## Example Project

<table>
<thead>
<tr>
<th>task</th>
<th>predecessor</th>
<th>duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>none</td>
<td>5</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>none</td>
<td>3</td>
</tr>
<tr>
<td>D</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>B,C</td>
<td>4</td>
</tr>
<tr>
<td>F</td>
<td>D</td>
<td>4</td>
</tr>
<tr>
<td>G</td>
<td>D</td>
<td>2</td>
</tr>
<tr>
<td>H</td>
<td>E</td>
<td>8</td>
</tr>
<tr>
<td>I</td>
<td>A</td>
<td>5</td>
</tr>
<tr>
<td>J</td>
<td>F,G,H</td>
<td>3</td>
</tr>
</tbody>
</table>

A project has two network representations:

**AON (Activity-On-Node)**

**AOA (Activity-On-Arrow)**
Project Network
(AON - Activity-On-Node)
Project Network
(Analogy: Activity-On-Arrow)

©D. Brickner, U. of I., 2000
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.Brickle, U.of I,A,2000
D has predecessors A, B, & C

INCORRECT:

1 \rightarrow 2 \rightarrow 3 \rightarrow D

CORRECT:

1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6

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

©D.Bricker, U.of IA,2000

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

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

**CORRECT**

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

©D.Brickler, U.of I& 2000
Longest Paths

Let the beginning of the project be the event \( O \).
Let the end of the project be the event \( n \).

Denote by \( ET(i) \) the length of the longest path from event \( O \) 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

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

Ignoring node 0, only this node has no predecessor so it will be #1
Labelling Events

Ignoring nodes 0 and 1, only this node has no predecessor; it will be #2
Labelling Events

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

©D. Bricker, U. of IA, 2000
Labelling Events

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

... etc.

(i,j) is an arc
⇒ i < j
**Algorithm "Forward Pass"**

\[ ET(i) = \text{earliest time at which event } i \text{ can occur} \]

ET(0) = 0

For j = 1 to n:

\[ ET(j) = \max\{ET(i) + d_{ij}\} \quad (i,j) \in A \]

Assumes i<j if (i,j) is an arc.
ET(0) = 0

Computing Earliest Time for Events
\[ \text{ET}(1) = \text{ET}(0) + 5 = 5 \]
$ET(2) = ET(1) + 3 = 8$
\[ ET(3) = \max\{ET(0)+3, ET(2)+0\} = \max\{3,8\} = 8 \]
$ET(4) = ET(2) + 2 = 10$
ET(5) = ET(3) + 4 = 12
\[ ET(6) = ET(4) + 4 = 14 \]
\[
ET(7) = \max\{ET(4)+2, \ ET(6)+0, \ ET(5)+8\} \\
= \max\{12, 14, 20\} = 20
\]
\[ ET(8) = \max\{ET(1) + 5, ET(7) + 3\} \]
\[ = \max\{10, 23\} = 23 \]
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

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

Assumes \( i \leq j \) if \( (i,j) \) is an arc

\[ \text{©D. Bricker, U of IA, 2000} \]
LT(8) = ET(8) = 23
\[ LT(7) = LT(8) - 3 = 20 \]
\[ LT(6) = LT(7) - 0 = 20 \]
\[ LT(5) = LT(7) - 8 = 12 \]
\[ LT(4) = \min\{ LT(6) - 4, LT(7) - 2\} \]
\[ = \min\{16, 18\} = 16 \]
\[ LT(3) = LT(5) - 4 = 8 \]
\[ LT(2) = \min\{LT(3) - 0, LT(4) - 2\} \]
\[ = \min\{8,14\} = 8 \]
\[
\text{LT}(1) = \min\{\text{LT}(2)-3, \text{LT}(8)-5\} \\
= \min\{5, 18\} = 5
\]
\[ LT(0) = \min\{LT(1) - 5, LT(3) - 3\} \]
\[ = \min\{0, 5\} = 0 \]
(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"

<table>
<thead>
<tr>
<th>TASK</th>
<th>I</th>
<th>D</th>
<th>ES</th>
<th>EF</th>
<th>LS</th>
<th>LF</th>
<th>TS</th>
<th>FS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start</strong></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>A</strong></td>
<td>2</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>8</td>
<td>5</td>
<td>8</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>C</strong></td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>5</td>
<td>8</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td><strong>D</strong></td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>10</td>
<td>14</td>
<td>16</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td><strong>E</strong></td>
<td>6</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>8</td>
<td>12</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>F</strong></td>
<td>7</td>
<td>4</td>
<td>10</td>
<td>14</td>
<td>16</td>
<td>20</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td><strong>G</strong></td>
<td>8</td>
<td>2</td>
<td>10</td>
<td>12</td>
<td>18</td>
<td>20</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td><strong>H</strong></td>
<td>9</td>
<td>8</td>
<td>12</td>
<td>20</td>
<td>12</td>
<td>20</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>I</strong></td>
<td>10</td>
<td>5</td>
<td>5</td>
<td>10</td>
<td>18</td>
<td>23</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td><strong>J</strong></td>
<td>11</td>
<td>3</td>
<td>20</td>
<td>23</td>
<td>20</td>
<td>23</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>End</strong></td>
<td>12</td>
<td>0</td>
<td>23</td>
<td>23</td>
<td>23</td>
<td>23</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

©D.Bricke, U.of I.2000
The Critical Path

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

Define \( Y_i = \text{starting time for activity } i \)

**Objective**  \( \text{Minimize } Y_{\text{end}} - Y_{\text{begin}} \)
Constraints

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

For example, "A must precede C" translates to

\[ Y_C \geq Y_A + d_A \]

(completion time for activity A)

where \( d_A \) is the duration of activity A.
LP Model

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
Now we wish to write the Dual of this LP.

Y, unrestricted in sign

Minimize \( Y A - Y \begin{array}{l}
Y B - Y \begin{array}{l}
Y C - Y \begin{array}{l}
Y D - Y \begin{array}{l}
Y E = \text{Y begin} \\
\geq 0
\end{array}
\end{array}
\end{array}
\end{array}
\end{array}
\)

subject to

\[ Y \text{ and } Y \begin{array}{l}
\leq d_a
\end{array}
\]

...
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

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

$$X_{\text{begin},A} - X_{AC} - X_{AD} = 0$$
Maximize \( d_A X_{AC} + d_B X_{BC} + d_A X_{AD} + \ldots \ldots + d_F X_{F,\text{end}} \)

subject to

\[
\begin{align*}
-X_{\text{begin},A} - X_{\text{begin},B} & = -1 \\
X_{\text{begin},A} - X_{AC} - X_{AD} & = 0 \\
X_{\text{begin},B} - X_{BC} - X_{BD} & = 0 \\
X_{AC} + X_{BC} - X_{CF} & = 0 \\
X_{AD} + X_{BD} - X_{DE} & = 0 \\
& \vdots \\
X_{F,\text{end}} & = 1 \\
X_{ij} & \geq 0 \; \forall (i,j)
\end{align*}
\]
Maximize \( d_A X_{AC} + d_B X_{BC} + d_A X_{AD} + \ldots \) + \( d_F X_{F,\text{end}} \)

subject to

\[
\begin{align*}
-X_{\text{begin},A} - X_{\text{begin},B} & = -1 \\
X_{\text{begin},A} & - X_{AC} - X_{AD} \\
X_{\text{begin},B} & - X_{BC} - X_{BD} \\
X_{AC} & + X_{BC} - X_{CF} \\
X_{AD} & + X_{BD} - X_{DE} \\
\vdots & \\
X_{F,\text{end}} & = 1
\end{align*}
\]

The constraints of the dual LP are conservation \( x_{ij} \geq 0 \forall (i,j) \) of flow equations for the AON network:

\[\text{begin} \rightarrow \text{A} \rightarrow \text{C} \rightarrow \text{F} \rightarrow \text{end} \]

\[\text{begin} \rightarrow \text{B} \rightarrow \text{D} \rightarrow \text{E} \]
Maximize \( d_A X_{AC} + d_B X_{BC} + d_A X_{AD} + \cdots + d_F X_{F,end} \)
subject to
\[
\begin{align*}
-X_{\text{begin,A}} - X_{\text{begin,B}} &= -1 \\
X_{\text{begin,A}} - X_{AC} - X_{AD} &= 0 \\
X_{\text{begin,B}} - X_{BC} - X_{BD} &= 0 \\
X_{AC} + X_{BC} - X_{CF} &= 0 \\
X_{AD} + X_{BD} - X_{DE} &= 0 \\
& \vdots \\
X_{F,end} &= 1
\end{align*}
\]

The dual LP is the problem of finding the \textit{longest} path through the network from "begin" to "end".
### Critical Path Method

#### Immediate Predecessor(s) and Normal Time

<table>
<thead>
<tr>
<th>Job</th>
<th>Immediate Predecessor(s)</th>
<th>Normal Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>none</td>
<td>5</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>6</td>
</tr>
<tr>
<td>C</td>
<td>A</td>
<td>10</td>
</tr>
<tr>
<td>D</td>
<td>A</td>
<td>7</td>
</tr>
<tr>
<td>E</td>
<td>B</td>
<td>3</td>
</tr>
<tr>
<td>F</td>
<td>C, E</td>
<td>3</td>
</tr>
<tr>
<td>G</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>H</td>
<td>D</td>
<td>6</td>
</tr>
<tr>
<td>I</td>
<td>none</td>
<td>10</td>
</tr>
</tbody>
</table>

- **Draw a network for the project**
- **determine the critical path & project duration**
<table>
<thead>
<tr>
<th>Job</th>
<th>Immediate Predecessor(s)</th>
<th>Normal time</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>none</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>none</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>none</td>
<td>4</td>
</tr>
<tr>
<td>D</td>
<td>none</td>
<td>3</td>
</tr>
<tr>
<td>E</td>
<td>A</td>
<td>6</td>
</tr>
<tr>
<td>F</td>
<td>C, H</td>
<td>7</td>
</tr>
<tr>
<td>G</td>
<td>E</td>
<td>4</td>
</tr>
<tr>
<td>H</td>
<td>B, E</td>
<td>5</td>
</tr>
<tr>
<td>I</td>
<td>C, H</td>
<td>6</td>
</tr>
<tr>
<td>J</td>
<td>H</td>
<td>4</td>
</tr>
<tr>
<td>K</td>
<td>G, H</td>
<td>4</td>
</tr>
<tr>
<td>L</td>
<td>I, J</td>
<td>2</td>
</tr>
<tr>
<td>M</td>
<td>D, F</td>
<td>5</td>
</tr>
</tbody>
</table>

- Draw a network for the project
- Determine the critical path & project duration.
<table>
<thead>
<tr>
<th>Job</th>
<th>Immediate Predecessor(s)</th>
<th>Normal time</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>none</td>
<td>9</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>8</td>
</tr>
<tr>
<td>C</td>
<td>A</td>
<td>8</td>
</tr>
<tr>
<td>D</td>
<td>B</td>
<td>6</td>
</tr>
<tr>
<td>E</td>
<td>C,G</td>
<td>12</td>
</tr>
<tr>
<td>F</td>
<td>A</td>
<td>12</td>
</tr>
<tr>
<td>G</td>
<td>F</td>
<td>5</td>
</tr>
<tr>
<td>H</td>
<td>G</td>
<td>8</td>
</tr>
<tr>
<td>I</td>
<td>D,H,E</td>
<td>7</td>
</tr>
<tr>
<td>J</td>
<td>D</td>
<td>10</td>
</tr>
</tbody>
</table>

- 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

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