IO543: Scheduling

Lecture 9

Brian Denton
Associate Professor, IOE

IOE Bld 2893
btdenton@umich.edu

Today

More Parallel Machine Models:

- Makespan with preemptions: $Pm \mid prmp \mid C_{max}$
- Total completion time: $Pm \mid \mid \sum_j C_j$, $Pm \mid \mid \sum_j w_j C_j$
- Project scheduling: $P\infty \mid prec \mid C_{max}$

Makespan With Preemptions

- The addition of **preemptions** makes many scheduling problems easier
- Allowing for preemption is a **relaxation** of the nonpreemptive problem

Makespan With Preemptions

<u>Lemma 5.2.2 (Pinedo):</u> Under the optimal schedule for $Pm\,|prmp\,|C_{max}$

$$C^*_{max} = max(p_1, \sum_j p_j/m)$$

Proof: follows from fact that p_1 has the longest processing time and using the LP formulation of $Pm\,|prmp\,|C_{max}$.

A **corollary** to this is that $max(p_1, \sum_j p_j/m)$ is a lower bound for $Pm\,|\,|C_{max}$

Makespan With Preemptions

Algorithm for $Pm\,|prmp\,|C_{max}$:

Step 1: Process the n jobs **<u>in any order</u>** on a single machine to get a schedule with makespan $\sum_j p_j \leq mC^*_{max}$

Step 2: Break the single machine schedule into m parts: $[0, C^*_{max}]$, $[C^*_{max}, 2C^*_{max}]$, $[2C^*_{max}, 3C^*_{max}]$, ...

Step 3: Use the processing sequence in the first interval for machine 1, the second interval for machine 2, etc. (<u>note: order of jobs is not important, only the order of the machines matters</u>)

Makespan With Preemptions

Example: Solve the following instance of $P3 \mid prmp \mid C_{max}$

Jobs	1	2	3	4	5	6	7	8
p_j	4	6	3	2	9	5	4	9

$\mathrm{P}m \mid\mid \sum_j C_j$

Important result for $\mathrm{P}m \mid\mid \sum_j C_j$ without preemptions

<u>Theorem 5.3.1 (Pinedo):</u> The SPT rule is optimal for $\mathrm{P}m \mid\mid \sum_j C_j$

Proof: Based on the Hardy- Littlewood inequality.

Total Completion Time : $\sum C_j$

Recall that the SPT rule is optimal for $1 \mid \mid \sum C_j$ which follows from expressing completion time as

$$\sum C_j = np_1 + (n-1)p_2 + \ \ldots + 2p_{n-1} + p_n$$

and using the Hardy-Littlewood inequality.

Interesting Story about Indian Mathematician Srinivas Ramanujan and his work with Hardy and Littlewood: https://en.wikipedia.org/wiki/Srinivasa_Ramanujan

MICHIGAN ENGINEERING
UNIVERSITY of MICHIGAN ■ COLLEGE of ENGINEERING

Example: Use SPT to solve the following instance of $P3 \mid\mid \sum_j C_j$

Jobs	1	2	3	4	5	6	7	8
p_j	4	6	3	2	9	5	4	9

$Pm \mid\mid \sum_j w_j C_j$

What happens when weights are added?

- The WSPT rule minimizes completion time for $1 \mid\mid \sum_j w_j C_j$
- For $\mathrm{Pm} \mid\mid \sum_j w_j C_j$ WSPT is not necessarily optimal but it has a **worst case guarantee**

$$\frac{\sum w_j C_j(WSPT)}{\sum w_j C_j(OPT)} \leq \frac{1}{2}(1+\sqrt{2})$$

Example: Use WSPT to solve the following instance of $P3 \mid\mid \sum_j w_j C_j$

Jobs	1	2	3	4	5	6	7	8
p_j	4	6	3	2	9	5	4	8
w_j	6	5	7	3	2	1	8	5

Project Scheduling

Unconstrained project scheduling can be expressed as the following parallel machine scheduling problem: $\boldsymbol{P\infty \mid prec \mid C_{max}}$

Examples:

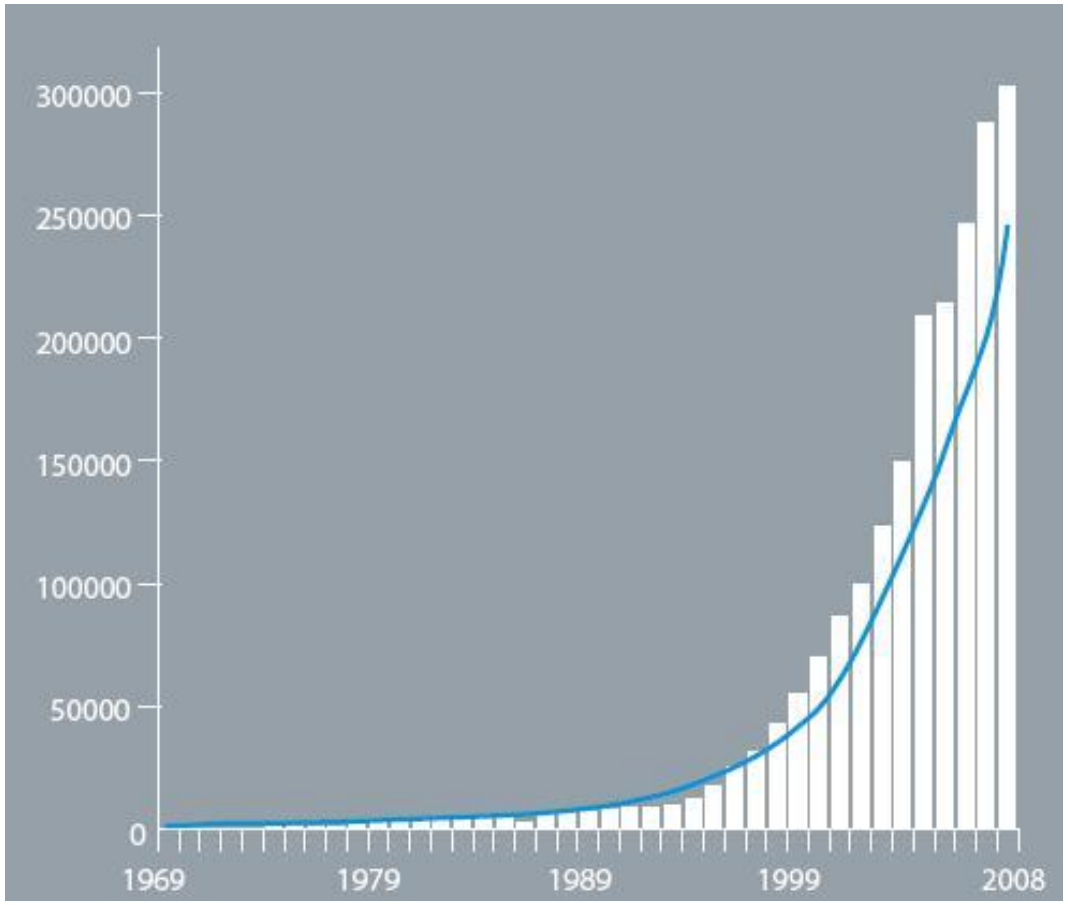
- Construction projects
- Installing new IT systems
- Product design
- Corporate mergers
- Space exploration
- Military operations

Project Managers

This graph shows the increase in Project Management Institute (PMI) membership over time.

Learn more about PMI:

http://www.youtube.com/watch?v=yOkPWKp3OOU



Project Networks

- Project networks are used as an aid in the management of large complex projects
- **Activities** are represented by **arcs** and **nodes** represent completion of a **set** of activities. This is called an **activity on arc** (**AOA**) network
- For each activity there is a set of **predecessors** that must be completed first

Building a Project Network

1. The **start node** represents the start of the project. Arcs from the start node denote activities without predecessors

2. The **finish node** represents completion of the project

3. Nodes representing the completion of an activity have labels that are **larger** than nodes representing the start of an activity

4. No activity is represented by **more than one arc**

5. Two nodes can be connected by **at most one arc**

Note: To avoid violating rules 4 and 5, it can be sometimes necessary to utilize a **dummy activity** that takes zero time

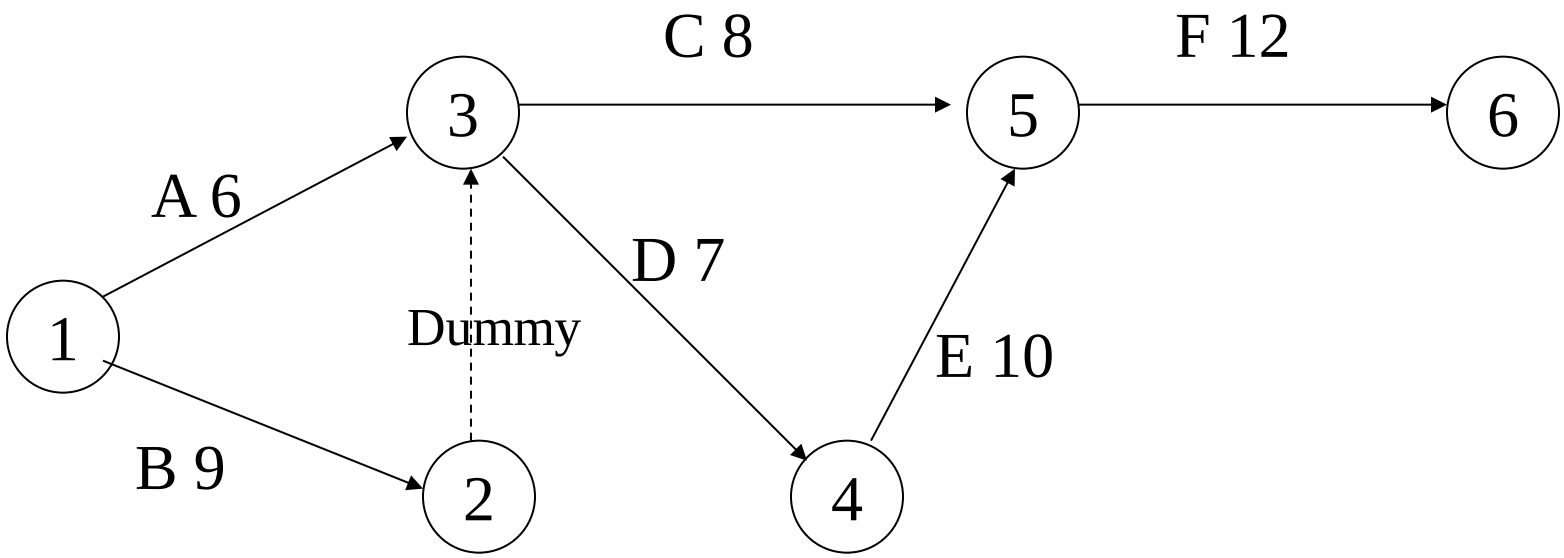
Example: $P\infty \mid prec \mid C_{max}$

A firm plans to launch a new product. A list of activities, their predecessors, and activity durations are below. Draw a project diagram for this project

Activity	Predecessors	Duration(days)
A:train workers	-	6
B:purchase raw materials	-	9
C:produce product 1	A, B	8
D:produce product 2	A, B	7
E:test product 2	D	10
F:assemble products 1&2	C, E	12

Example: $P\infty \mid prec \mid C_{max}$

Project Network



Node 1 = starting node
Node 6 = finish node

Definitions

- The **early event time** for node *i*, represented by *ET*(*i*), is the earliest time at which the node *i* event can occur

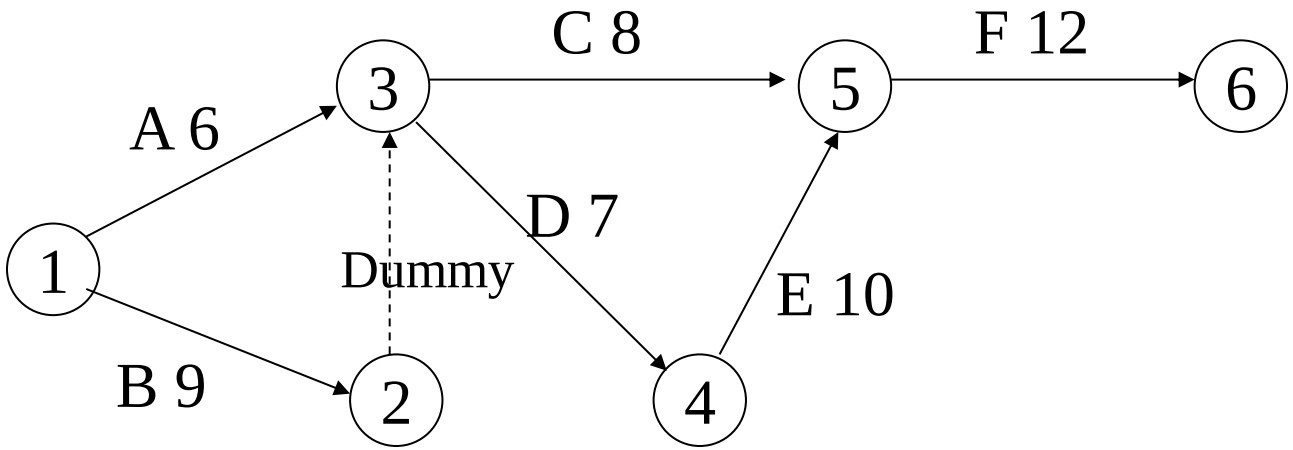
- The **late event time** for node *i*, represented by *LT*(*i*), is the latest time at which the event corresponding to node *i* can occur **without delaying** project completion

Early Event Time

To find the early event time for each node in the project use the following steps:

1. Start with the first node, ET(1) = 0
2. For ET(i)
   - Find each predecessor to node *i*
   - For each predecessor sum the early event time and duration of the activity connecting the immediate predecessor to node *i*
   - *ET*(*i*) equals the maximum of the sums

Example: $P\infty \mid prec \mid C_{max}$



Compute ET(*i*)'s

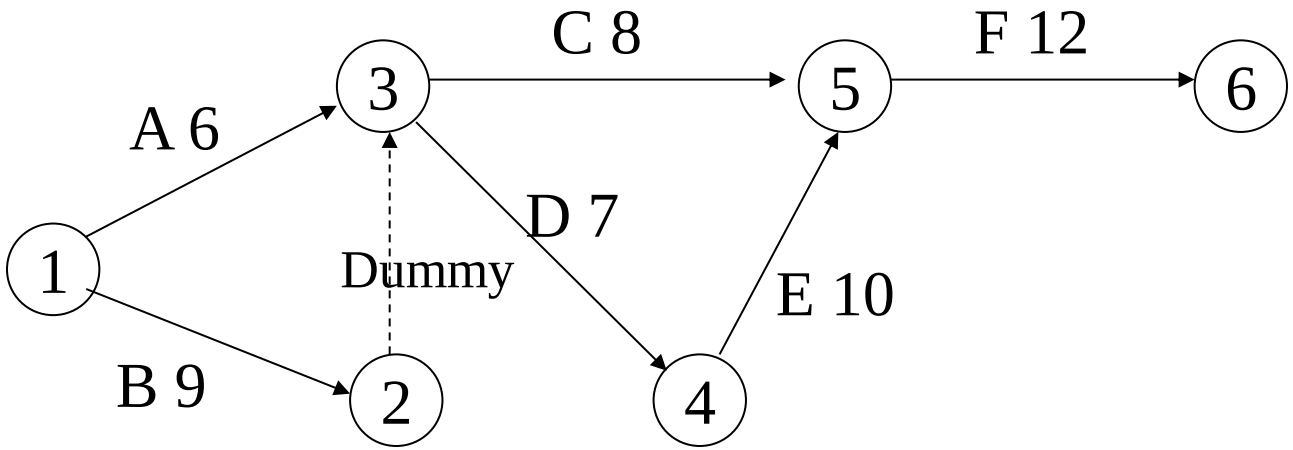
- Start with ET(1) = 0
- ET(2) = 9
- ET(3) = max{ET(2) + 0,ET(1) + 6}=9
- ET(4) = ET(3) + 7 = 16
- ET(5) = max{ET(3) + 8, ET(4) + 10} = 26
- ET(6) = ET(5) + 12 = 38

Late Event Time

To find the late event time for each node in the project use the following steps:

1. Start with the last node, LT(n) = project duration
2. For LT(i)
   - Find each successor to node *i*
   - For each successor find the difference between the late event time and the duration of the activity joining the successor to node i
   - LT(i) is the smallest of the differences computed in the previous step

Example: $P\infty \mid prec \mid C_{max}$

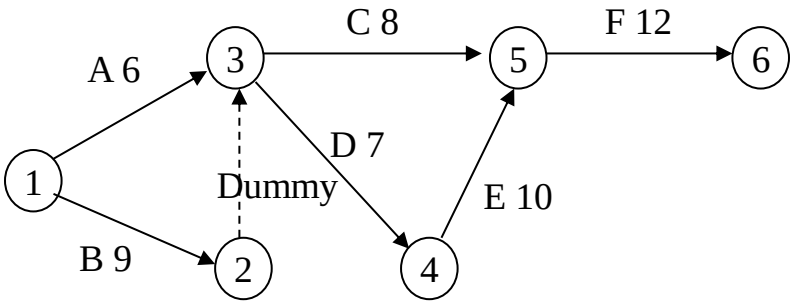


Compute LT(*i*)'s

- Start with LT(6) = ET(6) = 38
- LT(5) = 38-12 = 26
- LT(4) = 26-10 = 16
- LT(3) = min{LT(5) – 8, LT(4) – 7} = min{18, 9} = 9
- LT(2) = LT(3) = 9
- LT(1) = min{LT(3) – 6, LT(2) – 9} = min{3,0} = 0

Example: $P\infty \mid prec \mid C_{max}$

ET(*i*)'s and *LT*(*i*)'s



Node	ET(i)	LT(i)
1	0	0
2	9	9
3	9	9
4	16	16
5	26	26
6	38	38

Total Float

The **total float** of an activity is a measure of how important it is to keep the activity on time

Total float, TF(*i*,*j*), is the amount by which the starting time of activity (*i*,*j*) could be delayed beyond its earliest possible starting time without delaying the project

$$TF(i,j) = LT(j) – ET(i) - t_{ij}$$

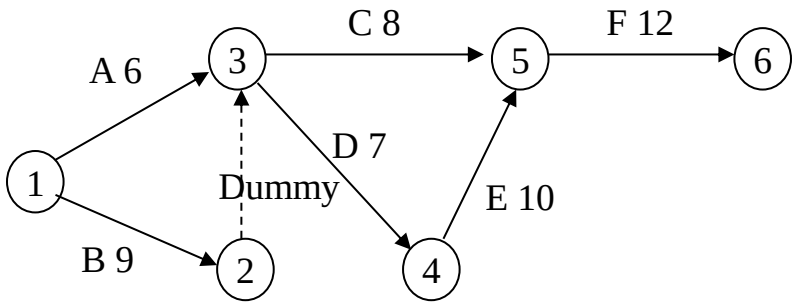
Critical Path

An activity with a total float of zero is a **critical activity**

A path from node 1 to the finish node that consists entirely of critical activities is called a **critical path**

The **critical path** determines the length of the project and delaying a critical activity delays the entire project

Example: $P\infty \mid prec \mid C_{max}$

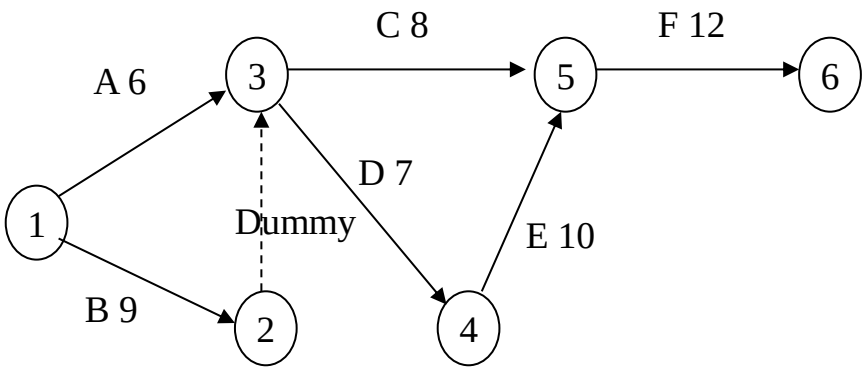


Activity	Total Float	CP
A	9-0-6=3	No
B	9-0-9=0	Yes
C	26-9-8=9	No
D	16-9-7=0	Yes
E	26-16-10=0	Yes
F	38-26-12=0	Yes

The critical path is 1-2-3-4-5-6

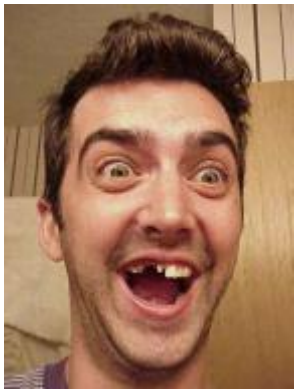
Resource Allocation

The **critical path** is important for making decisions about how to allocate resources



The critical path is 1-2-3-4-5-6

Michigan Grad



Ohio Grad