

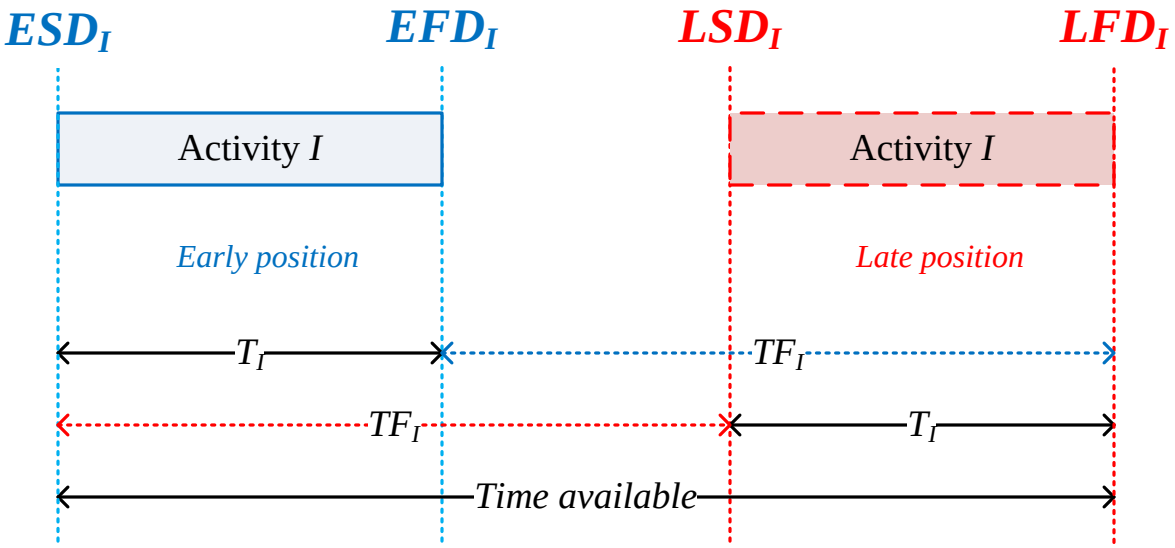


Figure 7.1 The four activity times — precedence diagram notation

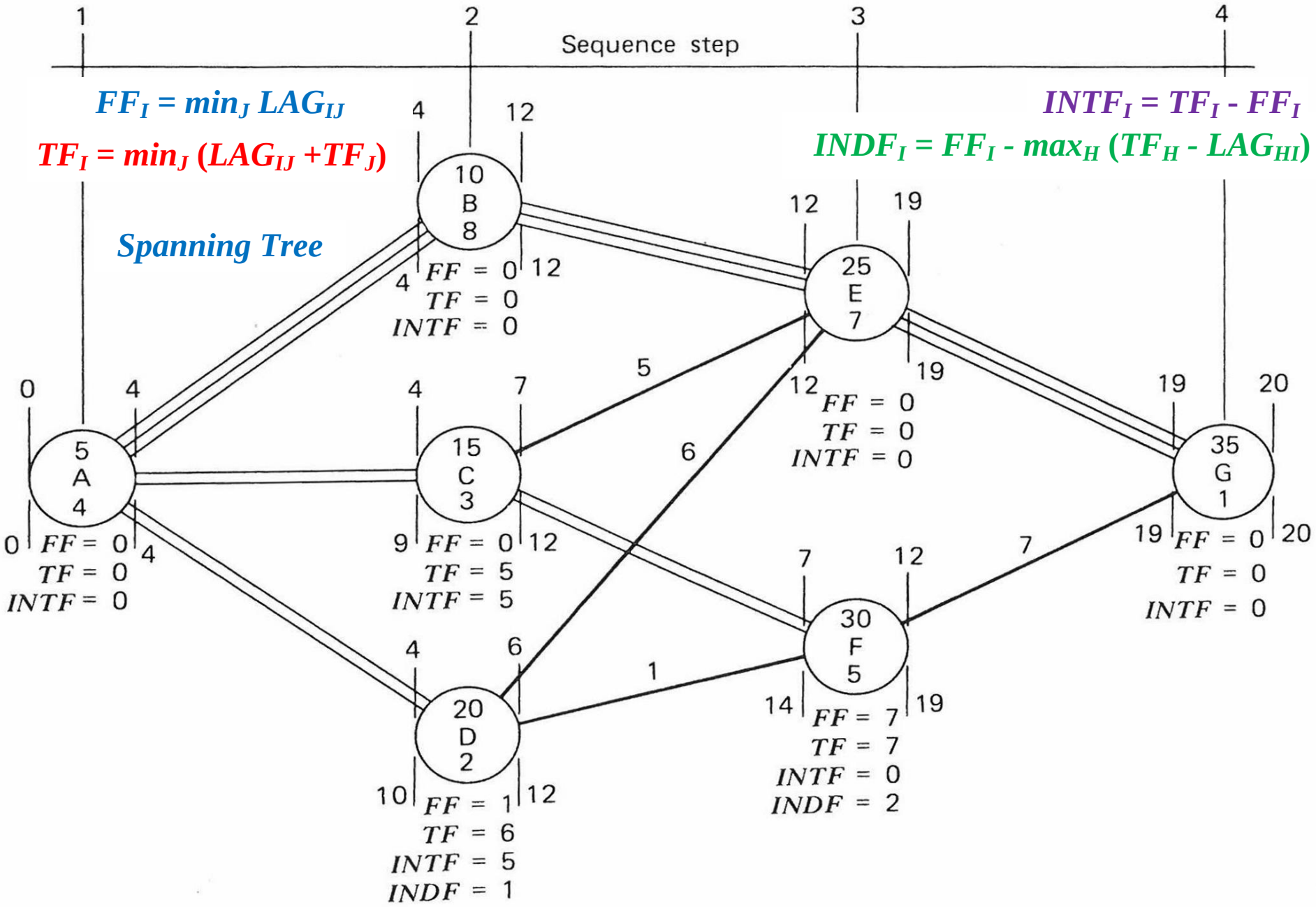
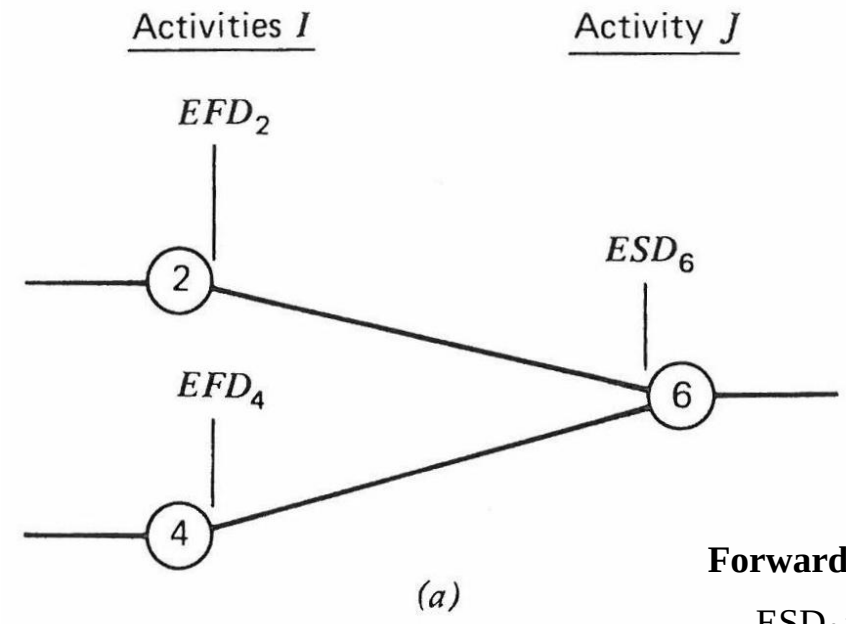


Figure 7.11 Late date determination—sample precedence network.



(a)

Forward Pass Algorithm:

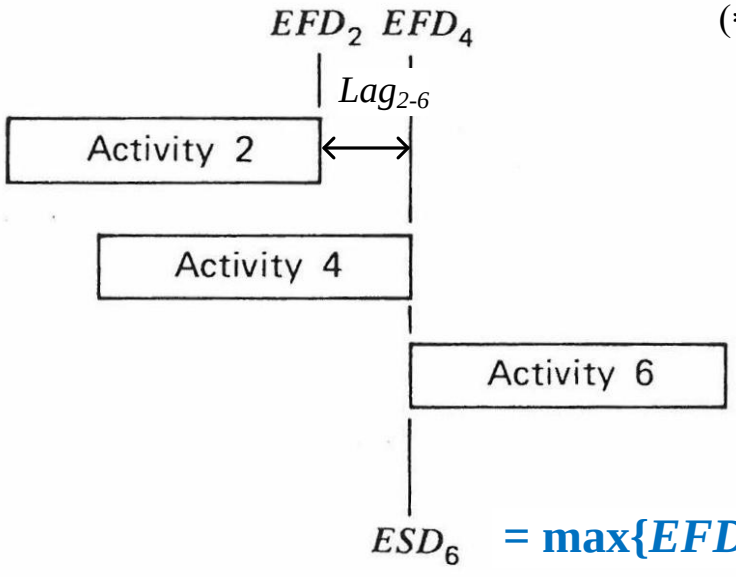
$ESD_0 = 0$ (assigned)

$EFD_0 = ESD_0 + T_0$

$ESD_J = \max EFD_I$ (*)

$EFD_J = ESD_J + T_J$

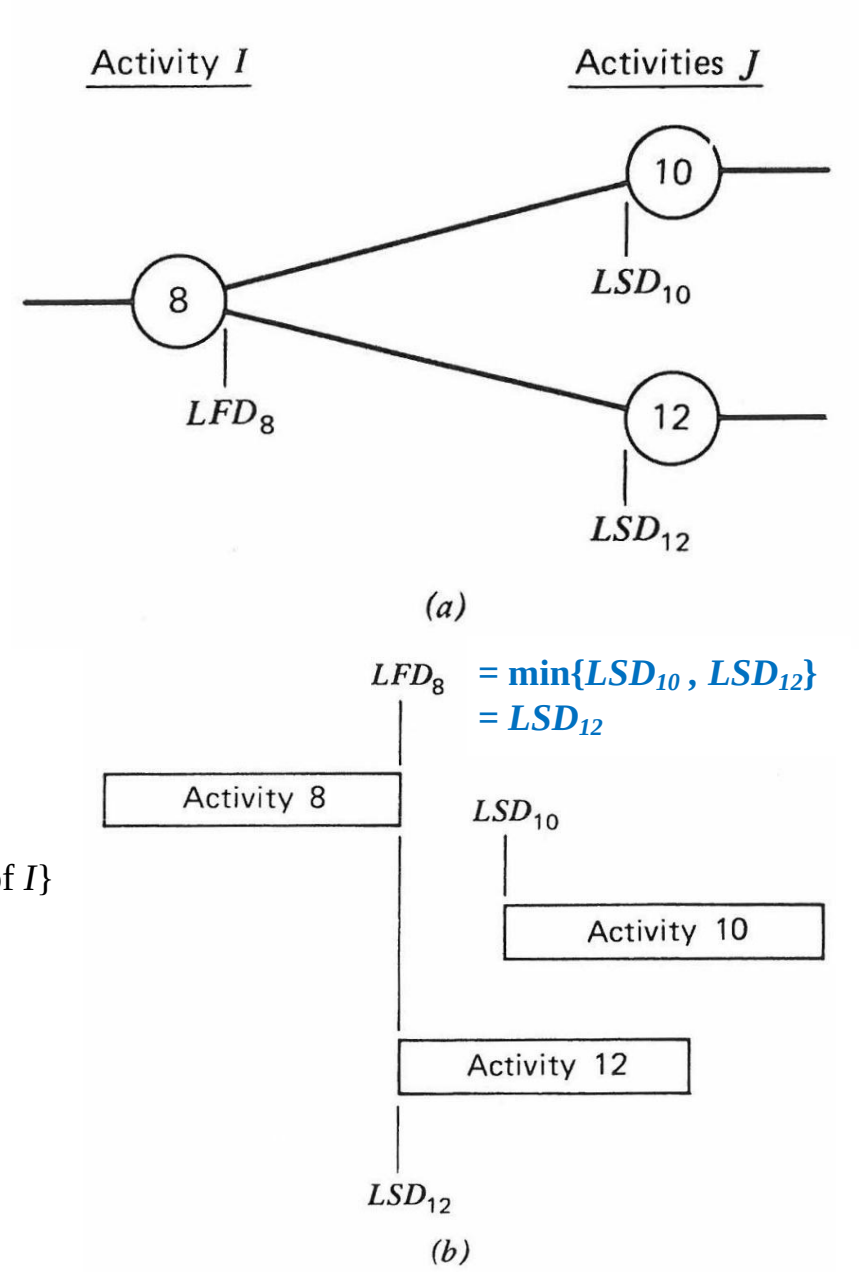
(*) $I \in$ {direct predecessors of J}



$= \max\{EFD_2 , EFD_4\} = EFD_4$

(b)

Figure 7.2 Early finish — early start relationships



Backward Pass Algorithm:

$LFD_f = EFD_f$ (assigned)

$LSD_f = LFD_f - T_f$

LFD_I = min LSD_J (*)

$LSD_I = LFD_I - T_I$

(*) $J \in$ {direct successors of I}

Figure 7.3 Late start — late finish relationships

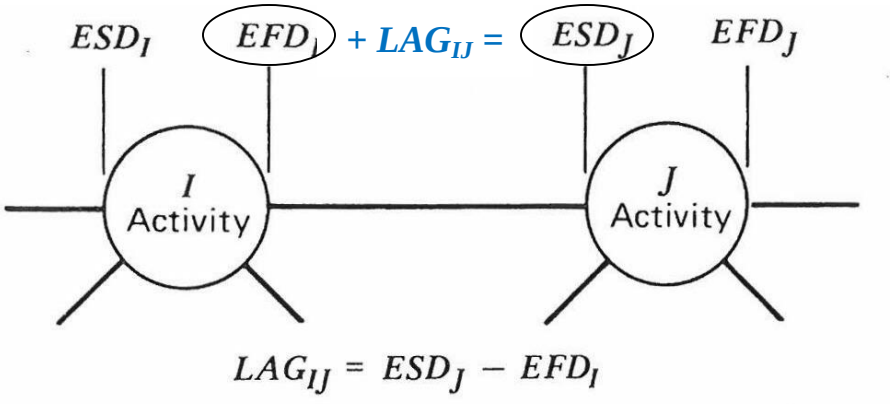


Figure 7.4 Link lag.

Activity-Oriented Approach

Forward Pass Algorithm:

1. Assign ESD_S (where S is the *start* activity; typically, $ESD_S = 0$)
2. Set $EFD_S = ESD_S + T_S$
3. Find an activity J for which the EFD_I for all immediate predecessors I are known.
4. Set $ESD_J = \max EFD_I, \ \forall\ I \in$ {direct predecessors of J}
5. Set $EFD_J = ESD_J + T_J$
6. If the ESD and EFD for all activities have been computed, then STOP, else GOTO 3.

Minimum Project Duration = EFD_F (where F is the *finish* activity)

Backward Pass Algorithm:

1. Assign LFD_F (typically, $LFD_F = EFD_F$)
2. Set $LSD_F = LFD_F - T_F$
3. Find an activity I for which the LSD_J for all immediate successors J are known.
4. Set $LFD_I = \min LSD_J, \ \forall\ J \in$ {direct successors of I}
5. Set $LSD_I = LFD_I - T_I$
6. If the LSD and LFD for all activities have been computed, then STOP, else GOTO 3.

Floats:

Total Float: $TF_I = LFD_I - EFD_I = LSD_I - ESD_I$

If $TF_I = 0$, then activity I is on the Critical Path.

Free Float: $FF_I = \min ESD_J - EFD_I, \ \forall\ J \in$ {direct successors of I}

$FF_I \le TF_I$

Interfering Float: $INTF_I = TF_I - FF_I = LFD_I - \min ESD_J, \ \forall\ J \in$ {direct successors of I}

$INTF_I \le TF_I$

Independent Float: $INDF_I = \min ESD_J - \max LFD_H - T_I, \quad \forall\ J \in$ {direct successors of I}

$\forall\ H \in$ {direct predecessors of I}

$INDF_I \le FF_I \le TF_I$. If $INDF_I < 0$, then set $INDF_I = 0$.

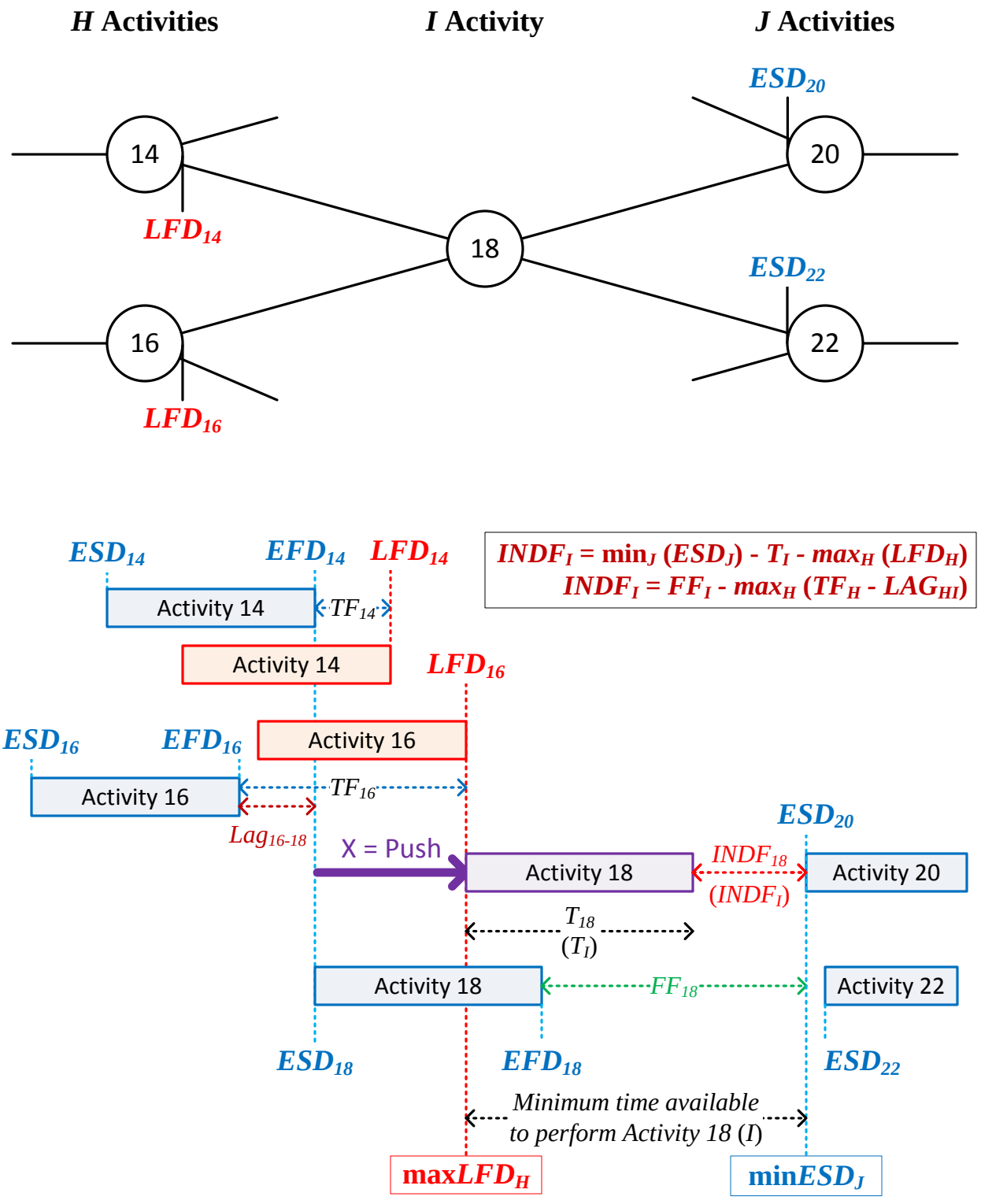


Figure 7.6 Independent float relationships

Link-Lag-Oriented Approach

Forward Pass Algorithm:

1. Assign ESD_S (where S is the *start* activity; typically, $ESD_S = 0$)
2. Set $EFD_S = ESD_S + T_S$
3. Find an activity J for which the EFD_I for all immediate predecessors I are known.
4. Set $ESD_J = \max EFD_I, \ \forall\ I \in$ {direct predecessors of J}
5. Set $EFD_J = ESD_J + T_J$
6. If the ESD and EFD for all activities have been computed, then STOP, else GOTO 3.

Minimum Project Duration = EFD_F (where F is the *finish* activity)

Compute Link Lags:

Set $LAG_{IJ} = ESD_J - EFD_I, \ \forall\ I, \forall\ J \in$ {direct successors of I}

Floats:

Free Float: $FF_I = \min ESD_J - EFD_I, \ \forall\ J \in$ {direct successors of I}

$= \min LAG_{IJ}, \ \forall\ J \in$ {direct successors of I}

$FF_I \le TF_I$

Total Float: $TF_I = LFD_I - EFD_I = \min LSD_J - EFD_I = \min \{LSD_J - EFD_I\}$

$= \min \{ESD_J + TF_J - EFD_I\} = \min \{LAG_{IJ} + TF_J\}, \ \forall\ J$

If $TF_I = 0$, then activity I is on the Critical Path.

Interfering Float: $INTF_I = TF_I - FF_I = LFD_I - \min ESD_J, \ \forall\ J \in$ {direct successors of I}

$INTF_I \le TF_I$

Independent Float: $INDF_I = \min ESD_J - \max LFD_H - T_I, \qquad \forall\ J, \ \forall\ H$

$= \min ESD_J - \max\{EFD_H + TF_H - T_I\}$

$= \min ESD_J + EFD_I - EFD_I - \max\{EFD_H + TF_H\} + ESD_I - ESD_I - T_I$

$= \min\{ESD_J - EFD_I\} - \max\{TF_H - \{ESD_I - EFD_H\}\} + \{EFD_I - ESD_I\} - T_I$

$= \min\{LAG_{IJ}\} - \max\{TF_H - LAG_{HI}\}$

$= FF_I - \max\{TF_H - LAG_{HI}\}, \ \forall\ H \in$ {direct predecessors of I}

$INDF_I \le FF_I \le TF_I$. If $INDF_I < 0$, then set $INDF_I = 0$.

$FF_I = \min_J LAG_{IJ}$ $\quad$ $INTF_I = TF_I - FF_I$

$TF_I = \min_J (LAG_{IJ} + TF_J)$ $\quad$ $INDF_I = FF_I - \max_H (TF_H - LAG_{HI})$

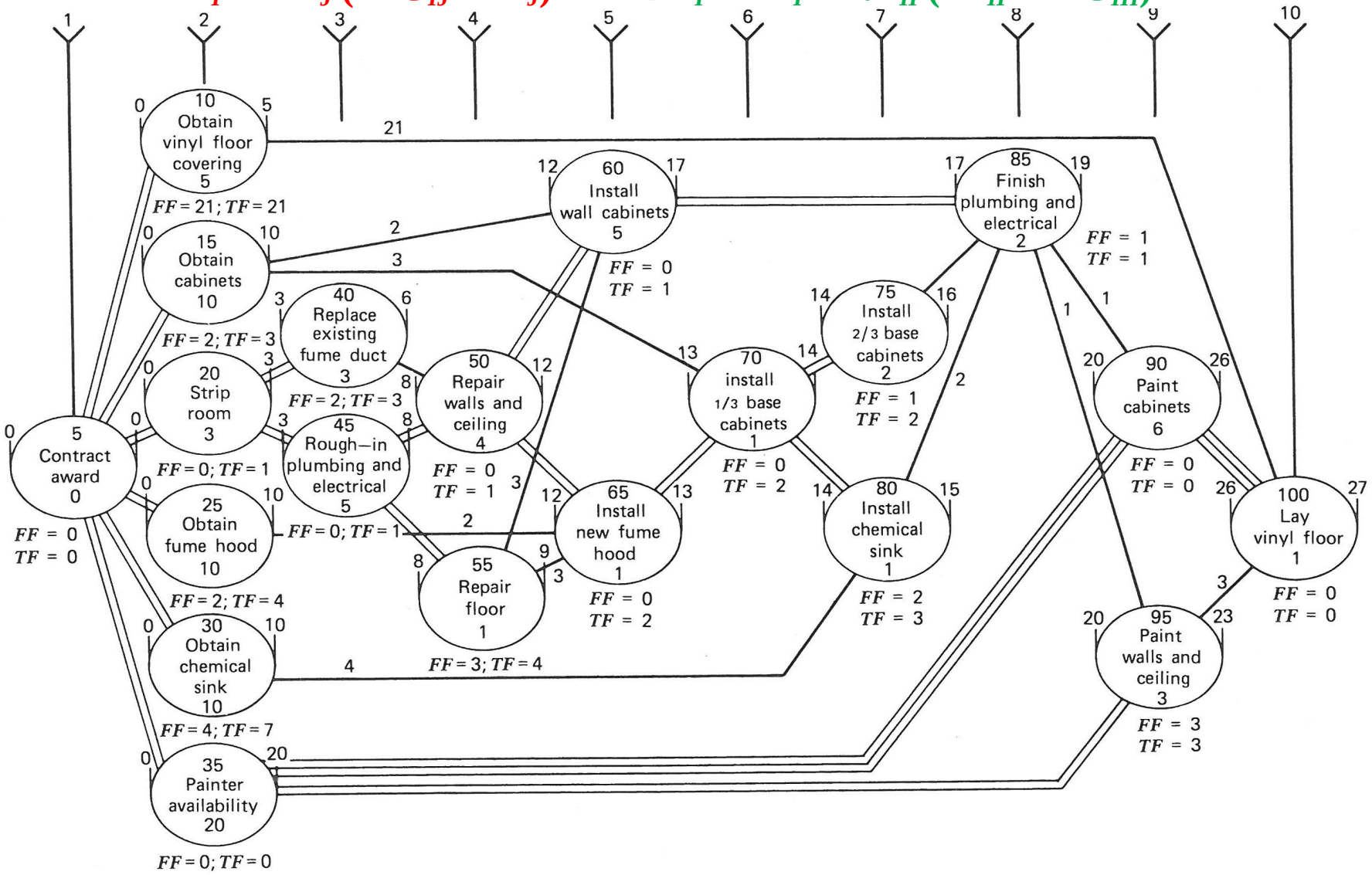


Figure 7.12 Precedence diagram—remodeling chemical laboratory.

TABLE 7.1

Activity List—Sample Precedence Network

No.	Activity	Depends upon
5	A	—
10	B	5
15	C	5
20	D	5
25	E	10, 15, 20
30	F	15, 20
35	G	25, 30

$$FF_I = \min_J LAG_{IJ} \qquad INTF_I = TF_I - FF_I$$

$$TF_I = \min_J (LAG_{IJ} + TF_J) \qquad INDF_I = FF_I - \max_H (TF_H - LAG_{HI})$$

TABLE 7.2

Computation Input—Sample Precedence Network

Activity	*T*	Act (*J*)	Pre (*I*)	*ESD*	*EFD*	*LAG*	*FF*	*TF*	*INTF*	*INDF*	*LSD*	*LFD*
A	4	5										
B	8	10	5									
C	3	15	5									
D	2	20	5									
E	7	25	10 15 20									
F	5	30	15 20									
G	1	35	25 30									

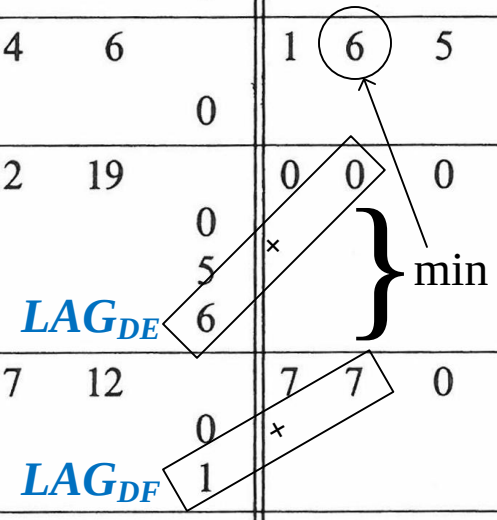
$FF_I = min_J\ LAG_{IJ}$ $INTF_I = TF_I - FF_I$

$TF_I = min_J\ (LAG_{IJ} + TF_J)$ $INDF_I = FF_I - max_H\ (TF_H - LAG_{HI})$

TABLE 7.5

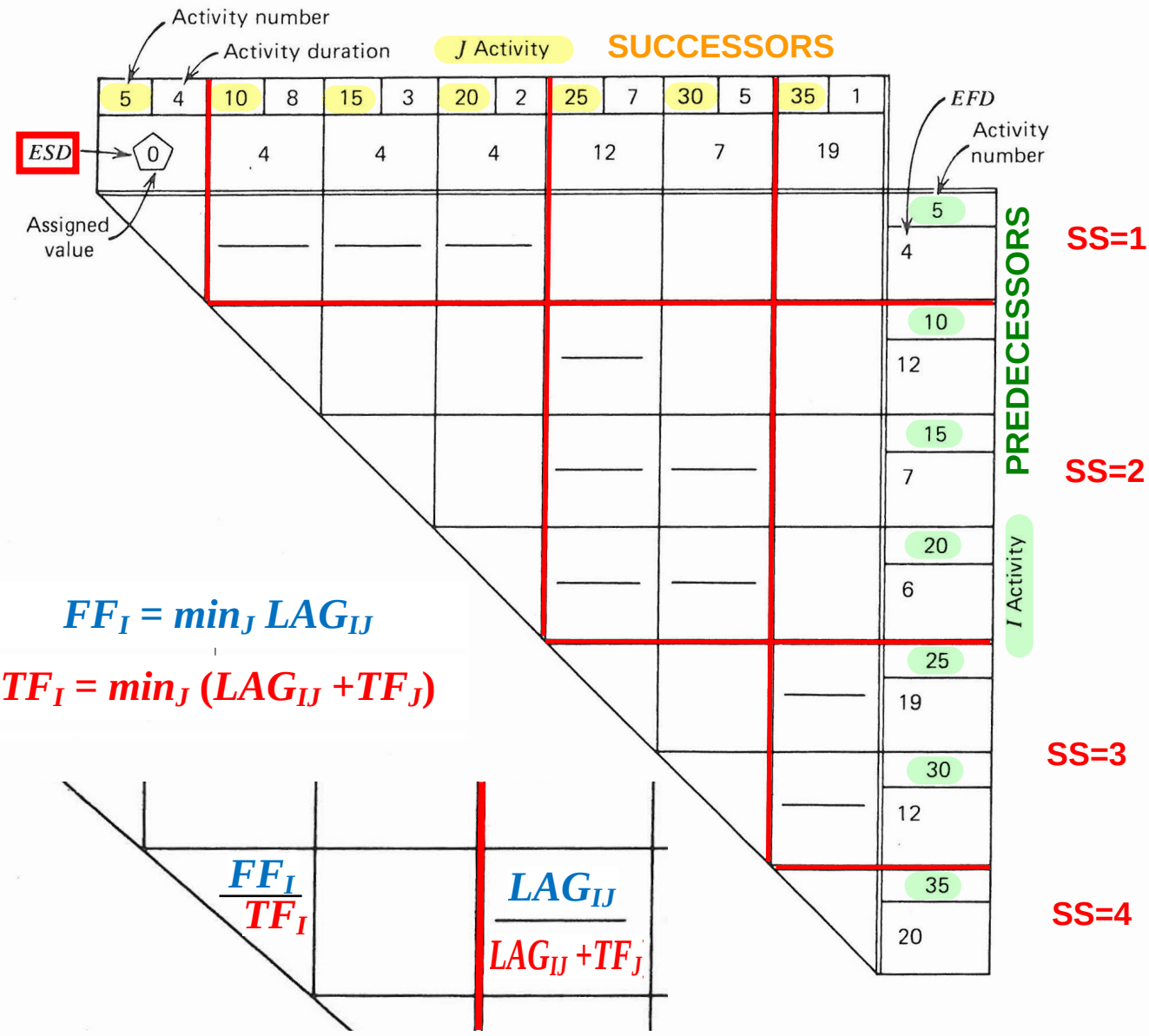
Late Date Determination—Sample Precedence Network

	Activity	T	Act (J)	Pre (I)	ESD	EFD	LAG	FF	TF	INTF	INDF	LSD	LFD	
*	A	4	5		0	4		0	0	0	—	0	4	*
*	B	8	10		4	12		0	0	0	—	4	12	*
				5			0							
	C	3	15		4	7		0	5	5	—	9	12	
				5			0							
	D	2	20		4	6		1	6	5	1	10	12	
				5			0							
*	E	7	25		12	19		0	0	0	—	12	19	*
				10			0							
				15			5							
				20			6							
	F	5	30		7	12		7	7	0	2	14	19	
				15			0							
				20			1							
*	G	1	35		19	20		0	0	0	—	19	20	*
				25			0							
				30			7							



SS=1 SS=2 SS=3 SS=4

Computations On A Matrix



$$FF_I = \min_J LAG_{IJ}$$

$$TF_I = \min_J (LAG_{IJ} + TF_J)$$

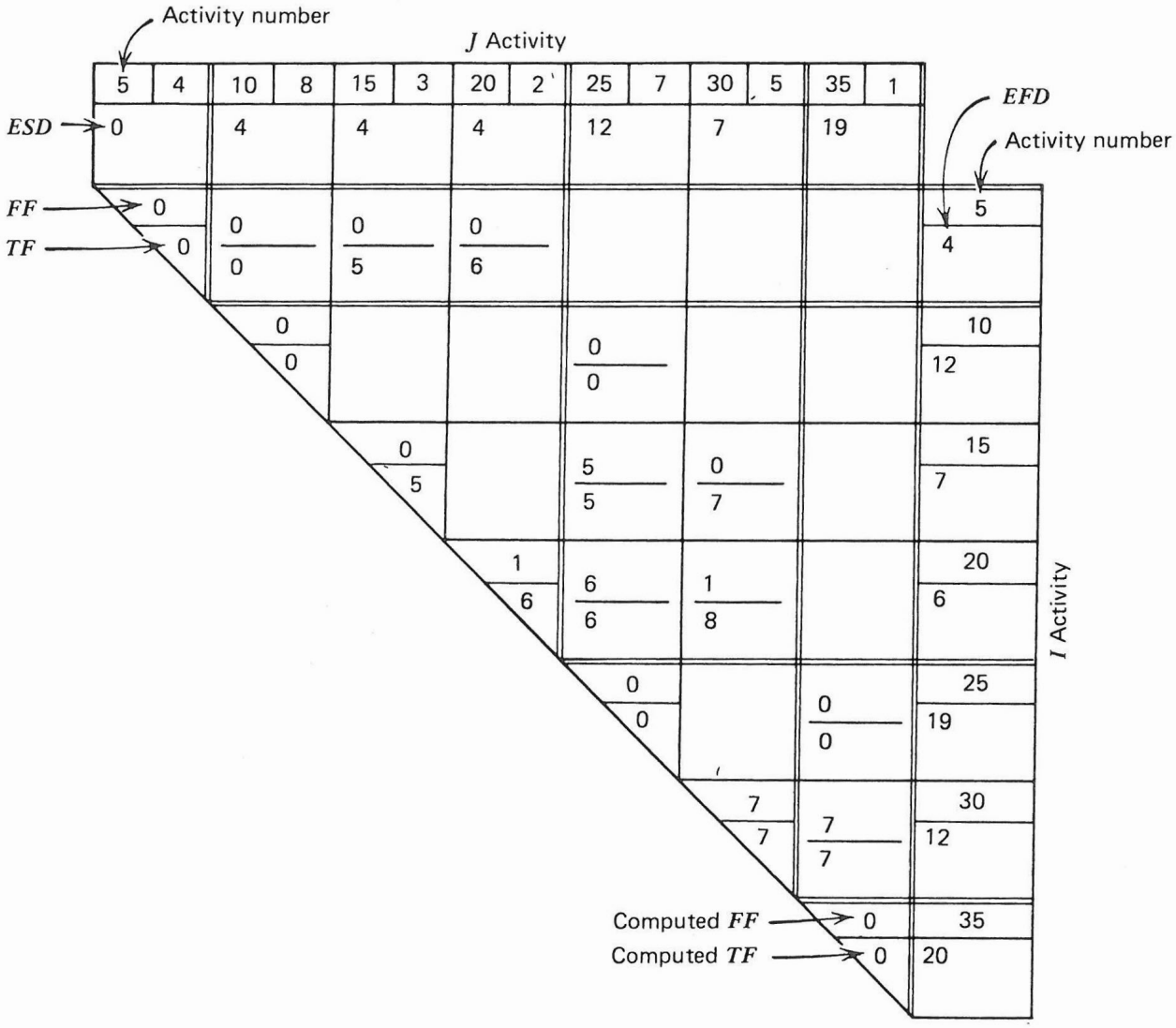


Figure 7.16 Total float computation—sample precedence network.

CHAPTER 7: Scheduling Precedence Networks

Activity-on-Node Network Facts

- The activities in an activity-on-node network can be organized and drawn in a rectangular visual grid, where the vertical columns are called "sequence steps", so that all precedence links have an unambiguous left-to-right direction and thus do not require arrowheads.
- The *sequence step* for an activity is the *leftmost visual column* in which the activity can be drawn and still maintain the left-to-right direction of all precedence links.
- The *ESD* of an activity is the max *EFD* of its direct predecessors.
- The *LFD* of an activity is the min *LSD* of its direct successors.
- The *ESD* of an activity is the duration of its *longest* path of *predecessors* (which includes the single *starting* activity).
- The *LFD* of an activity is the duration of the project minus the duration of its longest path of *successors* (which includes the single *finishing* activity).
- The *critical path* is the *longest* path (sequence of activities) that starts with the single *starting* activity and ends with the single *finishing* activity.
- A network may have multiple critical paths, all with the same duration. Critical activities may belong to multiple critical paths.
- *TF* is by how much an activity can be delayed without delaying the project.
- An activity with *TF*=0 is a *critical* activity. Delaying a critical activity delays the project.
- The *critical path* is a sequence of *critical activities*, all with *TF*=0.
- For a critical activity, all floats are zero, not just the *TF*.
- *TF* consists of two parts; the beginning part is *FF* and the rest is *INTF*.
- *TF* = *FF* + *INTF*. As a result, *FF*<=*TF* and *INTF*<=*TF*
- *FF* is by how much an activity can be delayed and not delay the *ESD* of *any* direct successor.
- *INTF* is that part of *TF* which ***does*** delay the *ESD* of a direct successor.
- For an activity to have *FF* it must have *TF*. If *TF*=0 then *FF*=0.
- *TF* and *FF* computations assume that an activity can start at its *ESD*.

- If an activity is succeeded *directly* by a critical activity, then *TF*=*FF* and *INTF*=0 (because any TF>FF would delay the critical direct successor and thus would also delay the project).
- Activities on the same path *share* their *TF*. If one is delayed by *TF*, then the TF of succeeding activities on the same path *may* be reduced.
- If an activity is delayed by *TF* then the *FF* of any successor *may* be reduced. (By definition of *INDF*, *FF* is reduced at most to *INDF*.)
- At least one of the *direct predecessors* of an activity must have *FF*=0.
- The direct predecessor with *FF*=0 is the one that has max*EFD* and is the one that determines the *ESD* of the activity.
- If an activity is the only direct predecessor of another then its *FF*=0.
- If an activity has two direct predecessors, then it is *possible* (but not necessary) for one of these predecessors to have *FF*>0.
- For an activity to have *FF*>0 it is necessary (but not sufficient) that it is not the *only* direct predecessor of any activity. (This way some other predecessor may be the one that must have *FF*=0.)
- *FF* is the only float that can be calculated using only ***early*** start and ***early*** finish times (thus, no backward pass is needed).
- *INDF* is by how much an activity can be delayed and not delay the *ESD* of ***any*** direct successor when its start has been delayed because ***all*** its direct predecessors used their *TF* and finished at their *LFD*.
- *INDF* assumes a worst-case scenario: All direct predecessors finish late (at LFD) and all direct successors must start early (at *ESD*).
- *INDF* is the *FF* that remains when the start of an activity is delayed because all direct predecessors are delayed by their *TF* and finish at their *LFD*.
- *INDF* is the latter (ending) part of *FF*: As a result, *INDF* <= *FF*.
- For an activity to have *INDF* it must have *FF*. If *FF*=0 then *INDF*=0.
- An *activity chain* is a sequence of activities that follow each other such that only the *first* activity in the chain has more than one direct predecessor and only the *last* activity in the chain has more than one direct successor. All other intermediate activities in the chain have exactly one direct predecessor and one direct successor.
- An activity chain is defined by its *first* activity and its *last* activity.

- All activities on an activity chain can be substituted by one "*virtual activity*" whose direct predecessors are those of the *first* activity, its direct successors are those of the *last* activity, and whose duration is the sum of the durations of all the activities on the activity chain.
- All activities on an activity chain have the same *TF* which equals the *TF* of the "virtual activity".
- Each activity on an activity chain (except the last) is the only predecessor of its direct successor. Hence, all activities on an activity chain have *FF*=0 except for the last activity which *may* have FF>0.
- The *Lag* of a precedence link is the time interval (time distance) from the *EFD* of the predecessor to the *ESD* of the successor.
- Link *Lags* depend only on early times (no backward pass is needed).
- There must be at least one link with *Lag=0* entering each activity.
- The link with *Lag*=0 comes from the predecessor with max*EFD* and indicates the predecessor that gave the activity its *ESD*.
- In a network, the set of all links with *Lag*=0 form a *tree*, called the *early spanning tree*, whose *root* is the single starting activity.
- The spanning tree is made up of *branches*, i.e., unbroken sequences of links (all with *Lag*=0) that begin at the tree's *root*, i.e., the *starting activity*, and visit each and every activity in the network (at least once).
- The *spanning tree* is like a *genealogy tree*. It shows the sequence of predecessors that gave each activity its *ESD*.
- The branch of the spanning tree (sequence of links with *Lag*=0) that enters any given activity defines the longest path to that activity.
- The longest path to an activity defines its *ESD.*
- There could be more than one longest path to an activity (all with the same duration).

7.10 Figure Ex. 7.1 represents a portion of a precedence diagram. From the data given compute the four float values for Activity E.

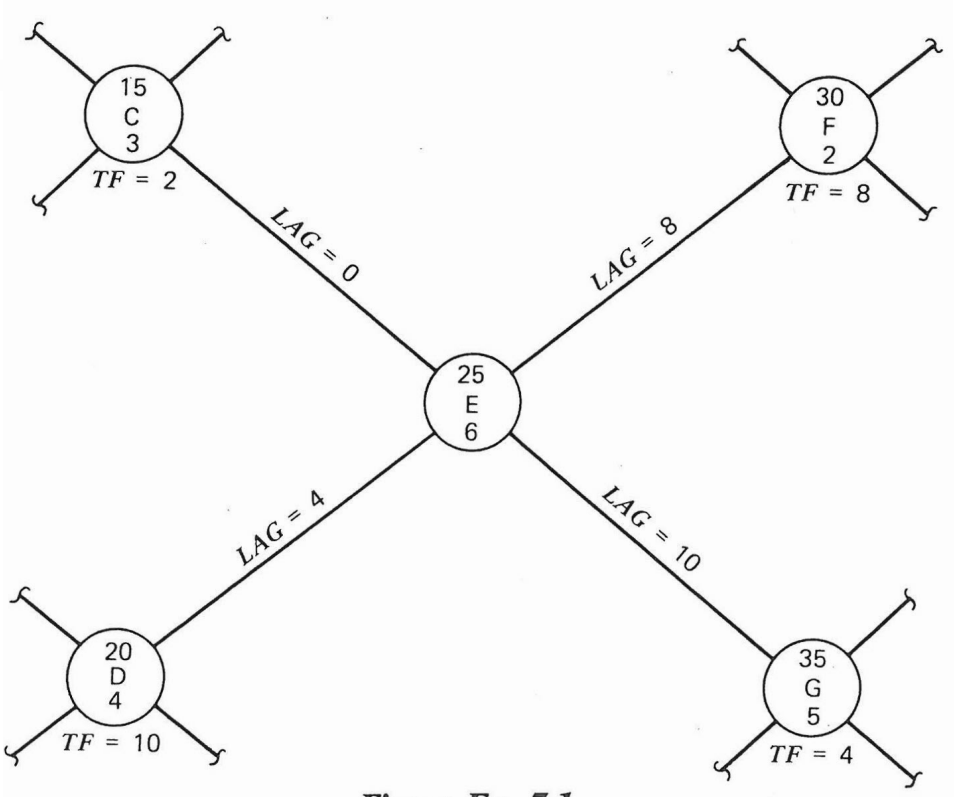


Figure Ex. 7.1

7.11 Figure Ex. 7.2 shows a precedence network of ten activities. Compute all the link lags and the total and free floats for the activities. Indicate the critical path.

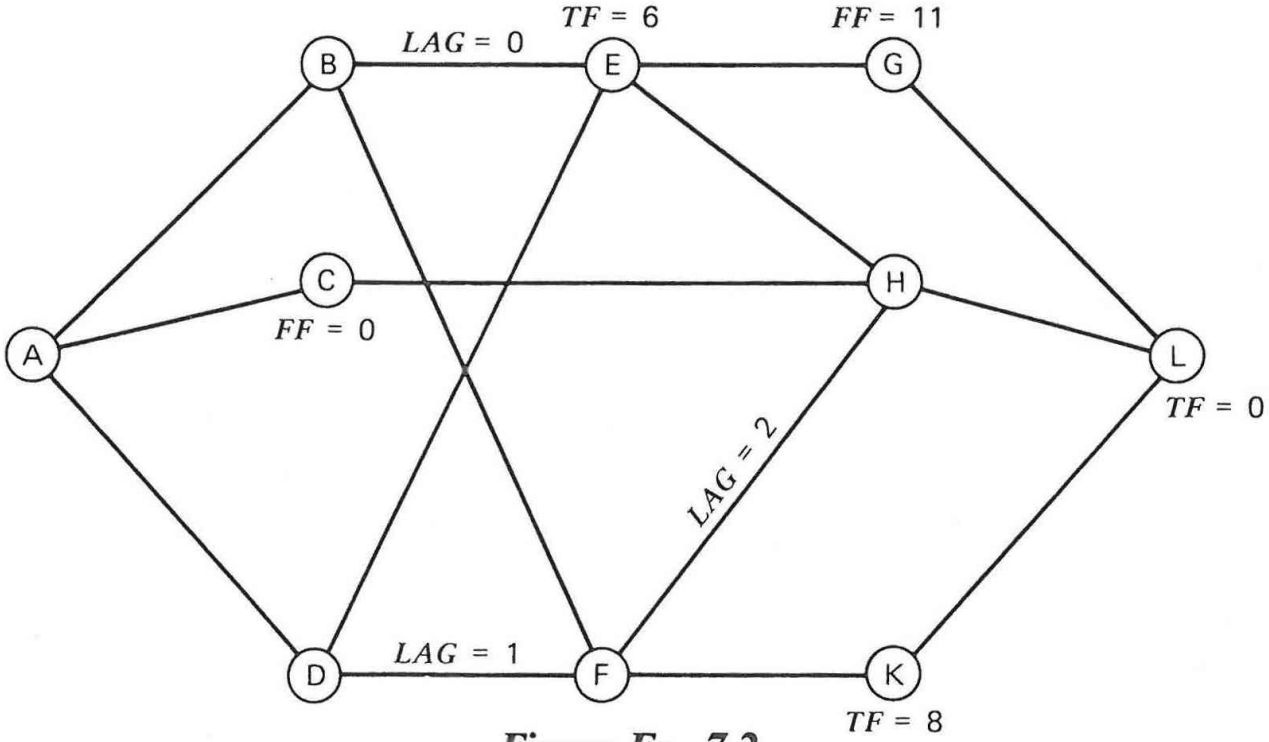


Figure Ex. 7.2