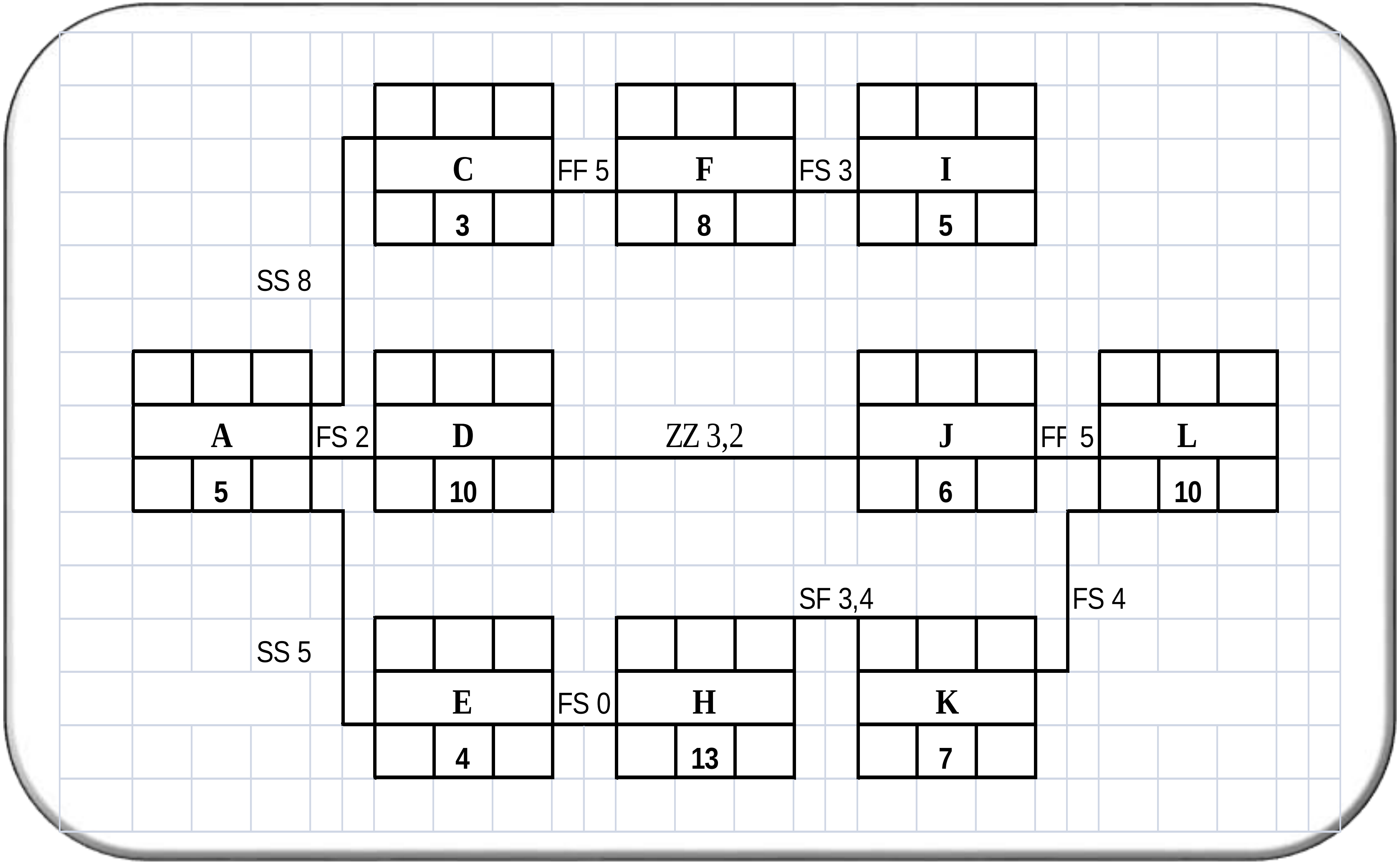
Example

Start-to-Start (SS_{ij})

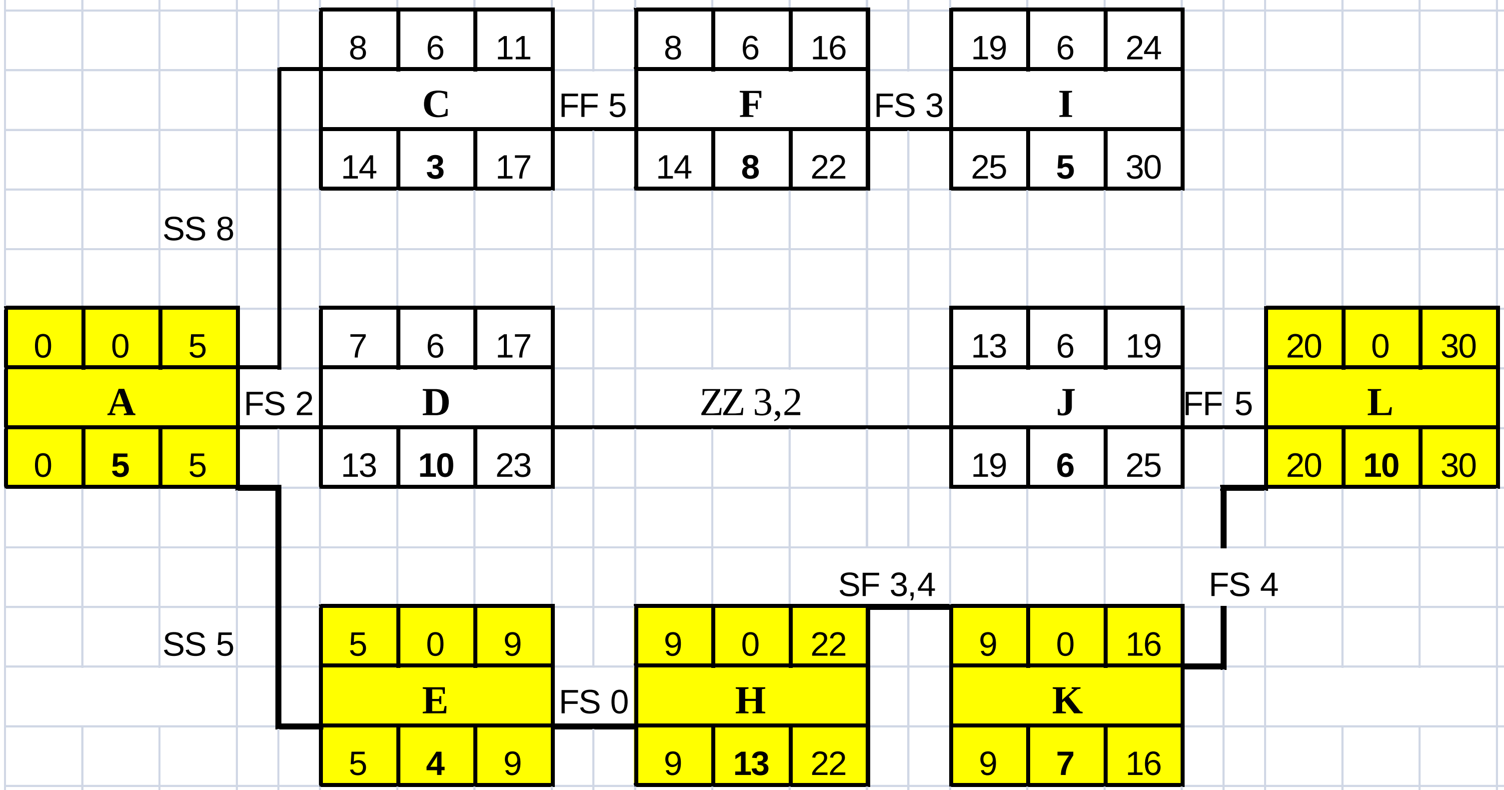
Finish-to-Start (FSij)

Start-to-Finish (SF_{ij})

Finish-to-Finish (FF_{ij})



Example


8
6
11
C
14
3
17
FF 5
8
6
16
F
14
8
22
FS 3
19
6
24
I
25
5
30
SS 8
0
0
5
A
0
5
5
FS 2
7
6
17
D
13
10
23
ZZ 3,2
13
6
19
J
19
6
25
FF 5
20
0
30
L
20
10
30
SF 3,4
FS 4
SS 5
5
0
9
E
5
4
9
FS 0
9
0
22
H
9
13
22
9
0
16
K
9
7
16

HAMMOCK ACTIVITY

- An activity that extends from one activity to another, but which has no estimated duration of its own.
- It is time-consuming and requires resources, but its duration is controlled, not by its own nature, but by the two activities between which it spans.
- Its ES and LS times are determined by the activity where it begins and its EF and LF times are dictated by the activity at its conclusion.
- Examples: Dewatering, Haul road maintenance

■ MILESTONES

- Milestones are points in time that have been identified as being important intermediate **reference points** during the accomplishment of the work.
- Milestone events can include dates imposed by the customer for the finishing of certain tasks as well as target dates set by the project manager for the completion of certain segments of the work.
- Distinctive geometric figure is preferred to represent a milestone (circles, ovals, or other shapes) can be used.
- Any information pertaining to a milestone and considered to be useful may be entered.