Envy-free Scheduling with unrelated Machines

Resources

- As far as the exam is concerned, it is enough studying (and then deeply understanding) the following slides.

- However, if interested, you can find results contained in these slides and much more at this paper:

V. Bilò, A. Fanelli, M. Flammini, G. Monaco, L. Moscardelli: The Price of Envy-Freeness in Machine Scheduling. *Theoretical Computer Science.* Vol. 613, pp. 65—78, 2016.

Introduction

- We consider the Envy-free Scheduling Unrelated Machine MAKESPAN (minimization) problem.
- We still suppose that in any schedule ***S*** all the machines get at least one job, (we can always remove machines getting no job).

k-Envy-free Scheduling Unrelated Machine MAKESPAN (minimization) problem.

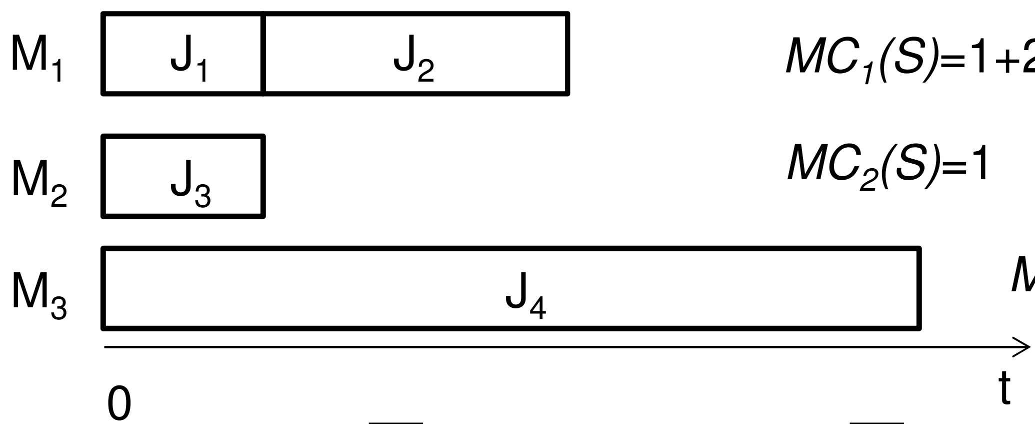
- INPUT: *m unrelated* machine ($h = 1, \ldots, m$), n jobs ($i = 1, \ldots, n$), $p_{ih} > 0$, parameter (positive integer) k≥1.
- OUTPUT: *k-envy free schedule* **S**=($S_{1,\ldots,}S_m$) *such that* $\sum_{i \in S_j} p_{ij} \le k \sum_{i \in S_{j'}} p_{ij}$ *for any machines j, j'.*
- GOAL*:* Minimizing the Machine MAKESPAN $Max_{h=1,\ldots,m}\{MC_h(S)\}$
- In other words, a schedule *S* is *k*-envy-free if, for any machine *j*, the completion time of machine *j* in *S* is at most *k* times than the completion time machine *j* could achieve by getting the jobs of any another machine *j'* (that gets at least one job in *S*), for a given factor *k*≥1.
- If for some machine *j*, there exists a machine *j'* such that $\sum_{i \in S_j} p_{ij} > k \sum_{i \in S_{j'}} p_{ij}$ we will say that machine *j* *envies* machine *j'* in the schedule *S*.

An example

- INPUT: *3* Unrelated machines, 4 jobs.

p_{ih}=

M\J	J_1	J_2	J_3	J_4
M_1	1	2	3	4
M_2	3	5	1	7
M_3	2	2	4	5



$MC_1(S)$=1+2=3

$MC_2(S)$=1

$MC_3(S)$=5

Is it a 1-envy free ?

$$\sum_{i\in S_1} p_{i1} = 1+2 = 3 \le \sum_{i\in S_2} p_{i1} = 3$$ M_1 does not envy M_2

$$\sum_{i\in S_1} p_{i1} = 1+2 = 3 \le \sum_{i\in S_3} p_{i1} = 4$$ M_1 does not envy M_3

It is easy to see that M_2 does not envy both M_1 and M_3

$$\sum_{i\in S_3} p_{i3} = 5 > \sum_{i\in S_1} p_{i3} = 2+2 = 4$$ M_3 envies M_1

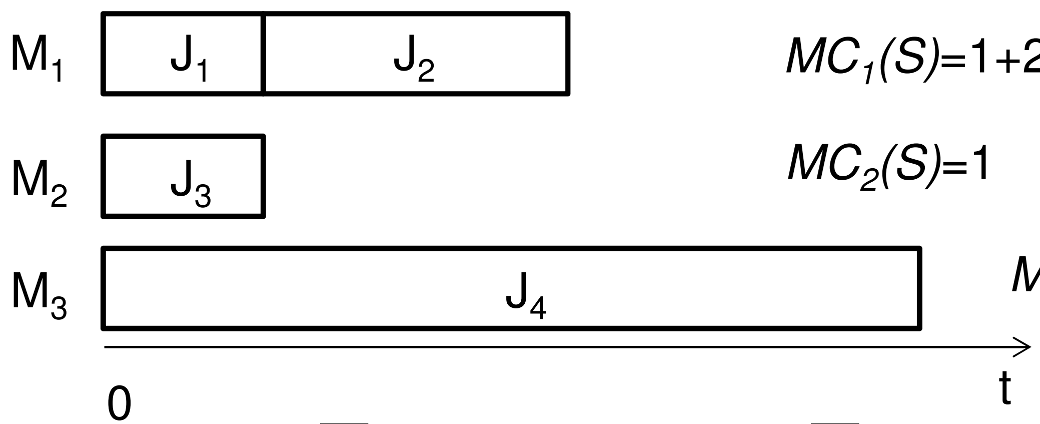
It is <u>not</u> 1-envy-free

An example

- INPUT: *3* Unrelated machines, 4 jobs.

p_{ih}=

M\J	J_1	J_2	J_3	J_4
M_1	1	2	3	4
M_2	3	5	1	7
M_3	2	2	4	5



$MC_1(S)$=1+2=3

$MC_2(S)$=1

$MC_3(S)$=5

Is it a 2-envy free ? $\sum_{i\in S_1} p_{i1} = 1+2=3 \leq 2*\sum_{i\in S_2} p_{i1} = 2*3=6$ M_1 does not envy M_2

$\sum_{i\in S_1} p_{i1} = 1+2=3 \leq 2*\sum_{i\in S_3} p_{i1} = 2*4=8$ M_1 does not envy M_3

It is easy to see that M_2 does not envy both M_1 and M_3

$\sum_{i\in S_3} p_{i3} = 5 \leq 2*\sum_{i\in S_1} p_{i3} = 2*4=8$ M_3 does not envy M_1

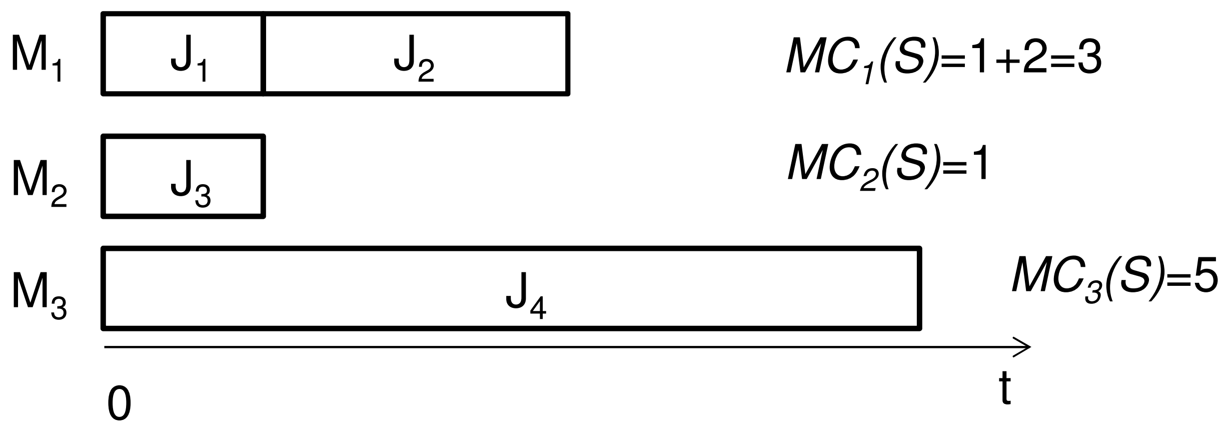
Yes! it is 2-envy-free $\sum_{i\in S_3} p_{i3} = 5 \leq 2*\sum_{i\in S_2} p_{i3} = 2*4=8$ M_3 does not envy M_2

An example

- INPUT: *3* Unrelated machines, 4 jobs.

p_{ih}=

M\J	J_1	J_2	J_3	J_4
M_1	1	2	3	4
M_2	3	5	1	7
M_3	2	2	4	5



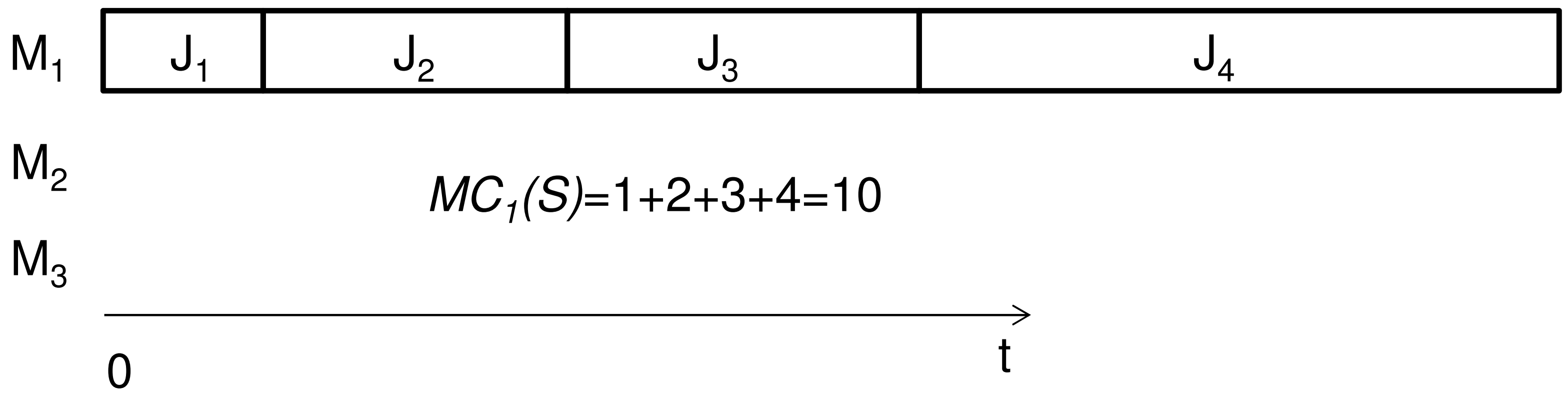
Since such schedule is 2-envy-free, then it is k-envy-free for any k≥2

An example

- INPUT: *3* Unrelated machines, 4 jobs.

p_{ih}=

M\J	J_1	J_2	J_3	J_4
M_1	1	2	3	4
M_2	3	5	1	7
M_3	2	2	4	5



$MC_1(S)$=1+2+3+4=10

An 1-envy-free schedule!

Notice that a 1-envy-free solution (schedule) can be always obtained by assigning all jobs to one machine.
That is, k-envy-free solutions always exist! For any k≥1

Our goal

- We are interested in bounding the *Price of k-envy-freeness* that is the ratio between the MAKESPAN of the best k-envy-free schedule and the MAKESPAN of an optimal scheduling (non necessarily k-envy-free).

- The *Price of k-envy-freeness* is important because measures how much we lose by requiring that schedulings have to be k-Envy-free.

An upper bound to the Price of k-envy freeness ($k \geq 1$)

Theorem 11: The *Price of k-envy-freeness for unrelated machines is at most* $(1+1/k)^{\min\{n,m\}-1}$ *, for any* $k \geq 1$*.*

Proof:

- We are going to show an algorithm that takes in input an optimal schedule *OPT* for the problem without k-envy-free constraint and transforms *OPT* into a k-envy-free scheduling whose MAKESPAN is at most $(1+1/k)^{\min\{n,m\}-1}$ the MAKESPAN of *OPT*.

- Let us consider the algorithm of the next slide:

…An upper bound to the Price of k-envy freeness (k≥1)

The algorithm:

INPUT: Optimal schedule *OPT* for the problem without k-envy-free constraint.

1. *S = OPT;*
2. While *there exists a pair of machines (j,j') such that j envies j' then:*
   - $S_j = S_j \cup S_{j'}$;
   - $S_{j'} = \emptyset$;
3. End while.
4. Return *S.*

- Since the algorithm removes all the envious, the returned schedule S is k-envy-free.

… An upper bound to the Price of k-envy freeness ($k \geq 1$)

- The algorithm iteratively moves the jobs from a machine *j'* to *j* whenever *j* envies *j'*;
- It is obvious that, since the number of non-empty machines in *S* is at most *min{n,m},* the number of iterations is at most *min{n,m} – 1*;
- Moreover notice that when *j* envies *j'*, we have that:

$$\sum_{i \in S_{j'}} p_{ij} < \frac{\sum_{i \in S_j} p_{ij}}{k}$$

- It follows that at each iteration, the current processing time of machine *j*, as a consequence of the movement of the job from *j'* to *j*, increase by a factor of at most (1+1/k);
- It results that the MAKESPAN of the final schedule *S* obtained from the algorithm is at most $(1+1/k)^{\min\{n,m\}-1}$ times the MAKESPAN of the optimal solution *OPT.*

□

A lower bound to the Price of k-envy freeness (k≥1)

Theorem 12: The *Price of k-envy-freeness for unrelated machines is at least* $[1+1/(k+\varepsilon)]^{\min\{n,m\}-1}$ *, for any k≥1 and any (small) ε>0.*

Proof:

- It is possible to show an instance where $\frac{C_{\max}(k-Envy\_OPT)}{C_{\max}(OPT)} \geq (1+\frac{1}{k+\varepsilon})^{\min\{n,m\}-1}$ for any small ε>0.

- (no proof here! Of course I can provide the proof to interested students!)

Final considerations

- An important feature of the proofs we used to upper bound the *Price of k-envy-freeness* in the various cases (for both identical and unrelated machines) is that they rely on polynomial time algorithms constructing k-envy-free assignments of "reasonably" low MAKESPAN.
- In particular in the case of Theorem 8, we find a scheduling without the need of an initial scheduling in input.
- However in the cases of Theorem 9 and Theorem 11, given in input an optimal solution *OPT* that minimizes the MAKESPAN to the problem without the k-envy-free constraint, all the designed algorithms rearrange the allocations defined by *OPT* so as to obtain in polynomial time a k-envy-free assignment *S* for which we were able to show that:

$$\frac{C_{\max}(S)}{C_{\max}(OPT)} \leq r$$

where r is the upper bound that depends on the different cases we have considered. For instance, in Theorem 8, $r = \min\{n,m\}$. In Theorem 9, $r = 1+1/k$. Finally, in Theorem 11, $r = (1+1/k)^{\min\{n,m\}-1}$.

Final considerations (2)

- Unfortunately we are not always able to compute *OPT* in polynomial time!
- However, all the designed algorithms we used in theorems 9 and 11 to upper bound the *Price of k-envy-freeness*, given in input any schedule *N* that could be not k-envy-free and with MAKESPAN equal to α times (for some $\alpha \geq 1$) the optimal MAKESPAN (that is $C_{max}(N) = \alpha * C_{max}(OPT)$), rearrange the allocation defined by *N* so as to obtain in polynomial time a k-envy-free assignment *S* such that:

$$\frac{C_{\max}(S)}{C_{\max}(N)} \leq r$$

where *r* is the upper bound we showed on the different cases we have considered.

- It follows that: $\dfrac{C_{\max}(S)}{C_{\max}(N)} = \dfrac{C_{\max}(S)}{\alpha * C_{\max}(OPT)} \leq r$

- It implies that $$\frac{C_{\max}(S)}{C_{\max}(OPT)} \leq \alpha * r$$

Final considerations (3)

- It means that for the cases where the problem is NP-Hard (recall that the problem of computing the minimum MAKESPAN is NP-Hard for identical machines and therefore for unrelated machines) we have polynomial time algorithms that compute k-envy-free solution with “reasonably” low MAKESPAN.

- Indeed we can use a polynomial time algorithm to compute an approximated scheduling without the k-envy-free constraint, and then we apply our upper bound to the price of k-envy-freeness algorithms (theorems 9 and 11) to return the final k-envy-free solution.

- Recall we have already seen (Theorem 5) the Algorithm *MAKESPAN* that is a (4/3 – 1/3m)-approximation algorithm for the Scheduling Identical Machine MAKESPAN (minimization) problem.
- In the literature (we have not seen them in this course) there exists a PTAS (polynomial time approximation scheme) for the Scheduling Identical Machine MAKESPAN (minimization) problem, and a 2-approximation algorithm for the Scheduling unrelated Machine MAKESPAN (minimization) problem.