Model 4

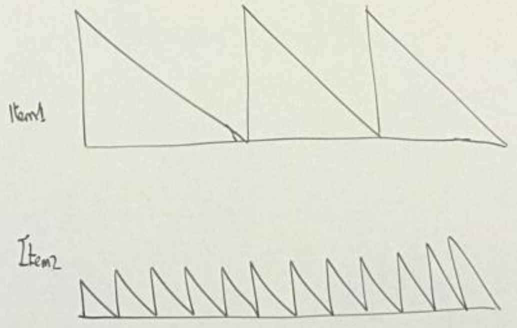
n distinct types of item with different demand rate $\lambda_1, \lambda_2, \ldots, \lambda_n$.

Each item is as in Model 1

A per order – regardless of what is ordered

Let T_j be ordering interval for item j.

We can argue that to minimise cost, $T_1 = T_2 = \cdots = T_n$



It is better to reduce T_1 to T_2

1) inventory cost goes down

2) Order cost goes down. We eliminate extra order costs

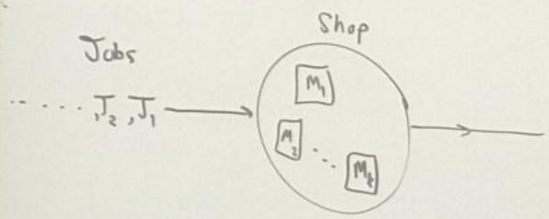
$$Q_j = \lambda_j T$$

T is same interval for all items

$$\text{Total Cost} = \underbrace{\frac{A}{T}}_{\text{Order cost}} + \underbrace{I \sum_{j=1}^{n} \frac{Q_j}{2}}_{\text{Inventory Cost}} = \frac{A}{T} + \frac{IT}{2} \underbrace{\sum_{j} \lambda_j}_{\lambda}$$

— Solve like Model 1

Job Shop Scheduling



Example 1

$$1 \mid \quad \mid \sum_{j=1}^{n} \omega_j C_j$$

number of machines

restrictions etc.

objective

n jobs

Processing time of job j is p_j

Priority weight of job j is ω_j

C_j = completion time of job j.

$$\leftarrow \boxed{M_1} \leftarrow (i_1) \leftarrow (i_2) \cdots\cdots \leftarrow (i_n)$$

Does the order matter?

$p_1 = 2$, $p_2 = 1$

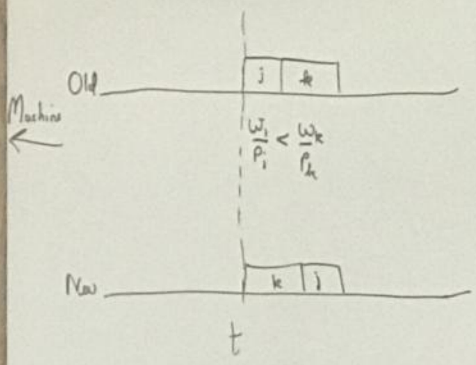
$\square$ ① ② $\qquad \underset{C_1}{2} + \underset{C_2}{3}$

$\square$ ② ① $\qquad \underset{C_1}{1} + \underset{C_2}{3}$

Solution: Order so that $\frac{\omega_1}{p_1} \geq \frac{\omega_2}{p_2} \geq \dots \geq \frac{\omega_n}{p_n}$

Proof

Suppose we don't follow this order.



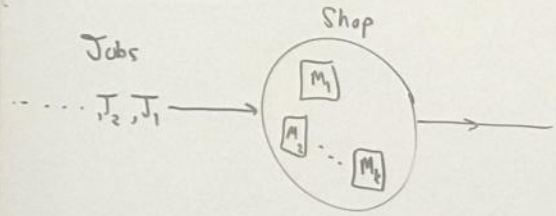
① All jobs other than j, k have the same completion time in Old & New.

$$\underbrace{\text{New} - \text{Old}}_{\text{objective value}} = \omega_k(t+p_k) + \omega_j(t+p_k+p_j) - \omega_j(t+p_j) + \omega_k(t+p_k+p_j)$$

$$= \omega_j p_k - \omega_k p_j$$

$$< 0.$$

Job Shop Scheduling



Example 2

$1 | \dots | L_{max}$

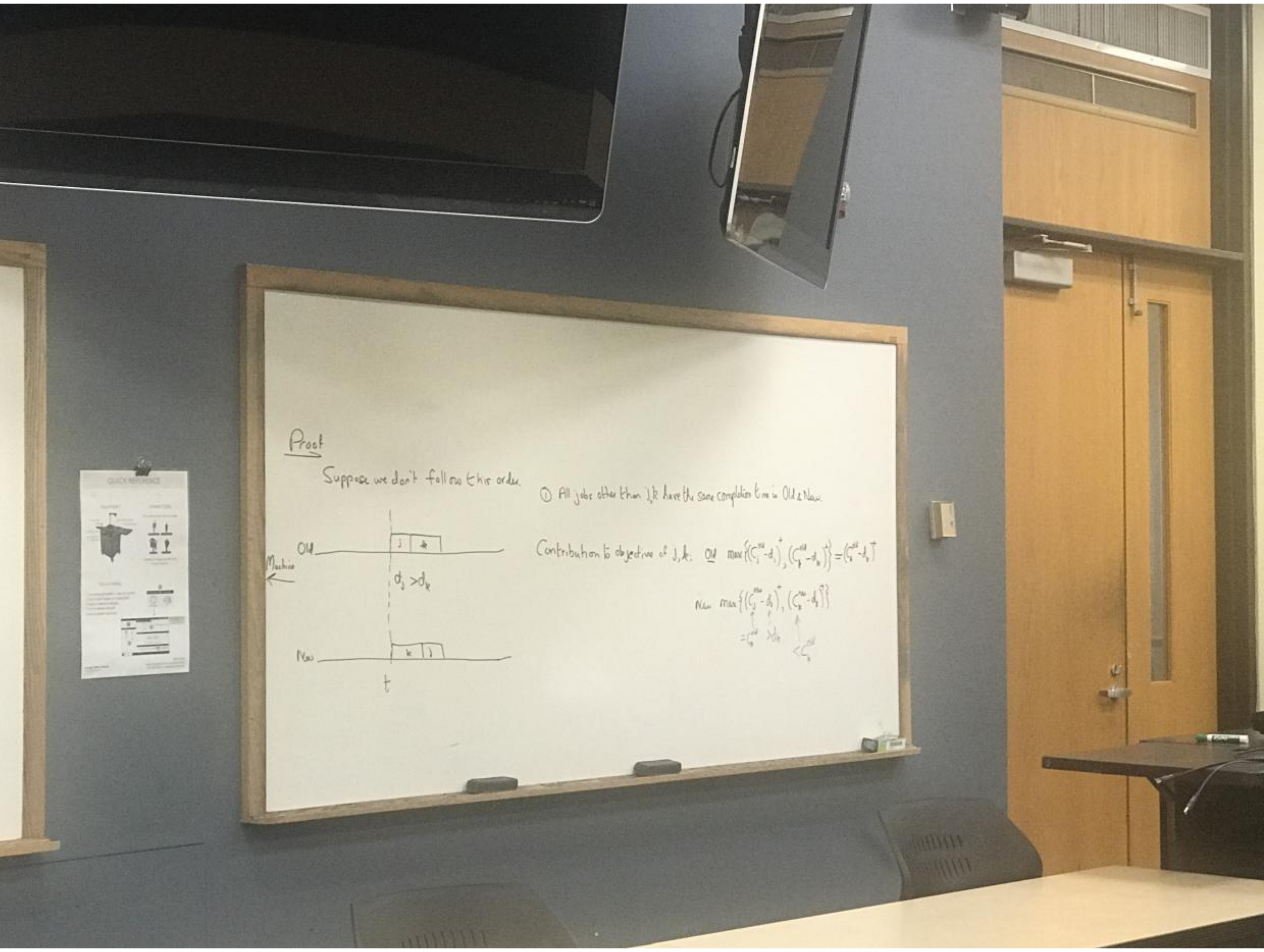
Job j has a due date d_j

$L_j = (C_j - d_j)^+$

$x^+ = \max\{x, 0\}$

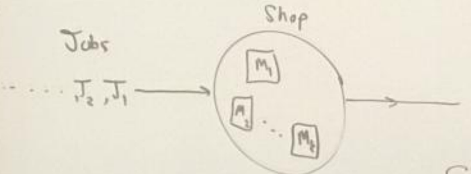
$L_{max} = \max_j L_j$

Sort so that $d_1 \le d_2 \le \dots \le d_n$



Proof
Suppose we don't follow this order.
Old
Machine
$d_j > d_k$
New
t
① All jobs other than j, k have the same completion time in Old & New.
Contribution to objective of j, k: Old $\max\{(C_j^{old} - d_j)^+, (C_k^{old} - d_k)^+\} = (C_k^{old} - d_k)^+$
New $\max\{(C_j^{new} - d_j)^+, (C_k^{new} - d_k)^+\}$
$= C_k^{old}$ $> d_k$ $< C_k^{old}$

Job Shop Scheduling



Example 3

$$1 \mid \text{preemption, release dates} \mid \sum_{j=1}^{n} C_j$$

SRPT rule

Process job with the

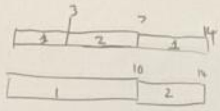
Shortest Remaining Process Time

Job j has a release date r_j – cannot start before r_j

preemption – jobs can be interrupted by newer jobs

Job 1 $r_1 = 0$, $p_1 = 10$

Job 2 $r_2 = 3$, $p_2 = 4$



14 + 7

10 + 14