CPM: Critical Path Method

This Hypercard stack was prepared by:
Dennis L. Bricker,
Dept. of Industrial Engineering,
University of Iowa,
Iowa City, Iowa 52242
e-mail: dbricker@icaen.uiowa.edu

author

Example Project

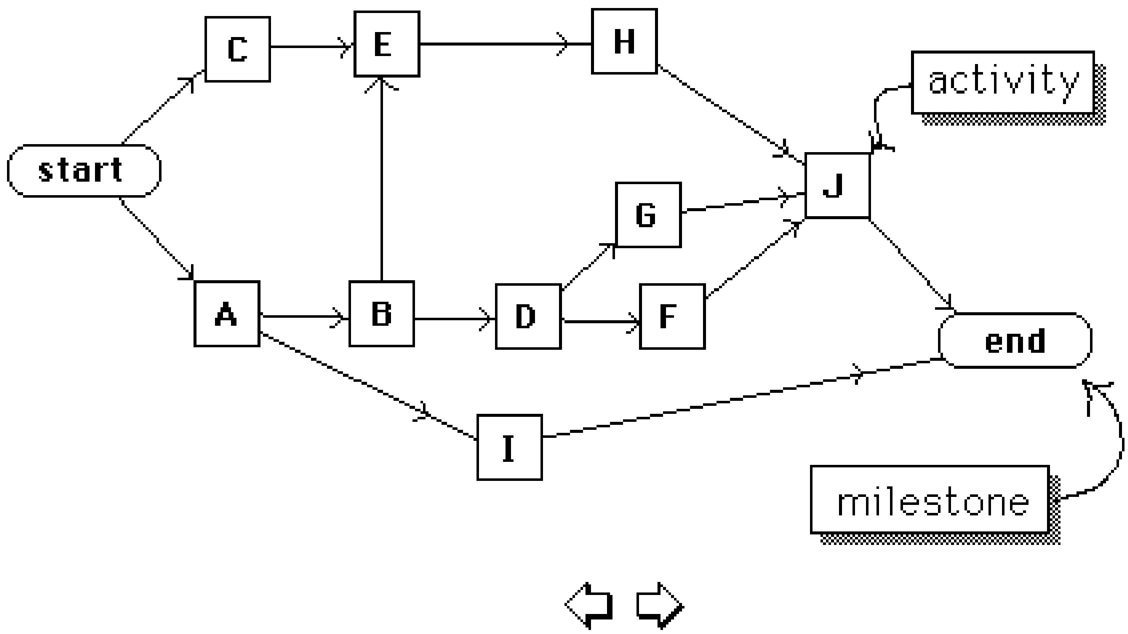
task	predecessor	duration
A	none	5
B	A	3
C	none	3
D	B	2
E	B,C	4
F	D	4
G	D	2
H	E	8
I	A	5
J	F,G,H	3

A project has two network representations:

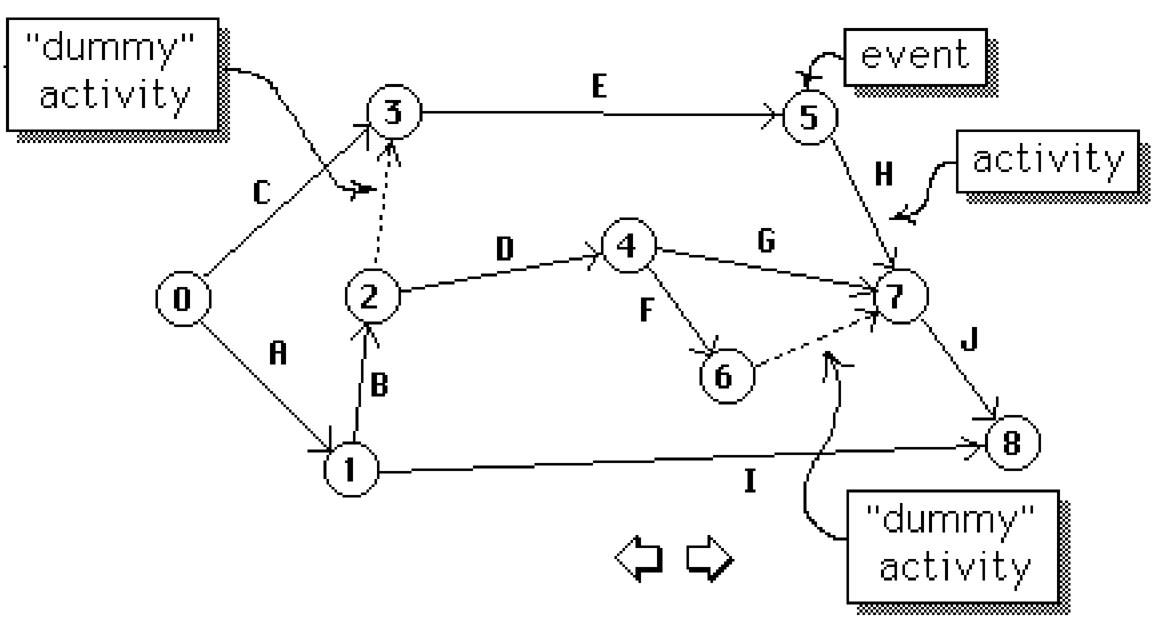
AON (**A**ctivity-**O**n-**N**ode)

AOA (**A**ctivity-**O**n-**A**rrow)

Project Network
(AON - Activity-On-Node)



Project Network
(AOA: Activity-On-Arrow)

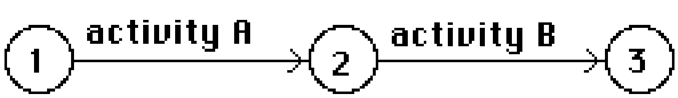


Project Network
(AOA: Activity-On-Arrow)

- a connected, directed network without circuits, with a unique source and a unique sink
- the vertices are called "**events**"
- the arcs are called "**activities**", each with an associated nonnegative **duration**

Predecessors & Successors

The project network indicates the order in which activities may be performed.



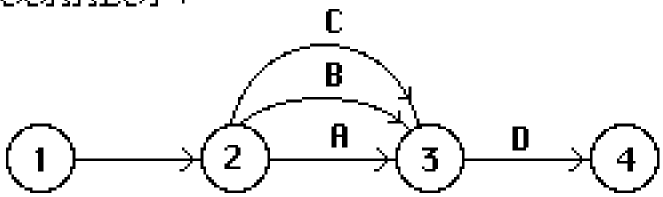
Activity B cannot begin until activity A has been completed

activity A is a predecessor of activity B

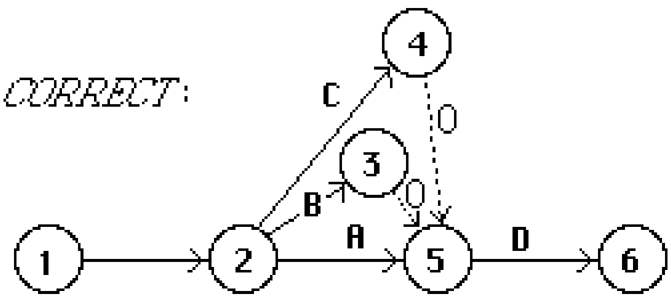
activity B is a successor of activity A

D has predecessors A, B, & C

INCORRECT:



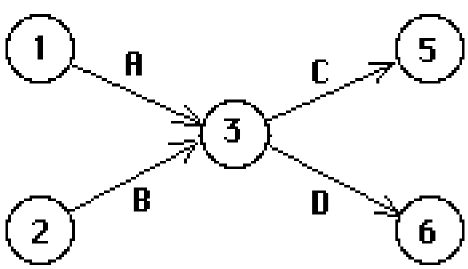
CORRECT:



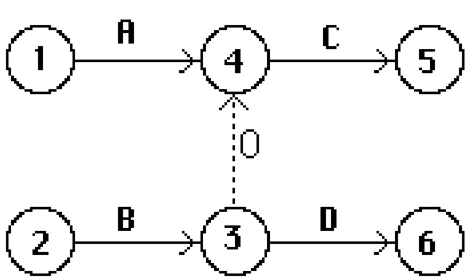
Only one activity is allowed between two vertices; dummy activities may be defined if necessary (with zero duration)

Activities (3,5) and (4,5) are "dummy" activities with zero duration

INCORRECT



CORRECT



A & B are predecessors of C, but only B is a predecessor of D

activity (3,4) is a "dummy" activity with zero duration

Longest Paths

Let the beginning of the project be the event **0**.

Let the end of the project be the event **n**.

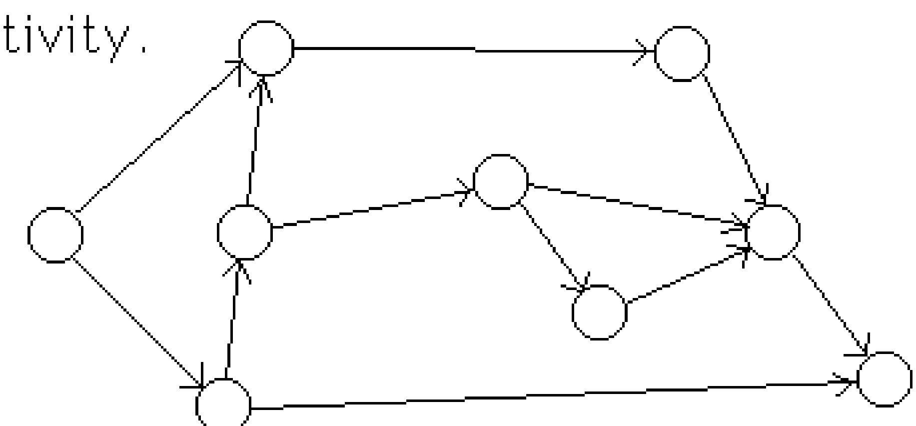
Denote by **ET(i)** the length of the longest path from event **0** to event **i**.

If the project begins at time zero, activity (i,j) can be scheduled to start as early as (but no earlier than) time **ET(i)**

ET(n) = minimum project duration

Labelling Events

It is convenient to label the events (vertices) of the project network so that i<j if (i,j) is an activity.



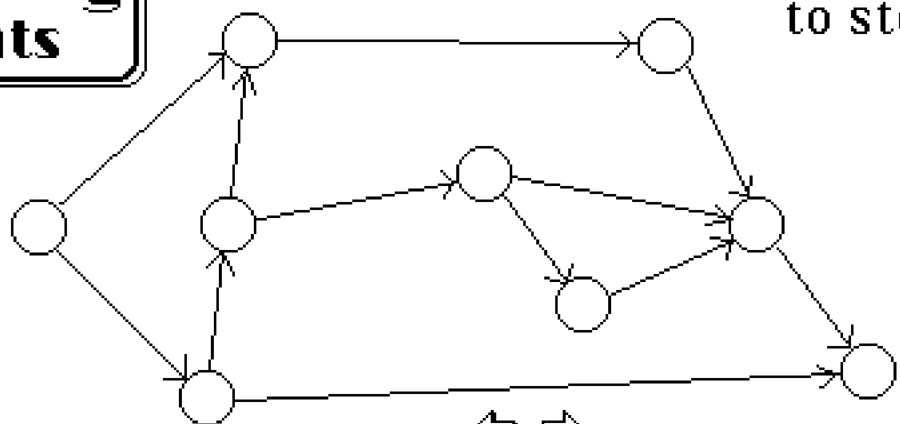
Algorithm

step 0: let j=0

step 1: find a vertex without an unlabelled predecessor. If none, quit; else label this vertex "j"

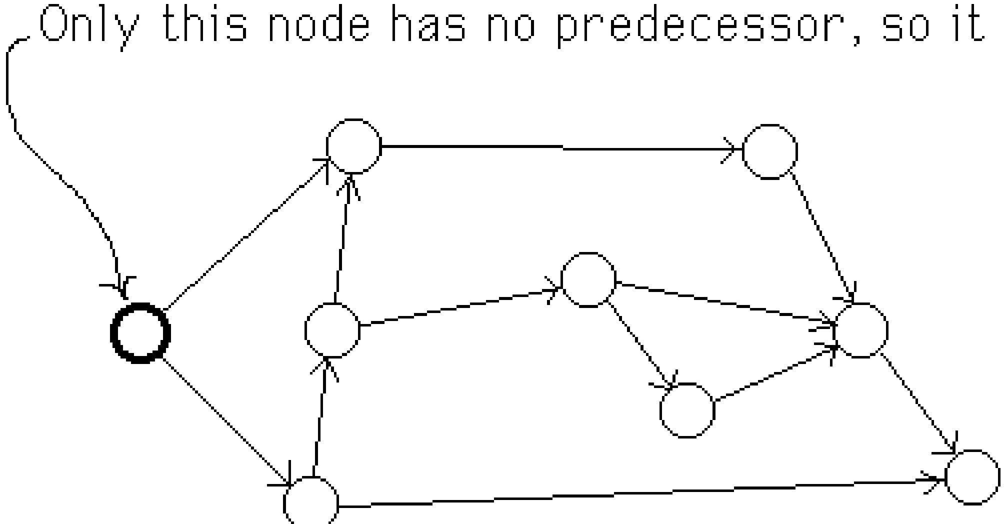
step 2: increment j by 1 and go to step 1.

Labelling Events

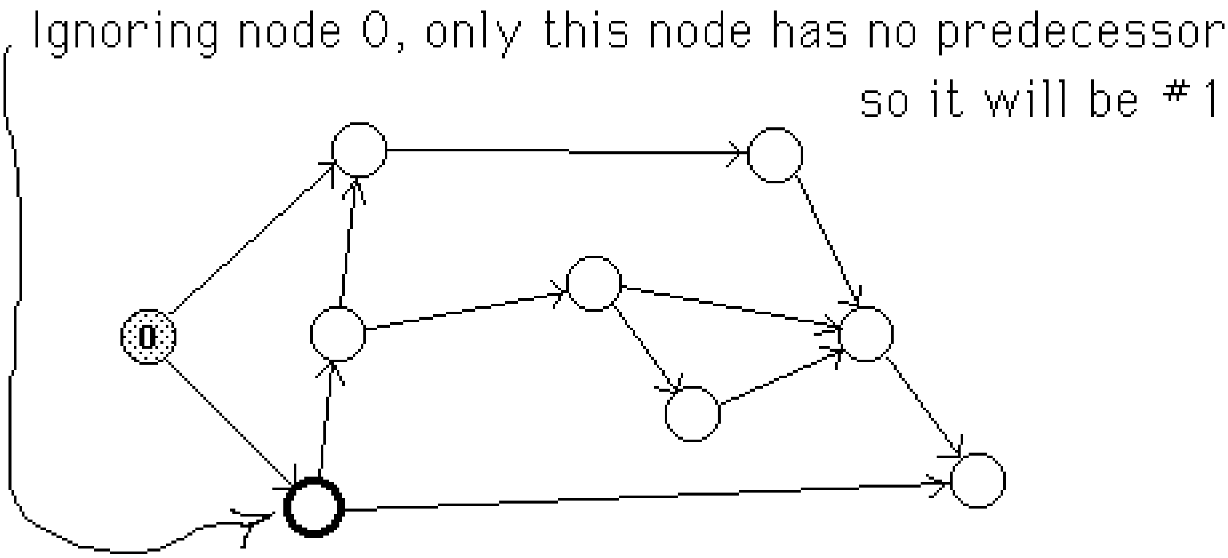


Labelling Events

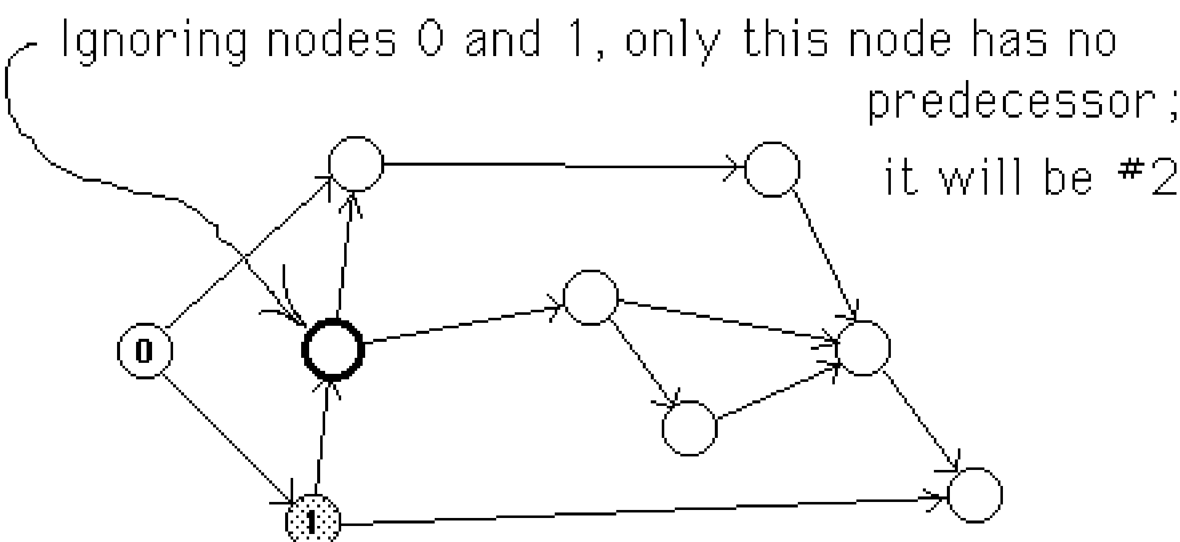
Only this node has no predecessor, so it is labelled 0



Labelling Events

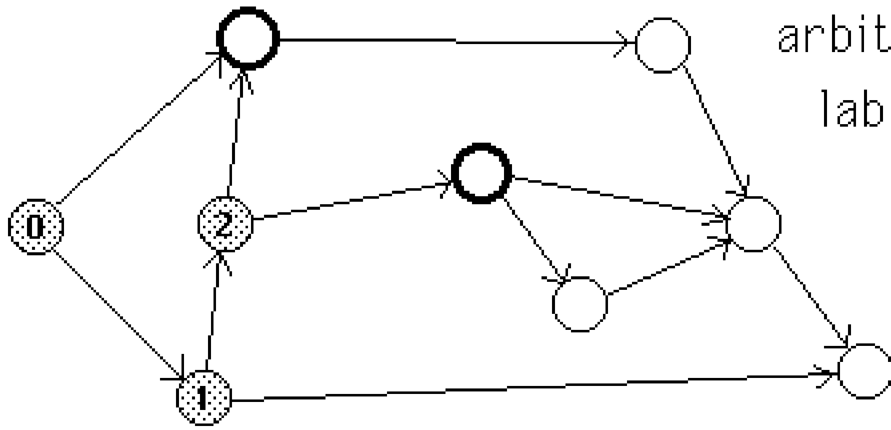


Labelling Events



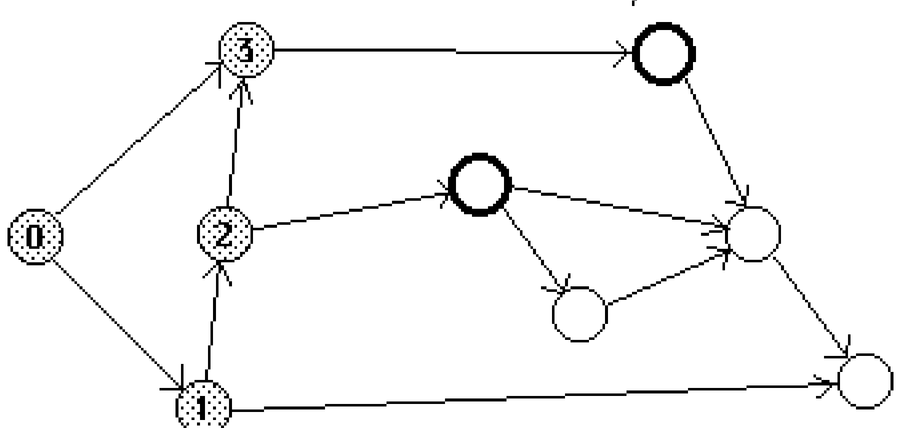
Labelling Events

Ignoring nodes 0,1,&2, there are two nodes having no predecessor; we choose one of them arbitrarily to be labelled #3



Labelling Events

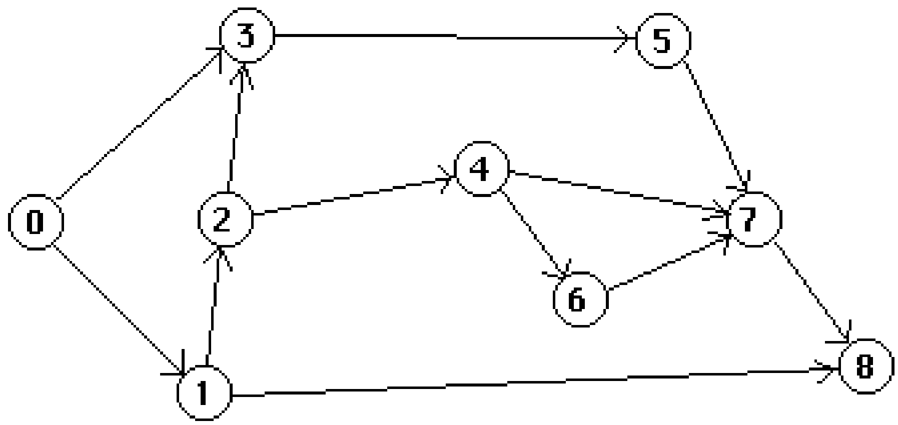
Again, there are two nodes without predecessors; we will choose one arbitrarily to be #4



Labelling Events

(i,j) is an arc
$\Longrightarrow i < j$

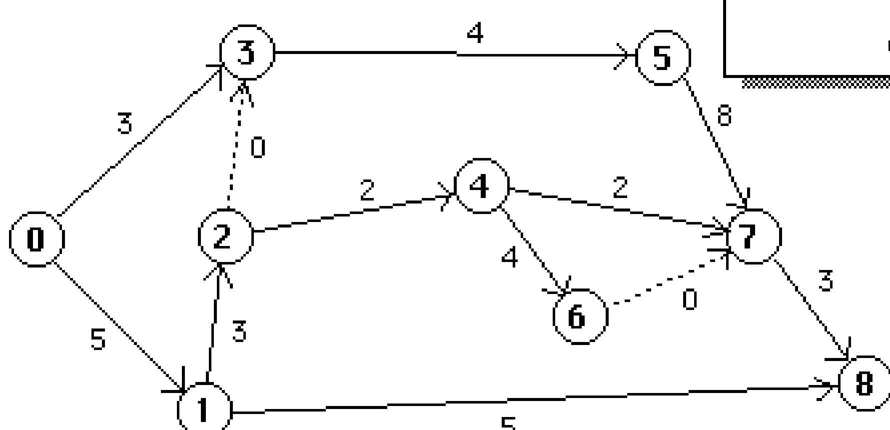
... etc.



Algorithm *"Forward Pass"*

ET(i) = earliest time at which event i can occur

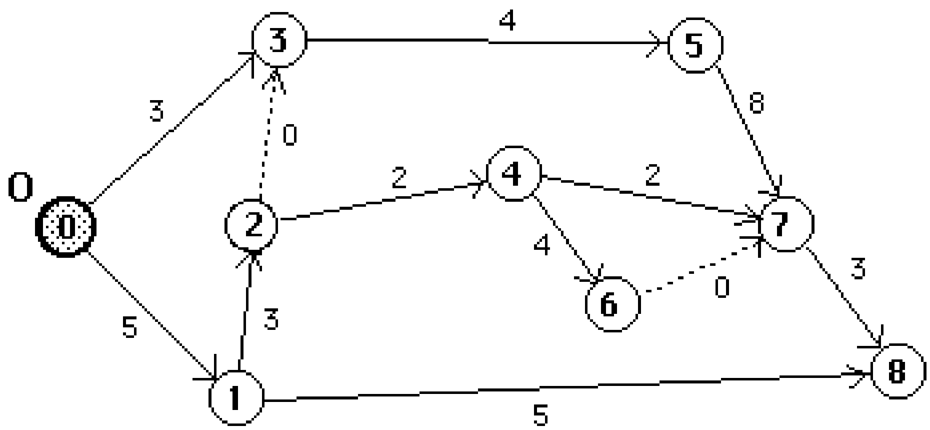
$ET(0)=0$
For $j=1$ to n:
$ET(j) = \max_{(i,j)\in A}\{ET(i)+d_{ij}\}$



Assumes i<j if (i,j) is an arc

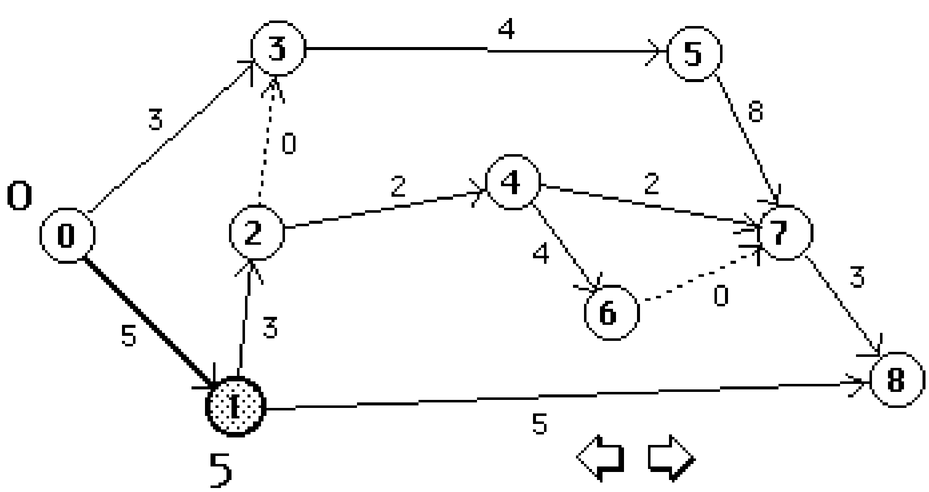
Computing Earliest Time for Events

$ET(0)=0$



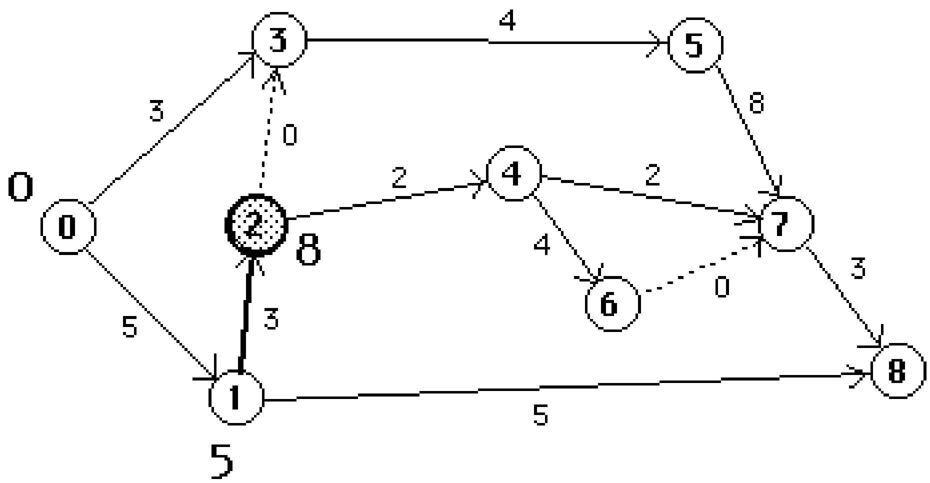
Computing Earliest Time for Events

$ET(1)=ET(0)+5 = 5$



Computing Earliest Time for Events

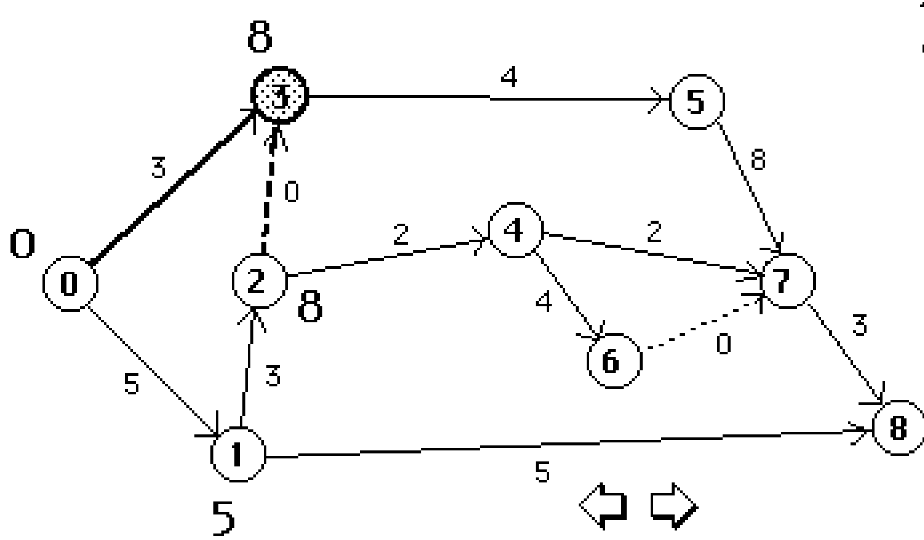
$$ET(2) = ET(1) + 3 = 8$$



Computing Earliest Time for Events

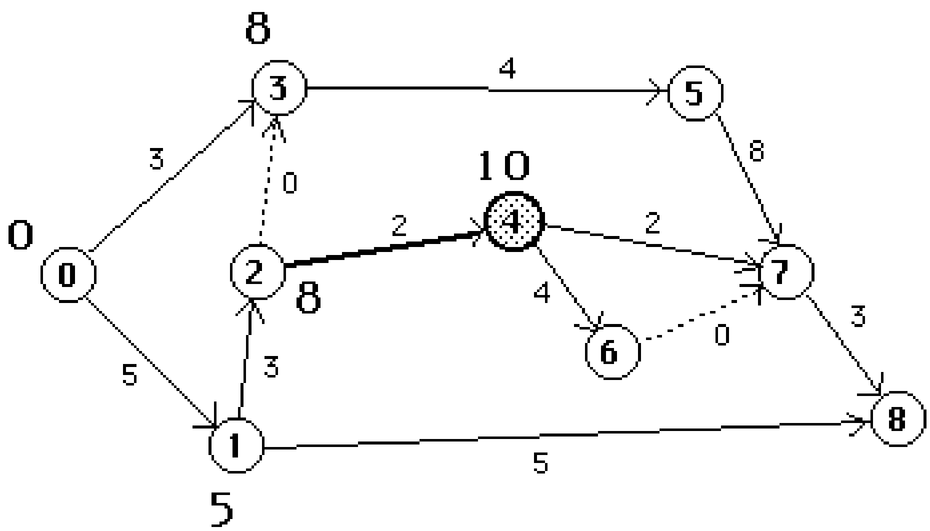
$$ET(3) = \max\{ET(0)+3,\ ET(2)+0\}$$
$$= \max\{3,8\} = 8$$

2 activities enter vertex 3



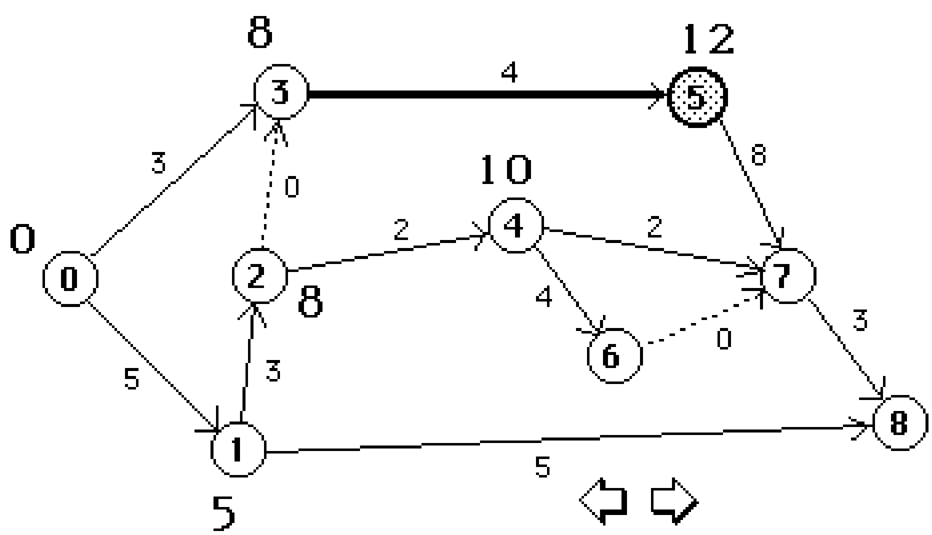
Computing Earliest Time for Events

$$ET(4) = ET(2) + 2 = 10$$



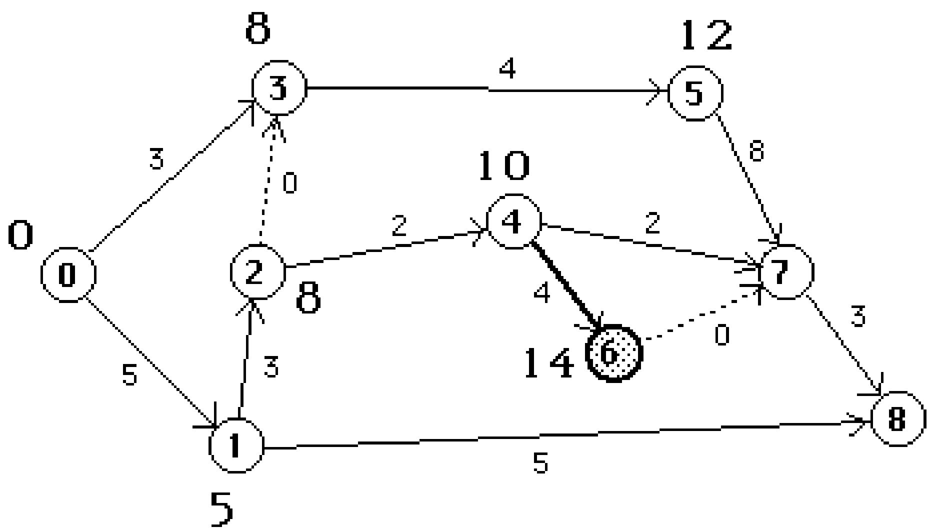
Computing Earliest Time for Events

$$ET(5) = ET(3) + 4 = 12$$



$$ET(6) = ET(4) + 4 = 14$$

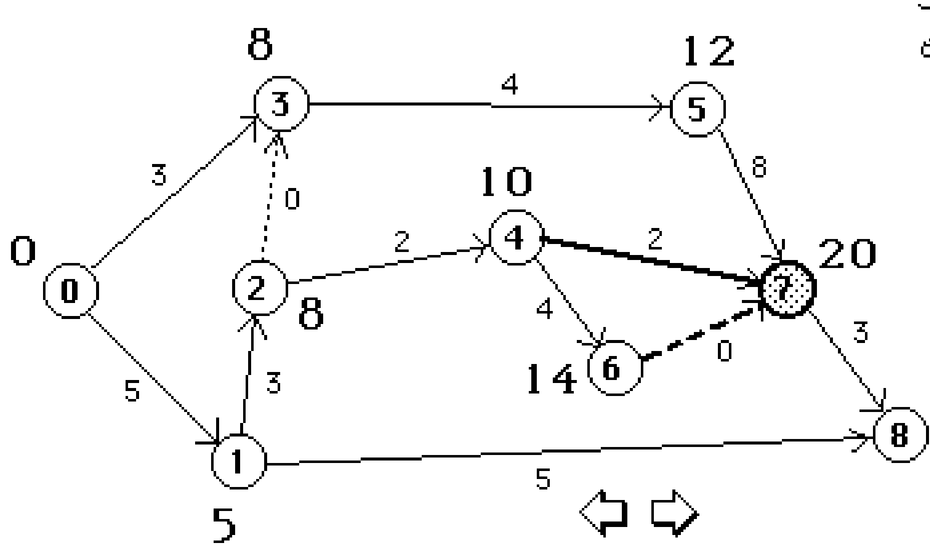
Computing Earliest Time for Events



$$ET(7) = \max\{ET(4)+2,\ ET(6)+0,\ ET(5)+8\}$$
$$= \max\{12,\ 14,\ 20\} = 20$$

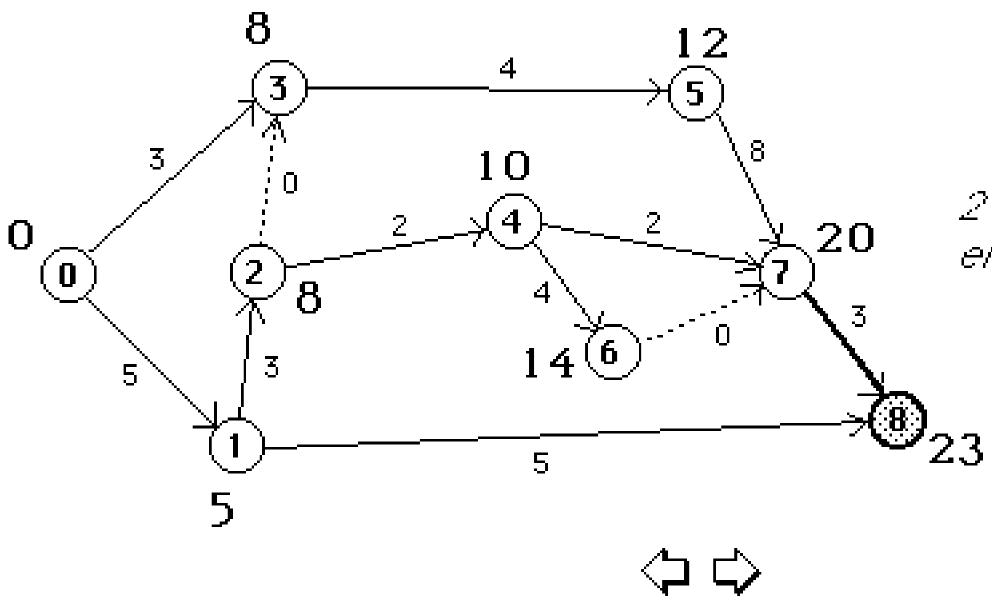
Computing Earliest Time for Events

3 activities enter vertex 7



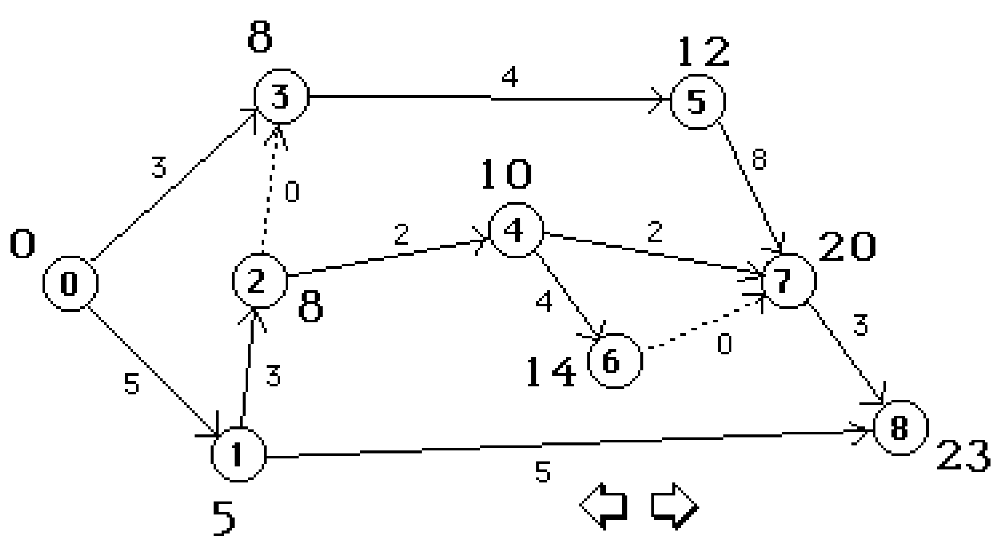
Computing Earliest Time for Events

$$ET(8) = \max\{ET(1)+5,\ ET(7)+3\}$$
$$= \max\{10, 23\} = 23$$



2 activities enter vertex 8

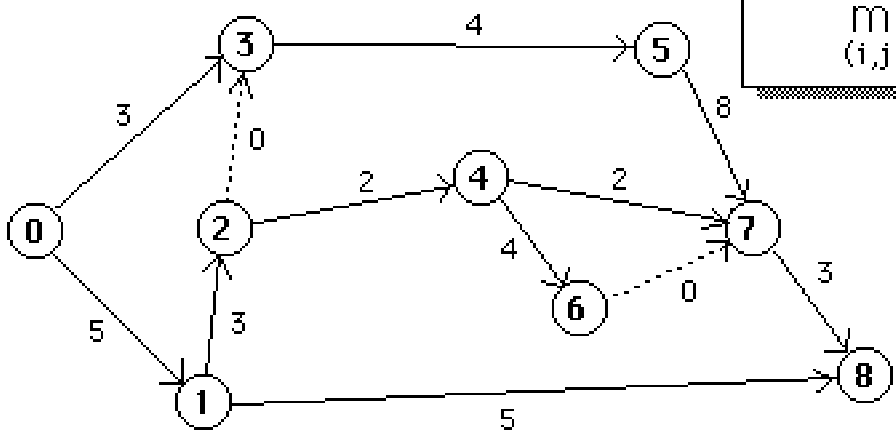
And so the earliest time for completion of the project (event #8) is 23



LT(i) = latest time at which event i can occur if the project is to be completed in minimum time

Algorithm *Backward Pass*

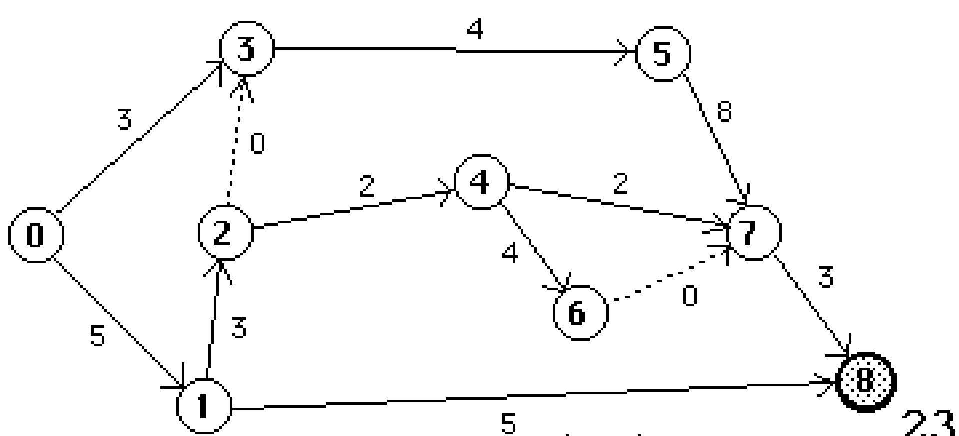
$LT(n) = ET(n)$
For $i = n-1, n-2, \ldots 0$
$LT(i) = \min_{(i,j) \in A}\{LT(j) - d_{ij}\}$



Assumes i<j if (i,j) is an arc

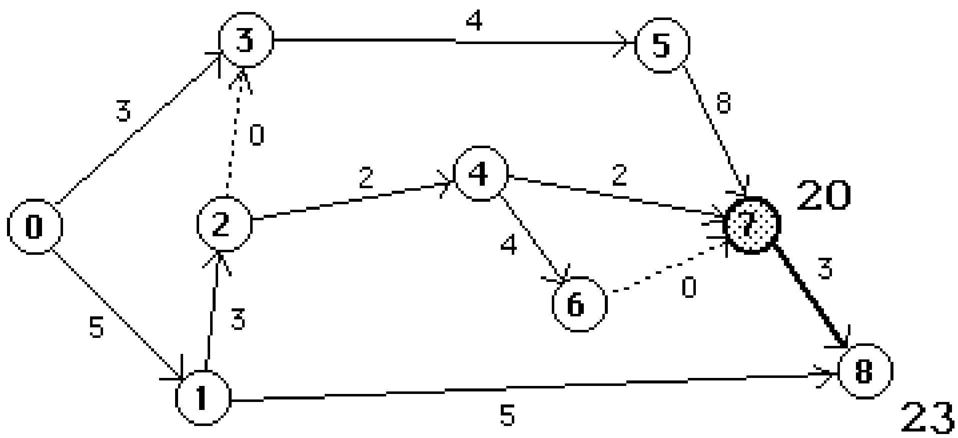
$LT(8) = ET(8) = 23$

Computing Latest Time for Events



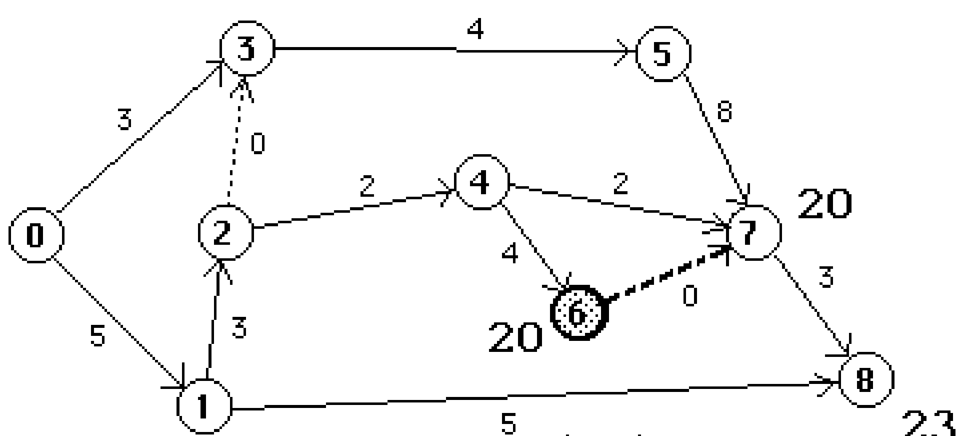
$$LT(7) = LT(8) - 3 = 20$$

Computing Latest Time for Events



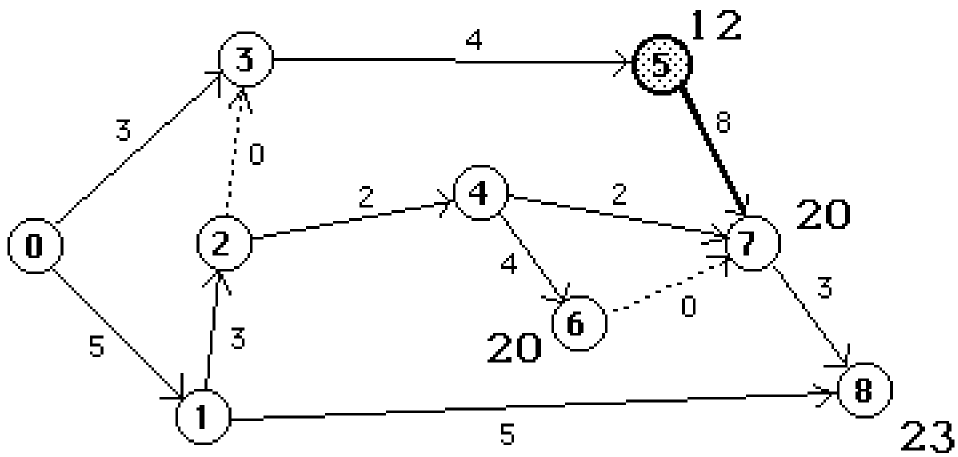
$$LT(6) = LT(7) - 0 = 20$$

Computing Latest Time for Events



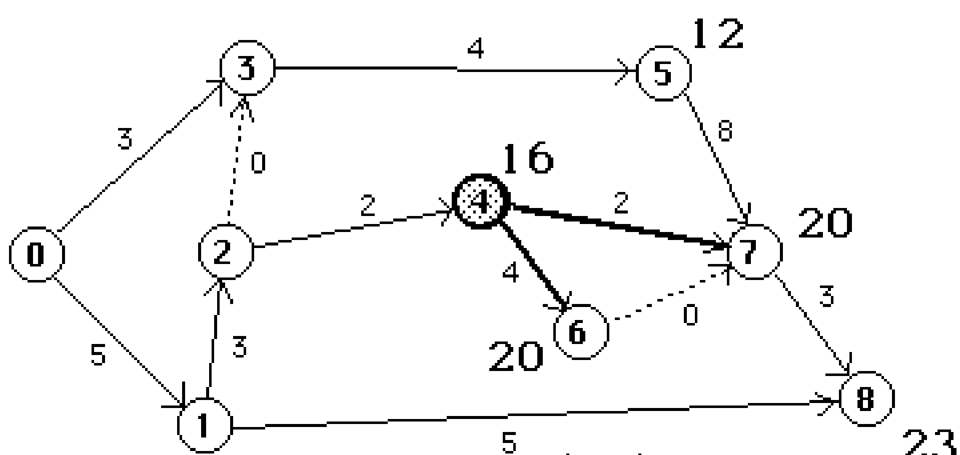
Computing Latest Time for Events

$$LT(5) = LT(7) - 8 = 12$$



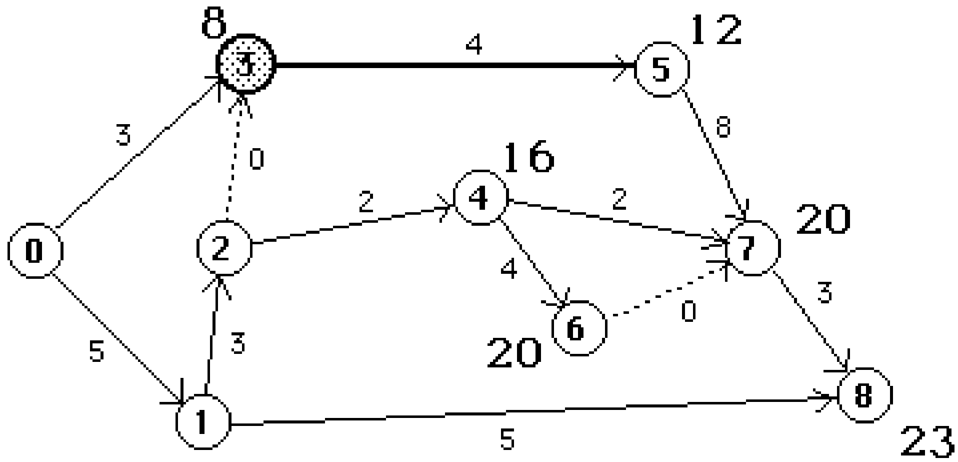
Computing Latest Time for Events

$$LT(4) = \min\{LT(6)-4, LT(7)-2\}$$
$$= \min\{16, 18\} = 16$$



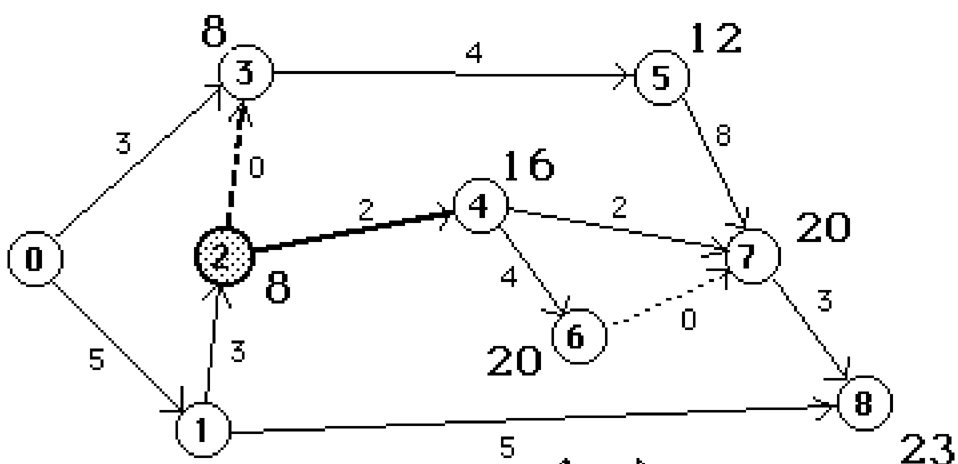
Computing Latest Time for Events

$$LT(3) = LT(5) - 4 = 8$$



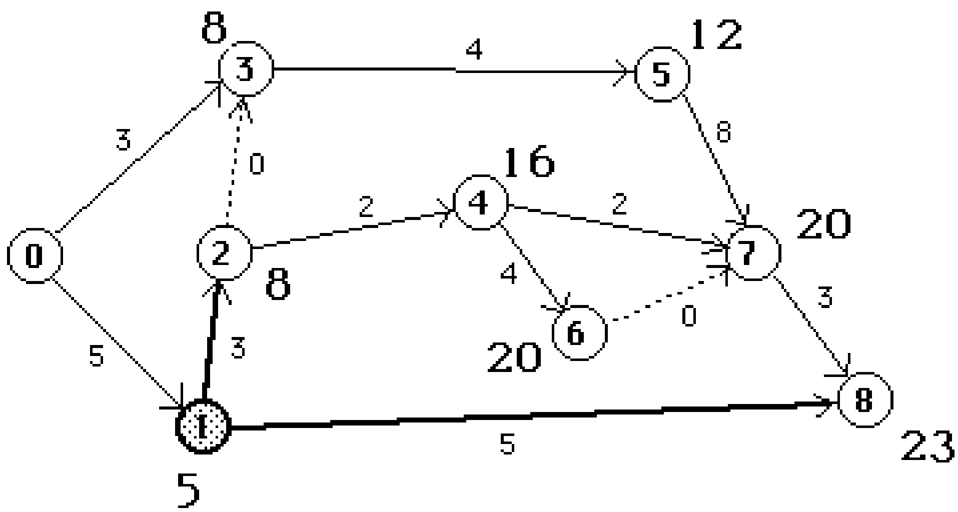
Computing Latest Time for Events

$$LT(2) = \min\{LT(3)-0,\ LT(4)-2\}$$
$$= \min\{8, 14\} = 8$$



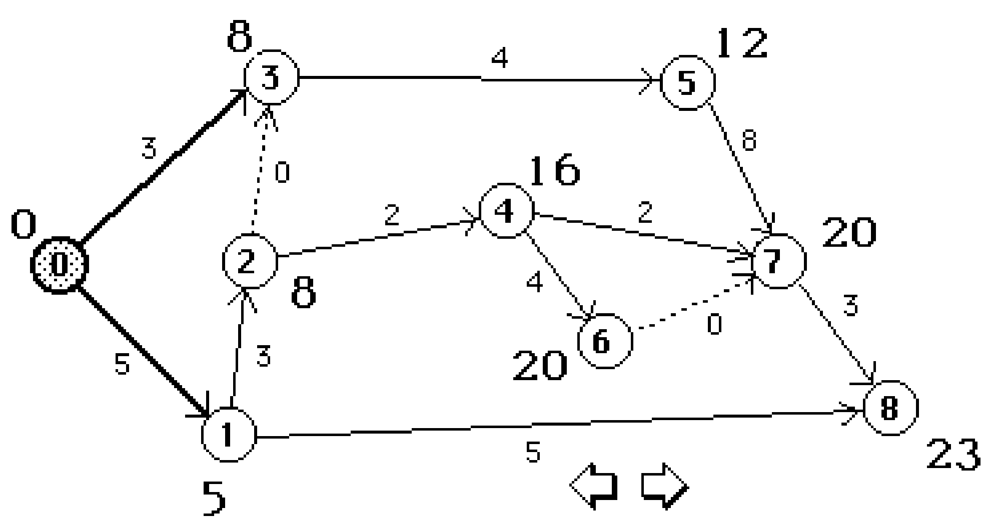
$$LT(1) = \min\{LT(2)-3,\ LT(8)-5\}$$
$$= \min\{5, 18\} = 5$$

Computing Latest Time for Events

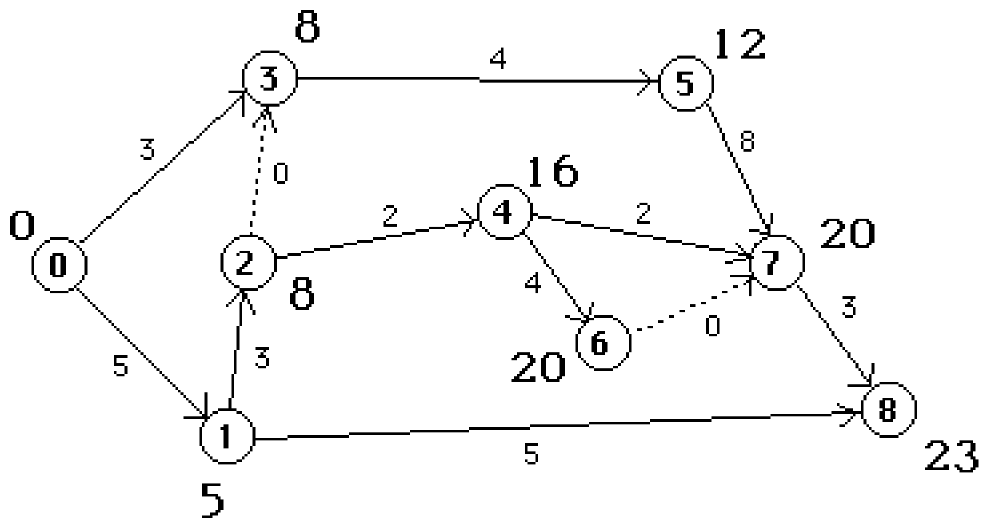


$$LT(0) = \min\{LT(1)-5,\ LT(3)-3\}$$
$$= \min\{0, 5\} = 0$$

Computing Latest Time for Events



(If $LT(0) \neq 0$, then an error was made!)



For each activity, define:

Earliest start time	$ES(i,j) = ET(i)$
Earliest finish time	$EF(i,j) = ET(i) + d_{ij}$
Latest finish time	$LF(i,j) = LT(j)$
Latest start time	$LS(i,j) = LT(j) - d_{ij}$

For each activity, define:

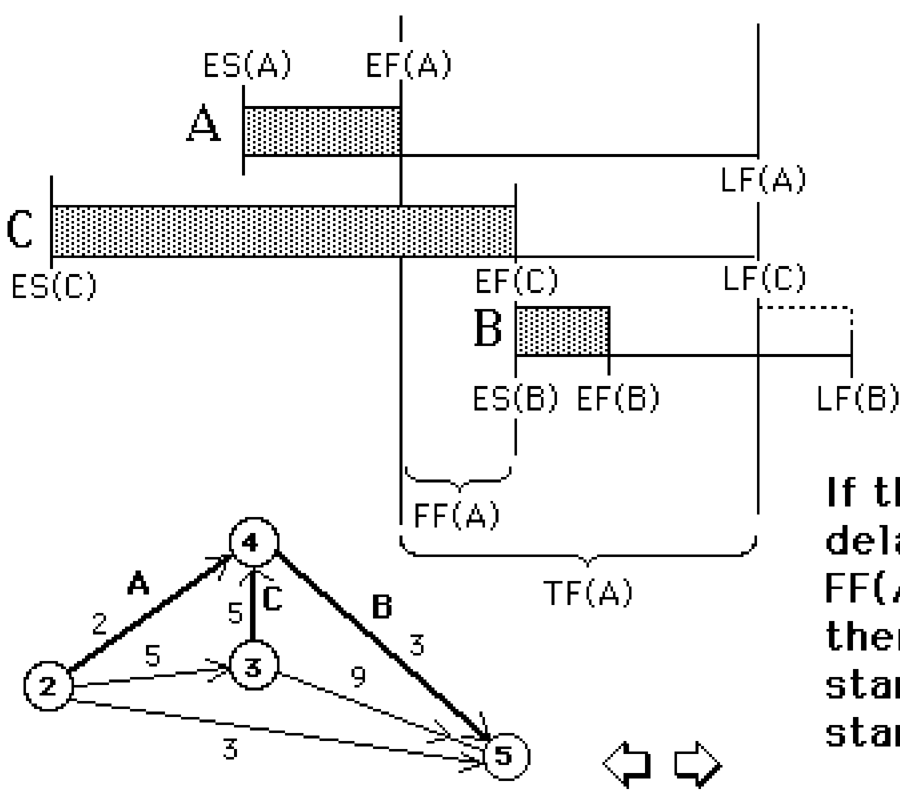
Total float $TF(i,j) = LS(i,j) - ES(i,j)$

Maximum possible time by which the start of the activity may be delayed, without delaying the project completion time.

Free float $FF(i,j) = [ET(j) - d_{ij}] - ET(i)$

*Maximum possible time by which the start may be delayed **IF** all successors start at their Early Start time*

Total Float & Free Float



If the start of A is delayed by more than FF(A), the "free float", then B cannot be started at its early start time, ES(B)

If the total float of an activity is zero, i.e., its Early Start Time=Late Start Time, then the activity is on the Critical Path

"TS" = total slack = total float = "TF"

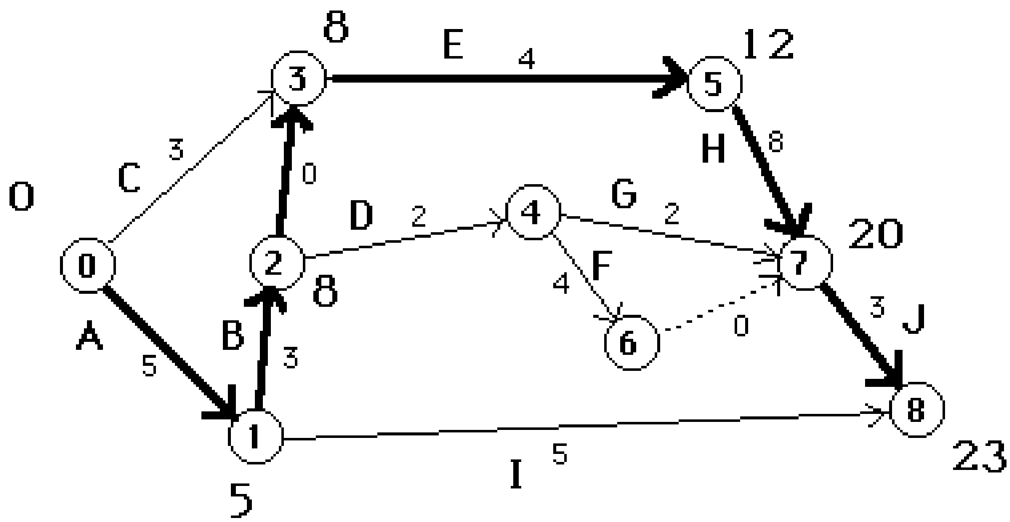
"FS" = free slack = free float = "FF"

Critical path

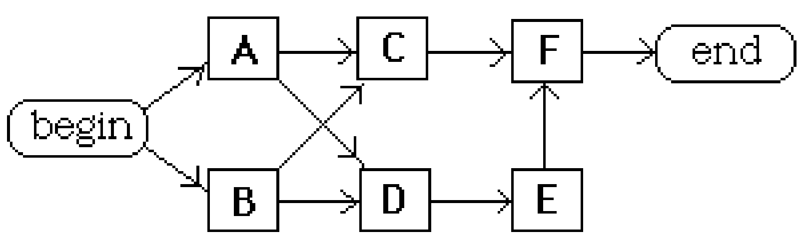
	TASK	I	D	ES	EF	LS	LF	TS	FS
**	Start	1	0	0	0	0	0	0	0
**	A	2	5	0	5	0	5	0	0
**	B	3	3	5	8	5	8	0	0
	C	4	3	0	3	5	8	5	5
	D	5	2	8	10	14	16	6	0
**	E	6	4	8	12	8	12	0	0
	F	7	4	10	14	16	20	6	6
	G	8	2	10	12	18	20	8	8
**	H	9	8	12	20	12	20	0	0
	I	10	5	5	10	18	23	13	13
**	J	11	3	20	23	20	23	0	0
**	End	12	0	23	23	23	23	0	0

The Critical Path

A delay in starting or finishing an activity on the critical path will delay the entire project!



Linear Programming Model



Define Y_i = starting time for activity i

Objective Minimize $Y_{end} - Y_{begin}$

Constraints

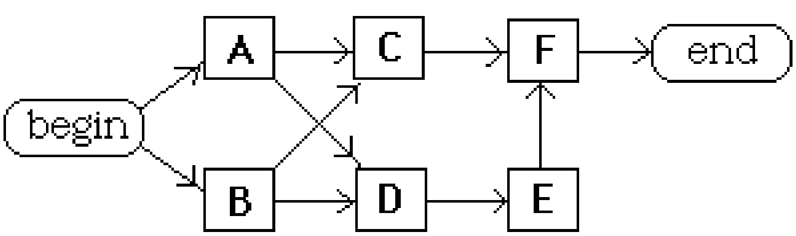
For every predecessor requirement, we will have an inequality constraint:

For example, "A must precede C" translates to

$$Y_C \geq \underbrace{Y_A + d_A}_{\textit{completion time for activity A}}$$

where d_A is the duration of activity A.

LP Model



$$\text{Minimize } Y_{end} - Y_{begin}$$

subject to

$$Y_A \geq Y_{begin}$$
$$Y_B \geq Y_{begin}$$
$$Y_C \geq Y_A + d_A$$
$$Y_C \geq Y_B + d_B$$
$$Y_D \geq Y_A + d_A$$
$$Y_D \geq Y_B + d_B$$
$$\vdots$$
$$Y_{end} \geq Y_F + d_F$$

Y_i unrestricted in sign

$$\text{Minimize } Y_{end} - Y_{begin}$$

subject to
$$Y_A - Y_{begin} \geq 0$$
$$Y_B - Y_{begin} \geq 0$$
$$Y_C - Y_A \geq d_A$$
$$Y_C - Y_B \geq d_B$$
$$Y_D - Y_A \geq d_A$$
$$Y_D - Y_B \geq d_B$$
$$\vdots$$
$$Y_{end} - Y_F \geq d_F$$

Y_i unrestricted in sign

Transferring all variables to the left-hand-side

Now we wish to write the Dual of this LP!

The Dual Variables

There will be a dual variable X_{ij} for every precedence restriction of the form "activity i must precede activity j"

The Dual Objective

$$\text{Maximize } d_A X_{AC} + d_B X_{BC} + \dots + d_F X_{F,end}$$

The Dual Constraints

There will be a dual constraint for every variable in the primal:
For example, corresponding to variable Y_A is the constraint:

$$X_{begin,A} - X_{AC} - X_{AD} = 0$$

Maximize $d_A X_{AC} + d_B X_{BC} + d_A X_{AD} + \cdots\cdots\cdots + d_F X_{F,end}$
subject to

$$\begin{array}{rcl}
-X_{begin,A} - X_{begin,B} & = & -1 \\
X_{begin,A} - X_{AC} - X_{AD} & = & 0 \\
X_{begin,B} - X_{BC} - X_{BD} & = & 0 \\
X_{AC} + X_{BC} - X_{CF} & = & 0 \\
X_{AD} + X_{BD} - X_{DE} & = & 0 \\
& \vdots & \\
X_{F,end} & = & 1
\end{array}$$

$$X_{ij} \geq 0 \ \forall \ (i,j)$$

The Dual LP

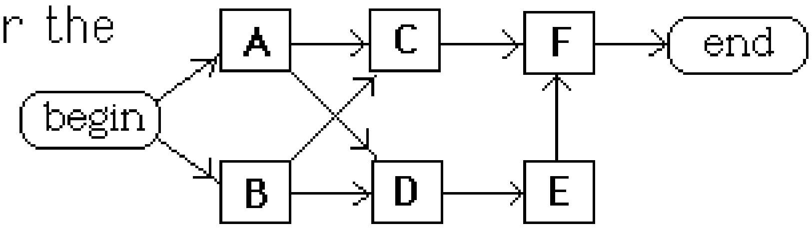
Maximize $d_A X_{AC} + d_B X_{BC} + d_A X_{AD} + \cdots\cdots + d_F X_{F,end}$

subject to

$$
\begin{array}{lllllllll}
-X_{begin,A} & -X_{begin,B} & & & & & & & = -1 \\
X_{begin,A} & & -X_{AC} & -X_{AD} & & & & & = 0 \\
 & X_{begin,B} & & & -X_{BC} & -X_{BD} & & & = 0 \\
 & & X_{AC} & & +X_{BC} & & -X_{CF} & & = 0 \\
 & & & X_{AD} & & +X_{BD} & & -X_{DE} & = 0 \\
 & & & & & & & & \vdots \\
 & & & & & & & X_{F,end} & = 1
\end{array}
$$

$X_{ij} \geq 0 \; \forall (i,j)$

The constraints of the dual LP are conservation of flow equations for the AON network:



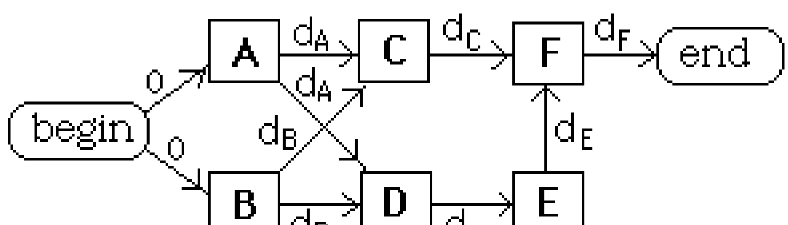
Maximize $d_A X_{AC} + d_B X_{BC} + d_A X_{AD} + \cdots\cdots + d_F X_{F,end}$

subject to

$$
\begin{array}{lllllllll}
-X_{begin,A} & -X_{begin,B} & & & & & & & = -1 \\
X_{begin,A} & & -X_{AC} & -X_{AD} & & & & & = 0 \\
 & X_{begin,B} & & & -X_{BC} & -X_{BD} & & & = 0 \\
 & & X_{AC} & & +X_{BC} & & -X_{CF} & & = 0 \\
 & & & X_{AD} & & +X_{BD} & & -X_{DE} & = 0 \\
 & & & & & & & & \vdots \\
 & & & & & & & X_{F,end} & = 1
\end{array}
$$

$X_{ij} \geq 0 \; \forall (i,j)$

The dual LP is the problem of finding the ***longest*** path through the network from "begin" to "end"



Job	Immediate Predecessor(s)	Normal time
A	none	5
B	A	6
C	A	10
D	A	7
E	B	3
F	C,E	3
G	C	2
H	D	6
I	none	10

- Draw a network for the project
- determine the critical path & project duration.

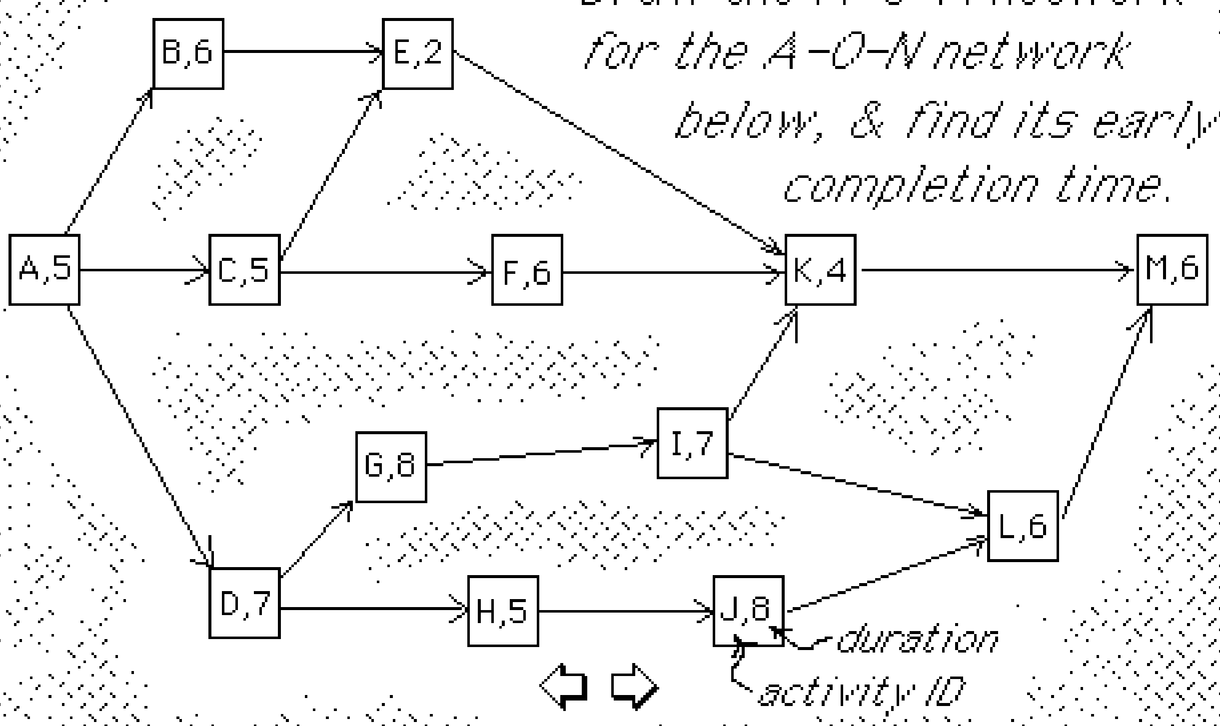
Job	Immediate Predecessor(s)	Normal time
A	none	3
B	none	5
C	none	4
D	none	3
E	A	6
F	C, H	7
G	E	4
H	B, E	5
I	C, H	6
J	H	4
K	G, H	4
L	I, J	2
M	D, F	5

- Draw a network for the project
- determine the critical path & project duration.

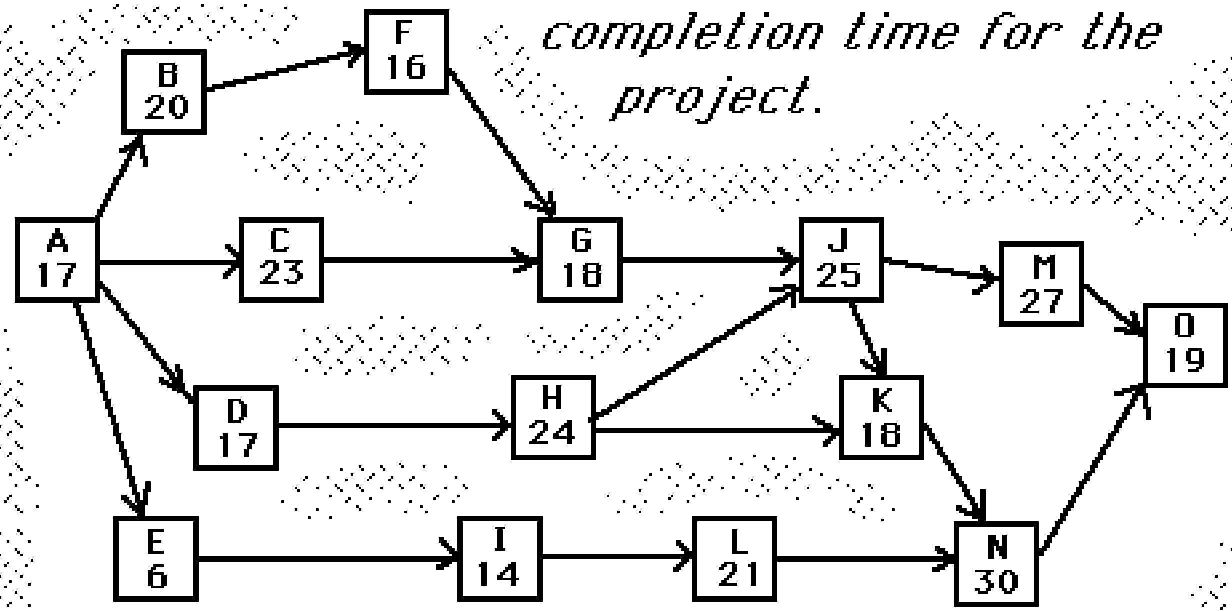
Job	Immediate Predecessor(s)	Normal time
A	none	9
B	A	8
C	A	8
D	B	6
E	C,G	12
F	A	12
G	F	5
H	G	8
I	D,H,E	7
J	D	10

- Draw a network for the project
- determine the critical path & project duration.

Draw the A-O-A network for the A-O-N network below, & find its early completion time.



Draw the A-O-A network corresponding to the A-O-N network below... & find the earliest completion time for the project.



A pipeline construction project

Task	*Description*	*Immediate predecessor(s)*	*Time*
A	Lead time	none	10
B	Equipment to site	A	20
C	Get pipe	A	40
D	Get valve	A	28
E	Lay out line	B	8
F	Excavate	E	30
G	Test pipe	C	3
H	Lay pipe	F,G	24
I	Concrete work	H	12
J	Install valve	D	10
K	Test pipe	I,J	6
L	Cover pipe	I,J	10
M	Clean up	K,L	4
N	Complete valve work	I,J	6
O	Leave site	M,N	4