

# Chapter 3: Single Machine Models

**Dr. Purush Damodaran**  
Dept. Industrial and Systems Engineering

[pdamodaran@niu.edu](mailto:pdamodaran@niu.edu)  
815.753.3172



# Introduction

- Simple and a special case of all other environments
- Insights can be used for other scheduling problems
- Complex problems can be decomposed into
  - Sub problems with single machines
- Consider an assembly line
  - Bottleneck machine should be analyzed to increase throughput

# Completion Time Related Objectives

# Makespan

- $1 \parallel C_{\max}$ : Any sequence is optimum

$$C_{\max} = \max (C_1, C_2, \dots, C_n)$$

$C_{(1)}$  completion time of 1<sup>st</sup> job in the sequence

$C_{(2)}$  completion time of 2<sup>nd</sup> job in the sequence

...

$C_{(n)}$  completion time of nth job in the sequence

$$C_{\max} = C_{(n)}$$

$$C_{(1)} = p_{(1)}$$

$$C_{(2)} = C_{(1)} + p_{(2)} = p_{(1)} + p_{(2)}$$

...

$$C_{(n)} = C_{(n-1)} + p_{(n)} = p_{(1)} + p_{(2)} + \dots + p_{(n)}$$

# Total Completion Time

- The **SPT rule** is optimal for  $1 \parallel \sum_j C_j$
- Shortest Processing Time (SPT) rule
  - Arrange the jobs in non-decreasing order of their processing times
- Proof: By contradiction
  - Suppose schedule S is not SPT, then there should be at least one adjacent pair of jobs (say  $h$  and  $k$ ) such that
    - $p_h > p_k$  but  $h$  is processed just before  $k$

# Total Completion Time

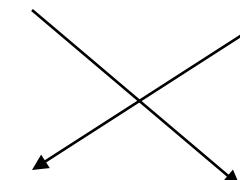
Schedule S



$t$

$$C_k = t + p_h + p_k$$

Does not follow SPT  
 $p_h > p_k$



Schedule S'



$t$

$$C_h = t + p_h + p_k$$

$$\text{Schedule } S: C_h + C_k = (t + p_h) + (t + p_h + p_k) = 2t + 2p_h + p_k$$

$$\text{Schedule } S': C_k + C_h = (t + p_k) + (t + p_k + p_h) = 2t + 2p_k + p_h$$

$S - S' = p_h - p_k > 0 \rightarrow \text{Schedule } S' \text{ is better than } S$

# Total Completion Time Example

j	1	2	3	4	5	6	7	8	9	10
$p_j$	12	2	3	5	1	8	9	5	3	4

SPT

j	5	2	3	9	10	4	8	6	7	1
$p_j$	1	2	3	3	4	5	5	8	9	12

$$\sum_j C_j = 196$$



# Lekin Output

The figure shows the LEKIN - flexible job-shop scheduling system interface. The top menu bar includes Workspace, File, Schedule, Tools, Window, and Help. Below the menu is a toolbar with various icons for file operations and scheduling. The main window contains three primary modules:

- Gantt Chart - Seqs (SPT)**: A horizontal timeline from 1 to 61. Each slot contains a colored bar representing a job's duration. The legend indicates colors for different machines: grey for machine 1, yellow for machine 2, red for machine 3, blue for machine 4, orange for machine 5, green for machine 6, light green for machine 7, magenta for machine 8, and dark blue for machine 9.
- Sequence - Seqs \* (SPT)**: A table showing the sequence of operations for each job. The columns are Mch/Job, Setup, Start, Stop, and Pr.tm. The first row shows a sequence for job J001 across 9 machines, with a total time of 52. Subsequent rows show summary statistics for each machine.
- Job Pool - Jobs \* (Single-operation jobs)**: A table listing individual jobs with their details. The columns are ID, Wght, Rls, Due, Pr.tm., Stat., Bgn, End, T, and wT. The table lists 10 jobs (J001 to J010) with their respective parameters.

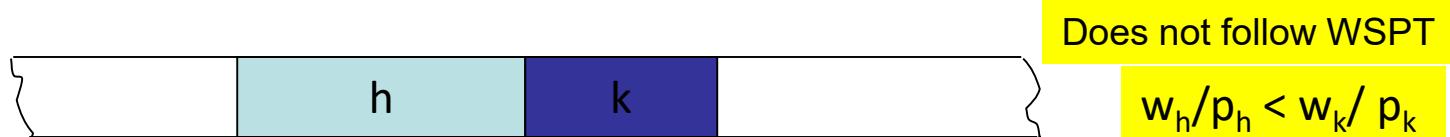


# Total Weighted Completion Time

- The WSPT rule is optimal for  $1 \parallel \sum_j w_j C_j$
- Weighted shortest processing time (WSPT) rule
  - Jobs are ordered in decreasing order of  $w_j/p_j$
- Proof. By contradiction
  - Suppose schedule S is not WSPT, then there should be at least one adjacent pair of jobs (say  $h$  and  $k$ ) such that
    - $w_h/p_h < w_k/p_k$  but  $h$  is processed just before  $k$

# Total Weighted Completion Time

Schedule S



$t$

$$C_k = t + p_h + p_k$$

Schedule S'



$t$

$$C_h = t + p_h + p_k$$

$$\text{Schedule S: } w_h C_h + w_k C_k = w_h(t+p_h) + w_k(t+p_h+p_k)$$

$$\text{Schedule S': } w_k C_k + w_h C_h = w_k(t+p_k) + w_h(t+p_k+p_h)$$

$$w_h/p_h < w_k/p_k \text{ implies } w_h p_k < w_k p_h$$

$$S - S' = w_k p_h - w_h p_k > 0 \rightarrow \text{Schedule S' is better than S}$$

# Total Weighted Completion Time Example

$j$	1	2	3	4	5
$p_j$	12	2	3	5	1
$w_j$	1	3	4	1	2
$w_j/p_j$	1/12	1.5	4/3	1/5	2

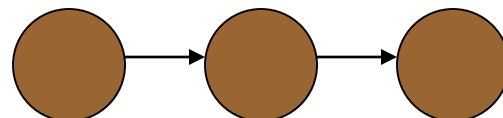
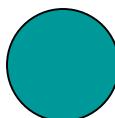
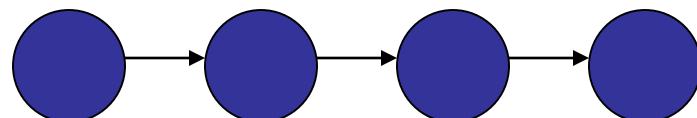
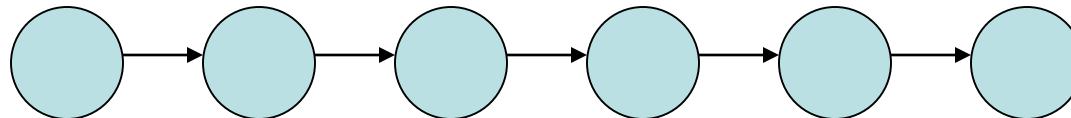
WSPT order       $j_5-j_2-j_3-j_4-j_1$



$$\sum_j w_j C_j = 69$$

$$1 \mid \text{prec} = \text{chains} \mid \sum_j w_j C_j$$

- Parallel chains



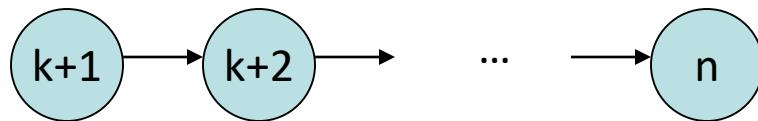
Polynomial time algorithm exists

$$1 \mid \text{chains} \mid \sum_j w_j C_j$$

- Chain I:



- Chain II:



- Assumption:

- once a chain is started, complete it entirely before starting another chain

- Decision: which chain should be processed first?

# 1 | chains | $\sum_j w_j C_j$

$$\text{if } \frac{\sum_{j=1}^k w_j}{\sum_{j=1}^k p_j} > \frac{\sum_{j=k+1}^n w_j}{\sum_{j=k+1}^n p_j}$$

$$\text{if } \frac{\sum_{j=1}^k w_j}{\sum_{j=1}^k p_j} < \frac{\sum_{j=k+1}^n w_j}{\sum_{j=k+1}^n p_j}$$

Process chain of jobs 1, 2, ..., k before  
the chain of jobs k+1, k+2, ...n

Process chain of jobs 1, 2, ..., k after  
the chain of jobs k+1, k+2, ...n

# 1 | chains | $\sum_j w_j C_j$

$$\text{if } \frac{\sum_{j=1}^k w_j}{\sum_{j=1}^k p_j} > \frac{\sum_{j=k+1}^n w_j}{\sum_{j=k+1}^n p_j}$$

Process chain of jobs 1, 2, ..., k before the chain of jobs k+1, k+2, ...n

Proof by Contradiction

Under the sequence 1, 2, ..., k, k+1, k+2, ..., n

$$S_1: \sum_j w_j C_j = w_1 p_1 + w_2(p_1+p_2) + \dots + w_k(p_1+\dots+p_k) + w_{k+1}(p_1+\dots+p_{k+1}) + \dots + w_n(p_1+\dots+p_n)$$

Under the sequence k+1, k+2, ..., n, 1, 2, ..., k

$$S_2: \sum_j w_j C_j = w_{k+1} p_{k+1} + w_{k+2}(p_{k+1}+p_{k+2}) + \dots + w_n(p_{k+1}+\dots+p_n) + \\ w_1(p_1+p_{k+1}+\dots+p_n) + \dots + w_k(p_1+\dots+p_k + p_{k+1} + \dots + p_n)$$

$$S_1 - S_2 : w_{k+1} \sum_{j=1}^k p_j + w_{k+2} \sum_{j=1}^k p_j + \dots + w_n \sum_{j=1}^k p_j - w_1 \sum_{j=k+1}^n p_j - \dots - w_k \sum_{j=k+1}^n p_j$$

# 1 | chains | $\sum_j w_j C_j$

$$S_1 - S_2 : w_{k+1} \sum_{j=1}^k p_j + w_{k+2} \sum_{j=1}^k p_j + \dots + w_n \sum_{j=1}^k p_j - w_1 \sum_{j=k+1}^n p_j - \dots - w_k \sum_{j=k+1}^n p_j$$

$$S_1 - S_2 : \sum_{j=k+1}^n w_j \sum_{j=1}^k p_j - \sum_{j=1}^k w_j \sum_{j=k+1}^n p_j$$

$$\sum_{j=1}^k w_j \sum_{j=k+1}^n p_j > \sum_{j=k+1}^n w_j \sum_{j=1}^k p_j$$

$S_1 - S_2 < 0$  --- Schedule  $S_1$  is better than  $S_2$

# 1 | chains | $\sum_j w_j C_j$ Example

Assumption: when you begin processing a chain it has to be completed

j	1	2	3	4	5	6	7
w <sub>j</sub>	0	18	12	8	8	17	16
p <sub>j</sub>	3	6	6	5	4	8	9

1 → 2  
3 → 4 → 5  
6 → 7

$$\frac{\sum_{j=1}^k w_j}{\sum_{j=1}^k p_j} \longrightarrow \begin{array}{l} \text{Chain 1} \\ 18/9 = 2 \end{array} \quad \begin{array}{l} \text{Chain 2} \\ 28/15 = 1.87 \end{array} \quad \begin{array}{l} \text{Chain 3} \\ 33/17 = 1.94 \end{array}$$

Chain 1 → Chain 3 → Chain 2

j	1	2	6	7	3	4	5
p <sub>j</sub>	3	6	8	9	6	5	4
c <sub>j</sub>	3	9	17	26	30	35	39

# $\rho$ Factor

Consider the chain  $1 \rightarrow 2 \rightarrow \dots \rightarrow k$

$$\rho(1,2,\dots,k) = \frac{\sum_{j=1}^{l^*} w_j}{\sum_{j=1}^{l^*} p_j} = \max_{1 \leq l \leq k} \left( \frac{\sum_{j=1}^l w_j}{\sum_{j=1}^l p_j} \right)$$

$l^*$  is the job that determines the  $\rho$  factor

# $\rho$ Factor

j	1	2	3	4	5	6	7
w <sub>j</sub>	0	18	12	8	8	17	16
p <sub>j</sub>	3	6	6	5	4	8	9

1 → 2  
3 → 4 → 5  
6 → 7

$$\rho(1,2) = \text{job 2} \quad \max\{0/3, (0+18)/(3+6)\} = 2 \text{ for job 2}$$

$$\rho(3,4,5) = \text{job 3} \quad \max\{12/6, (12+8)/(6+5), (12+8+8)/(6+5+4)\} = 2 \text{ for job 3}$$

$$\rho(6,7) = \text{job 6} \quad \max\{17/8, 33/17\} = 2.125 \text{ for job 6}$$

# $\rho$ Factor

- If we relax the assumption “a chain has to be processed in entirety”, then
  - If job  $i^*$  determines  $\rho(1, \dots, k)$ , then there exists an optimal sequence that processes jobs  $1, \dots, i^*$  one after another without any interruption by jobs from other chains.
  - Proof by contradiction. Suppose  $1, \dots, i^*$  is interrupted by a job, say  $v$ , from another chain
    - Optimal sequence will contain the subsequence  $S: 1, \dots, u, v, u+1, \dots, i^*$

# $\rho$ Factor

- Consider
  - Subsequence  $S'$ :  $v, 1, \dots, u, u+1, \dots l^*$
  - Subsequence  $S''$ :  $1, \dots, u, u+1, \dots l^*, v$
- If  $\sum_j w_j C_j$  of  $S$  is less than  $S'$  then
  - $w_v / p_v < (w_1 + \dots + w_u) / (p_1 + \dots + p_u)$
- If  $\sum_j w_j C_j$  of  $S$  is less than  $S''$  then
  - $w_v / p_v > (w_{u+1} + \dots + w_{l^*}) / (p_{u+1} + \dots + p_{l^*})$
- Since  $l^*$  is the  $\rho(1, \dots, k)$ 
  - $(w_{u+1} + \dots + w_{l^*}) / (p_{u+1} + \dots + p_{l^*}) > (w_1 + \dots + w_u) / (p_1 + \dots + p_u)$
- Contradiction

# $\rho$ Factor

- Similarly compare  $S$  with  $S''$  to show the contradiction
- Using similar argument we can show the same result for more than one job from a chain

$$1 \mid \text{chains} \mid \sum_j w_j C_j$$

- Algorithm:
  - whenever the machine is freed, select among the remaining chains the one with the highest  $\rho$  factor. Process this chain without interruption up to and including the job that determines its  $\rho$  factor

# 1 | chains | $\sum_j w_j C_j$ (Example 1)

1 → 2 → 3 → 4

5 → 6 → 7

j	1	2	3	4	5	6	7
w <sub>j</sub>	6	18	12	8	8	17	18
p <sub>j</sub>	3	6	6	5	4	8	10

$$\rho(1,2,3,4) = \max(6/3, 24/9, 36/15, 44/19) = 2.667 \text{ (job 2)}$$

$$\rho(5,6,7) = \max(8/4, 25/12, 43/22) = 2.08 \text{ (job 6)}$$

$\rho(1,2,3,4) > \rho(5,6,7) \rightarrow$  jobs 1 and 2 are scheduled first. **Partial schedule is 1,2.**

$$\rho(3,4) = \max(12/6, 20/11) = 2 \text{ (job 3)}$$

$\rho(5,6,7) > \rho(3,4) \rightarrow$  jobs 5 and 6 are scheduled next. **Partial schedule is 1,2,5,6.**

$\rho(7) < \rho(3,4) \rightarrow$  jobs 3 is scheduled next. **Partial schedule is 1,2,5,6,3.**

Do the next steps to find the solution.

Answer: **1,2,5,6,3,7,4**

# 1 | chains | $\sum_j w_j C_j$ (Example 2)

j	1	2	3	4	5	6	7
w <sub>j</sub>	0	18	12	8	8	17	16
p <sub>j</sub>	3	6	6	5	4	8	9

$1 \rightarrow 2$   
 $3 \rightarrow 4 \rightarrow 5$   
 $6 \rightarrow 7$

Solve this problem in class.

$$1 \mid \text{prec} \mid \sum_j w_j C_j$$

- Polynomial time algorithms exist for chains
- Arbitrary precedence – NP hard

$$1 \mid r_j, p \text{rmp} \mid \sum_j w_j C_j$$

- Preemptive version of WSPT
  - At any point in time, the available job with the highest  $w/(remaining\ p)$  ratio is selected for processing
    - If a job was started its  $w/(remaining\ p)$  ratio will increase
    - Thus this job cannot be preempted by the jobs which were available when the job first began processing
    - However, they can be preempted by newly arriving jobs
  - Does not guarantee an optimal solution
  - The problem is NP-hard

$$1 \mid r_j, p_r m_p \mid \sum_j w_j C_j$$

$j$	1	2	3	4	5	6	7
$w_j$	6	18	12	8	8	17	18
$p_j$	3	6	6	5	4	8	10
$r_j$	0	1	1	3	3	0	0

Solve this problem in class

$$1 \ |r_j, p r m p| \sum_j C_j$$

- When  $w_j = w$  for all  $j$ 
  - Easy
  - What is the heuristic?
- $1 \ |r_j| \sum_j C_j$ 
  - NP-hard

# Completion Time Objectives

- $1 \parallel C_{\max}$  Any sequence
  - $1 \parallel \sum_j C_j$  SPT
  - $1 \mid r_j, p_{rmp} \mid \sum_j C_j$  preemptive SPT
  - $1 \mid r_j \mid \sum_j C_j$  NP-hard
- 
- $1 \parallel \sum_j w_j C_j$  WSPT
  - $1 \mid \text{chains} \mid \sum_j w_j C_j$  use  $\rho$  factor and WSPT
  - $1 \mid \text{prec} \mid \sum_j w_j C_j$  NP-hard

# Single Machine Problem with Completion Time Objective

