Scheduling Algorithms

(Part II)

Scheduling Unrelated Job SUM (minimization) problem: a special case with m=1.

- INPUT: *one* machine, n jobs ($i = 1, . . . , n$), $p_i > 0$. (we use p_i since when m=1 clearly there is no difference between unrelated and identical machines)

- OUTPUT: a schedule $\boldsymbol{S}=(S_1)$ *where* $S_1=(S_{1,1},\ldots,S_{1,n})$.

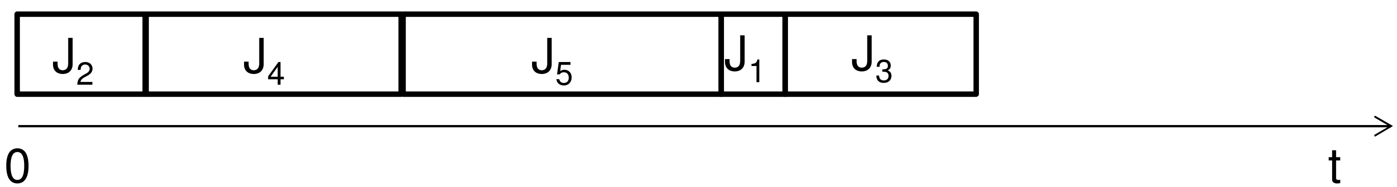
- GOAL*:* Minimizing the Job SUM, that is minimizing $\sum_{i=1}^{n} JC_i(S)$

Does such problem admit a polynomial time algorithm that finds a schedule that minimizes the Job SUM?

Scheduling Unrelated Job SUM (minimization) problem with m=1: an example.

- INPUT: 5 jobs.

	J_1	J_2	J_3	J_4	J_5
p_i=	1	2	3	4	5



$JC_2(S)=2$; $JC_4(S)=2+4=6$; $JC_5(S)=6+5=11$; $JC_1(S)=11+1=12$; $JC_3(S)=12+3=15$.

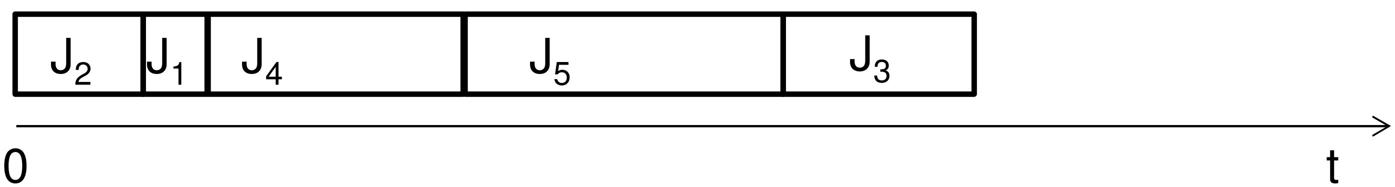
$$\sum_{i=1}^{5} JC_i(S) = 12+2+15+6+11 = 46$$

Is it optimal?

Scheduling Unrelated Job SUM (minimization) problem with m=1: an example (2).

- INPUT: 5 jobs.

	J_1	J_2	J_3	J_4	J_5
$p_i=$	1	2	3	4	5



$JC_2(S)=2$; $JC_1(S)=2+1=3$; $JC_4(S)=3+4=7$; $JC_5(S)=7+5=12$; $JC_3(S)=12+3=15$.

$$\sum_{i=1}^{5} JC_i(S) = 3+2+15+7+12 = 39$$

Is it optimal?

Scheduling Unrelated Job SUM (minimization) problem with m=1: a simple polynomial time algorithm.

- INPUT: *one* machine, *n* jobs ($i = 1, \ldots, n$), $p_i > 0$.
- OUTPUT: a schedule ***S***$=(S_1)$ *where* $S_1 = (S_{1,1}, \ldots, S_{1,n})$.
- GOAL: Minimizing the Job SUM, that is minimizing $\sum_{i=1}^{n} JC_i(S)$

ALGORITHM 1-JSUM:

1. Arrange jobs in non-decreasing order of p_i, and assign jobs to the unique machine by following such arrangement.
2. Return the obtained schedule *S*.

COMPLEXITY: O(n*log(n)) (A simple sort can be done in O(n*log(n)) time).

Theorem 2: *Algorithm 1-JSUM finds an optimal solution for the Scheduling Unrelated Job SUM (minimization) problem with m=1.*

Proof of Theorem 2.

Proof:

- Given that we have just one machine (m=1), in any schedule all the jobs have to be assigned to such a machine.
- Given a schedule *S*, let $p_{(j)}(S)$ be the processing time of the job in the *j*th position in the schedule *S* (i.e., $p_{(j)}(S)$ is the processing time of the job $S_{1,j}$). By the definition of job completion time, it follows that:

$$\sum_{i=1}^{n} JC_i(S) = n * p_{(1)}(S) + (n-1) * p_{(2)}(S) + \ldots + 2 * p_{(n-1)}(S) + p_{(n)}(S)$$

- Clearly the above sum in minimized when $p_{(1)}(S)$ is the smallest value among all the processing times, $p_{(2)}(S)$ is the second smallest value and so on. The Algorithm JSUM uses such order to assign jobs to the machine.

Proof of Theorem 2.

Proof:

- In fact, by contradiction, suppose that a schedule *S'*, that does not assign in non-decreasing order of p_i, is optimal.
- In the schedule *S'* (different than S) there exists a job *i* which is processed before job *j* and $p_i > p_j$. Then, by exachanging job *i* and *j* we get another scheduling with strictly smaller Job SUM.

- Thus, the schedule returned by Algorithm 1-JSUM is optimal.

□

Scheduling Unrelated Job SUM (minimization) problem with weights and m=1.

- INPUT: *one* machine, *n* jobs ($i = 1, \ldots, n$), $p_i > 0$, $w_i > 0$.
- OUTPUT: a schedule $\boldsymbol{S}=(S_1)$ *where* $S_1=(S_{1,1},\ldots,S_{1,n})$.
- GOAL: Minimizing the Job (weighted) SUM, that is minimizing

$$\sum_{i=1}^{n} w_i * JC_i(S)$$

Does such problem admit a polynomial time algorithm that finds a schedule that minimizes the Job (weighted) SUM?

Scheduling Unrelated Job SUM (minimization) problem with weights and m=1: an example.

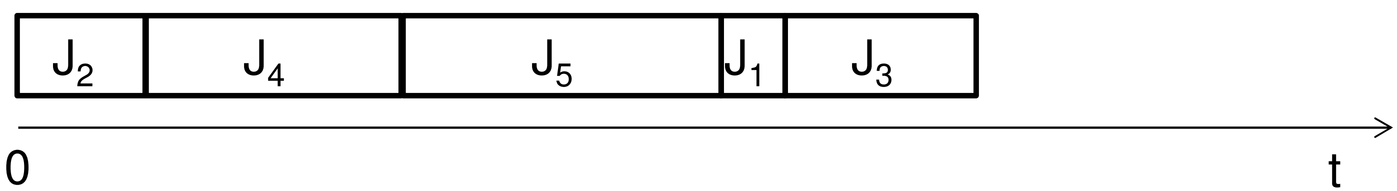
- INPUT: 5 jobs.

p_i=

J_1	J_2	J_3	J_4	J_5
1	2	3	4	5

w_i=

J_1	J_2	J_3	J_4	J_5
1	3	6	2	2



$JC_2(S)=2$; $JC_4(S)=2+4=6$; $JC_5(S)=6+5=11$; $JC_1(S)=11+1=12$; $JC_3(S)=12+3=15$.

$$\sum_{i=1}^{5} w_i * JC_i(S) = 1*12+3*2+6*15+2*6+2*11=142$$

Is it optimal?

Scheduling Unrelated Job SUM (minimization) problem with weights and m=1: an example (2).

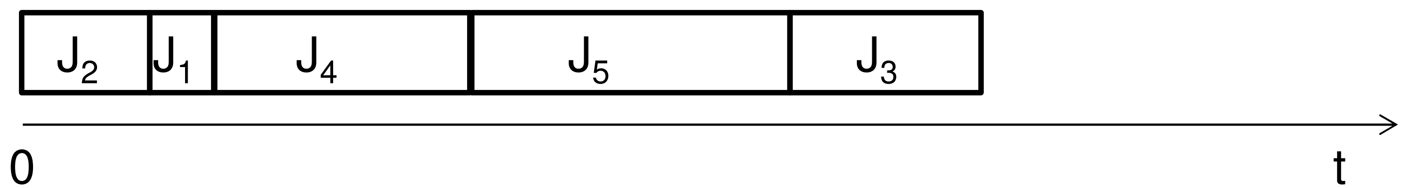
- INPUT: 5 jobs.

p_i=

J_1	J_2	J_3	J_4	J_5
1	2	3	4	5

w_i=

J_1	J_2	J_3	J_4	J_5
1	3	6	2	2



$JC_2(S)=2$; $JC_1(S)=2+1=3$; $JC_4(S)=3+4=7$; $JC_5(S)=7+5=12$; $JC_3(S)=12+3=15$.

$$\sum_{i=1}^{5} w_i * JC_i(S) = 1*3+3*2+6*15+2*7+2*12 = 137$$

Is it optimal?

Scheduling Unrelated Job SUM (minimization) problem with weights and m=1: an example (3).

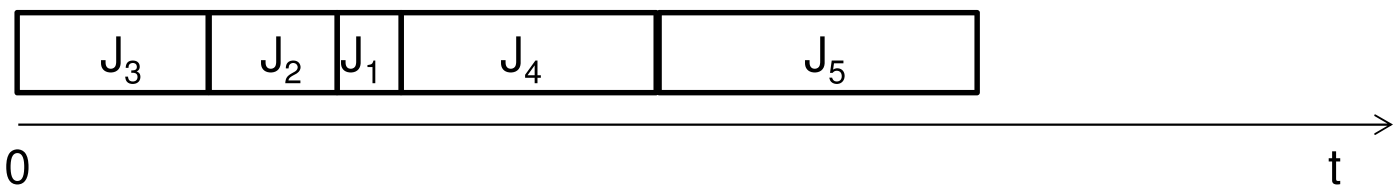
- INPUT: 5 jobs.

p_i=

J_1	J_2	J_3	J_4	J_5
1	2	3	4	5

w_i=

J_1	J_2	J_3	J_4	J_5
1	3	6	2	2



$JC_3(S)=3$; $JC_2(S)=3+2=5$; $JC_1(S)=5+1=6$; $JC_4(S)=6+4=10$; $JC_5(S)=10+5=15$.

$$\sum_{i=1}^{5} w_i * JC_i(S) = 1*6+3*5+6*3+2*10+2*15=89$$

Is it optimal?

Scheduling Unrelated Job SUM (minimization) problem with weights and m=1: a simple polynomial time algorithm.

- INPUT: *one* machine, *n* jobs (*i = 1, . . . , n*), $p_i > 0$, $w_i > 0$.
- OUTPUT: a schedule ***S**=(S* $S=(S_1)$ *where* $S_1=(S_{1,1},\ldots,S_{1,n})$.
- GOAL*:* Minimizing the Job (weighted) SUM, that is minimizing

$$\sum_{i=1}^{n} w_i * JC_i(S)$$

ALGORITHM (weighted)1-JSUM:

1. Arrange jobs in non-increasing order of w_i/p_i, and assign jobs to the machine by following such arrangement.
2. Return the obtained schedule *S*.

COMPLEXITY: O(n*log(n))

Theorem 3: *Algorithm (weighted)1-JSUM finds an optimal solution for the Scheduling Unrelated Job SUM (minimization) problem with weights and m=1.*

Proof of Theorem 3.

Proof:

- By contradiction. Suppose a schedule *S'*, that does not assign jobs in non-increasing order of w_i/pi, is optimal.
- In this schedule there must be at least two adjacent (or consecutive) jobs, say job i_1 followed by job i_2, such that:

$$\frac{w_{i_1}}{p_{i_1}} < \frac{w_{i_2}}{p_{i_2}}$$

- Assume job i_1 starts its processing at some time t. Now suppose to exchange jobs i_1 and i_2 and let us call such a new schedule S'' (i.e. in S'' i_2 *is* scheduled before i_1). All other jobs remain in their original position of *S'*.

Proof of Theorem 3. (2)

Proof:

- The total weighted job completion times of the jobs processed before jobs i_1 and i_2 is not affected by the interchange. Neither is the total weighted job completion times of the jobs processed after jobs i_1 and i_2.
- Thus the difference in the values of the objectives under schedules *S'* and S'' is due only to jobs i_1 and i_2.
- Under the schedule S', the total weighted job completion times of jobs i_1 and i_2 is:

$$(t + p_{i_1}) * w_{i_1} + (t + p_{i_1} + p_{i_2}) * w_{i_2}$$

- While under the schedule S'', the total weighted job completion times of jobs i_1 and i_2 is:

$$(t + p_{i_2}) * w_{i_2} + (t + p_{i_2} + p_{i_1}) * w_{i_1}$$

Proof of Theorem 3. (3)

Proof:

- It can be verified that if $w_{i_1} / p_{i_1} < w_{i_2} / p_{i_2}$ (that is our assumption in this proof!), then the sum of the two weighted job completion times under schedule *S''* is strictly less than under *S'*. In fact:

$$s' - s'' = (t + p_{i_1}) * w_{i_1} + (t + p_{i_1} + p_{i_2}) * w_{i_2} - [(t + p_{i_2}) * w_{i_2} + (t + p_{i_2} + p_{i_1}) * w_{i_1}]$$

$$\Downarrow$$

$$s' - s'' = tw_{i_1} + p_{i_1} w_{i_1} + tw_{i_2} + p_{i_1} w_{i_2} + p_{i_2} w_{i_2} - [tw_{i_2} + p_{i_2} w_{i_2} + tw_{i_1} + p_{i_2} w_{i_1} + p_{i_1} w_{i_1}]$$

$$\Downarrow$$

$$s' - s'' = p_{i_1} w_{i_2} - p_{i_2} w_{i_1}$$

- We want to prove that S' - S''>0

Proof of Theorem 3. (4)

Our assumption

$$\frac{w_{i_2}}{p_{i_2}} > \frac{w_{i_1}}{p_{i_1}}$$

Proof:

$$s' - s'' = p_{i_1} w_{i_2} - p_{i_2} w_{i_1} \overset{?}{>} 0$$

$$p_{i_1} w_{i_2} > p_{i_2} w_{i_1} \xrightarrow{\text{Diving by } p_{i_2}} \frac{p_{i_1} w_{i_2}}{p_{i_2}} > w_{i_1} \xrightarrow{\text{Diving by } p_{i_1}} \frac{w_{i_2}}{p_{i_2}} > \frac{w_{i_1}}{p_{i_1}}$$

- This contradicts the optimality of *S'* and completes the proof of the theorem.

□

Scheduling Identical Job SUM (minimization) problem.

- INPUT: *m identical* machine ($h = 1, ..., m$), n jobs ($i = 1, ..., n$), $p_i > 0$.

- OUTPUT: a schedule $\boldsymbol{S}=(S_{1,...,}S_m)$ where $S_h=(S_{h,1},...,S_{h,k_h})$.

- GOAL: Minimizing the Job SUM, that is minimizing

$$\sum_{i=1}^{n} JC_i(S)$$

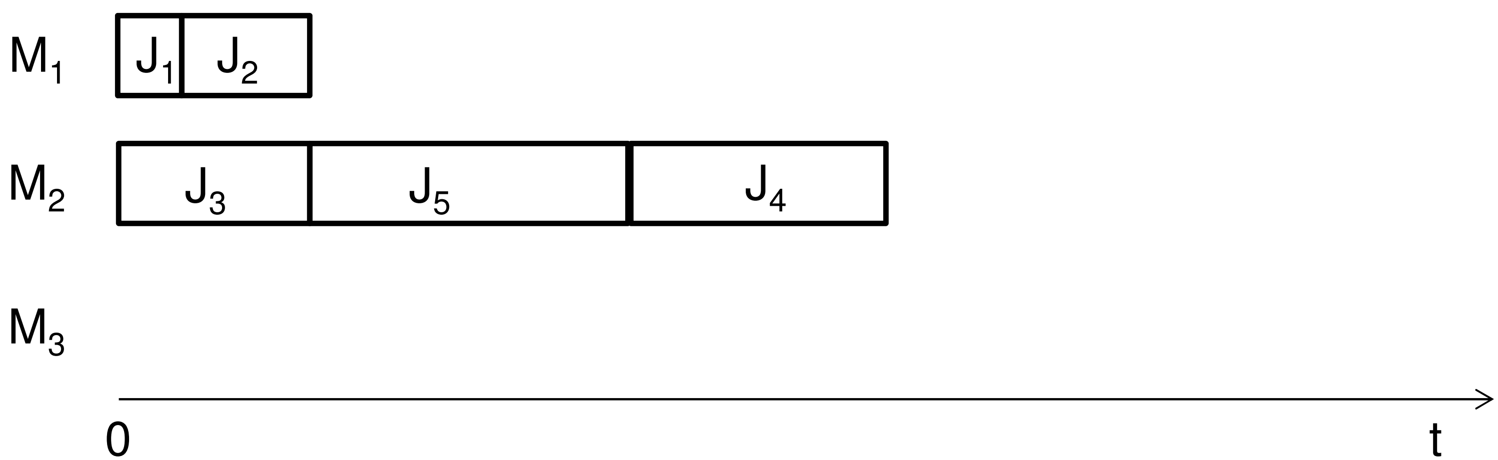
Does such problem admit a polynomial time algorithm that finds a schedule that minimizes the Job SUM?

Scheduling Identical Job SUM (minimization) problem: an example.

- INPUT: 5 jobs, 3 machines.

p_i=

J_1	J_2	J_3	J_4	J_5
1	2	3	4	5



$JC_1(S)=1; JC_2(S)=1+2=3; JC_3(S)=3; JC_5(S)=3+5=8; JC_4(S)=8+4=12.$

$$\sum_{i=1}^{5} JC_i(S) = 1+3+3+12+8 = 27$$

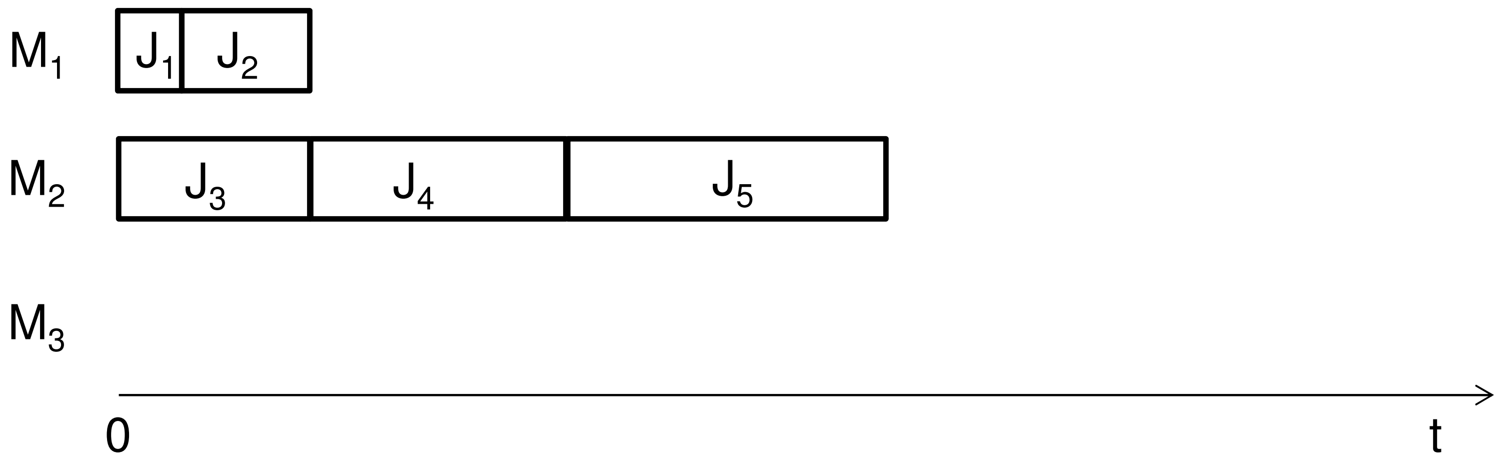
Is it optimal?

Of course not!
(By Theorem 2)

Scheduling Identical Job SUM (minimization) problem: an example. (2)

- INPUT: 5 jobs, 3 machines.

$p_i=$	J_1	J_2	J_3	J_4	J_5
	1	2	3	4	5



$JC_1(S)=1$; $JC_2(S)=1+2=3$; $JC_3(S)=3$; $JC_4(S)=3+4=7$; $JC_5(S)=7+5=12$.

$$\sum_{i=1}^{5} JC_i(S) = 1+3+3+7+12 = 26$$

Is it optimal?