About coordination mechanisms for selfish scheduling with multiple tasks

Johanne Cohen, Université de Versailles PRISM-CNRS

Fanny Pascual, Université Paris 6 - LIP6

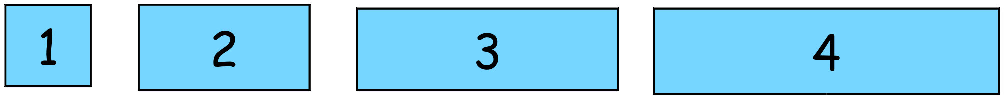
Outline

1. Problem
2. Properties of coordination mechanisms
3. Stability of classical mechanisms
4. Conclusion and future work

A scheduling problem

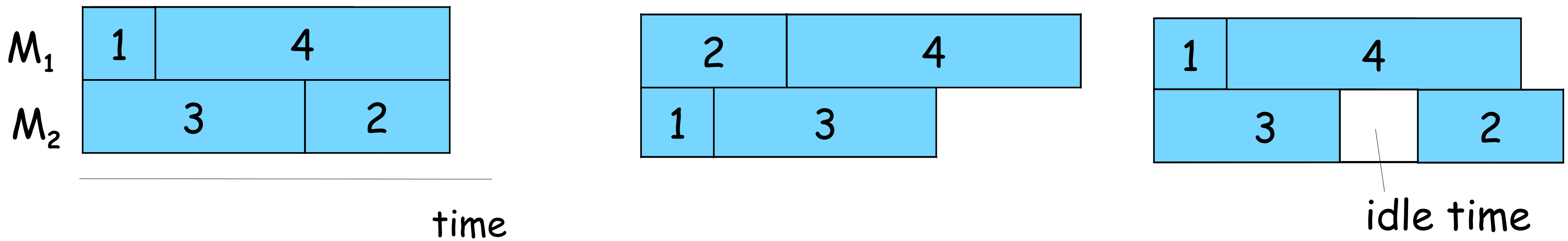
- Data : m machines $M_1, \ldots, M_m$, a set of n tasks.
  - An instance :

    Tasks :



    2 identical parallel machines M_1 and M_2 .

- Possible schedules :



$\sum C_j$ = 1+5+3+5 = 14 $\sum C_j$ = 2+6+1+4 = 13 $\sum C_j$ = 1+5+3+6 = 15

- An objective function
  - Example : Min. average completion time

Context

- Algorithmic game theory (AGT).
  Shared ressources: agents with conflicting interests interact.
  Centralized protocols are not always possible.

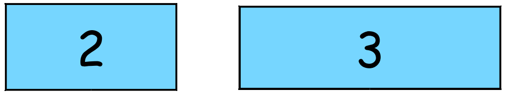
  Machine : processor, printer, link in a network …

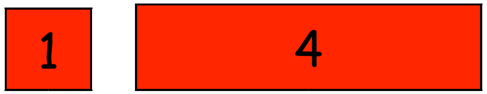
  Scheduling problems have been studied in AGT.

- Each agent has one objective, and a set of possible strategies. We focus on (pure) Nash equilibrium: no agent can improve its objective function by unilateraly changing its strategy.

Example

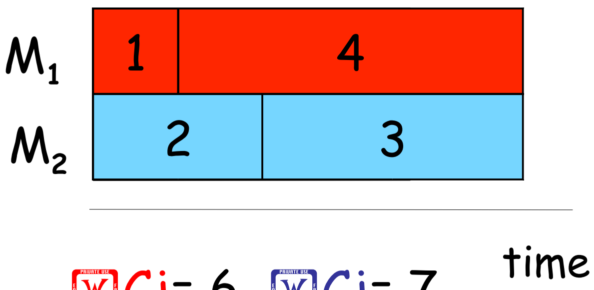
- 2 agents A et B

  A has 2 tasks: 

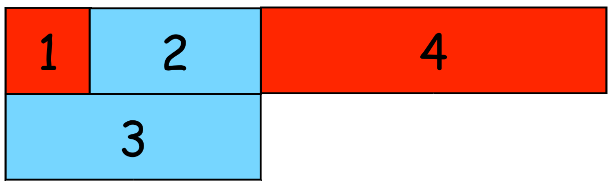
  and B has 2 tasks: 

Strategy : choose on which machine to schedule each task.

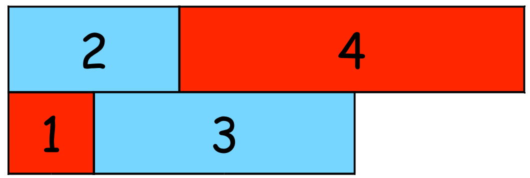
- The machines schedule the tasks by increasing order of lengths.
- Aim of the agents : Min average completion time



$\sum C_j$= 6, $\sum C_j$= 7



$\sum C_j$= 8, $\sum C_j$= 6



$\sum C_j$= 7, $\sum C_j$= 6

Nash equilibrium

Price of anarchy

- Global objective function (social cost)
  - Example : Min. sum of completion times

- Price of anarchy = $\dfrac{\text{Social cost (Worst Nash equilibrium)}}{\text{Social cost (optimal solution)}}$

Measures the loss of efficiency du to the lack of cooperation between the selfish agents.

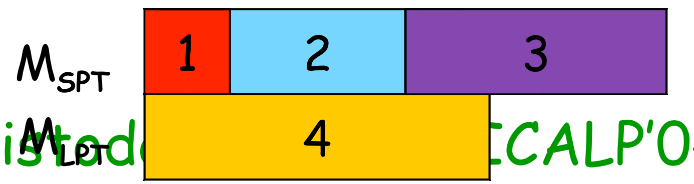
Coordination mechanisms

- Introduced by Christodoulou et al. in Icalp'04.
- Coordination mechanism = set of scheduling policies, one for each machine.

  Each policy :
  - Gives the order of the tasks on the machine, and may introduce idle times.
  - Is local : it depends on the tasks scheduled on the machine only.
  - Does not distinguish the tasks of the different agents. Each task is identified by its length and its identification number.

- Classical policies :
  - SPT (LPT): tasks are scheduled in increasing order of their lengths (resp. in decreasing order of their lengths).
  - Random : tasks are scheduled in a random order.

- Example of a coordination mechanism :



- Christod… CALP'04], introduced the coordination mechanisms when 1 agent = 1 task.
- Immorlica et al. [TCS 09] : study of the convergence and the price of anarchy of the schedules induced by the classical policies.

Our problem

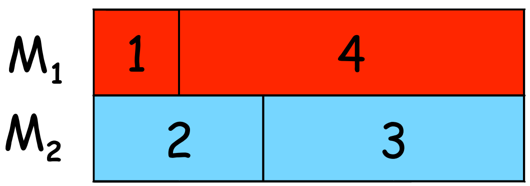
- m machines shared between 2 agents A et B having each one a set of tasks.
- Aim of each agent : Minimize the sum of the completion times of its tasks.

- Does there exist a coordination mechanism which always induce Nash equilibria ?

- What is the stability of the solutions obtained with the classical policies SPT, LPT, and Random ?

Stability of a schedule

⋄-approximate Nash equilibrium : no agent can improve its objective (its sum of completion times) by a ratio larger than ⋄ by changing its strategy (by moving its tasks).



Agent A: Cj= 7; could obtain 6
-> improvement ratio =7/6 .

Agent B : Cj= 6; could obtain 5
-> improvement ratio =6/5 .

=> 6/5-approximate Nash equilibrium.

Outline

1. Problem

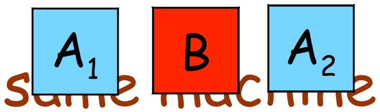
2. Properties of coordination mechanisms
   - Stability
   - Price of anarchy

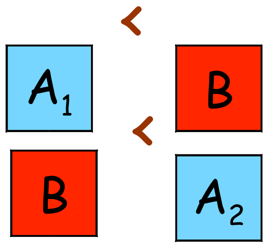
1. Stability of classical mechanisms

2. Conclusion and future work

Stability

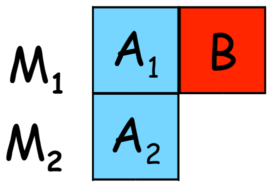
- Property : If all the machines use the same deterministic policy which doesn't use idle times, then there does not always exist a Nash equilibrium.

- Proof: (m=2) Let 3 tasks  s.t.
  - If A_1 and B are alone on a same machine :
  - If B and A_2 are alone on a same machine :

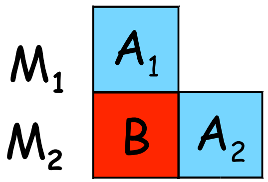


There is no Nash equilibrium.

Agent A wants :



Agent B wants :



Price of anarchy

- Property : If all the machines have the same deterministic policy, then the price of anarchy is at least 2.
- Proof: m=2, 3 tasks of length 1 s.t.

  A_1 < B

  B < A_2

  Idle times :

  - If there are 2 tasks of length 1 on the same machine :

    | i1 | T | i2 | T |

  - If there is … of length 1 alone on a machine :

    | i3 | T |

- We distinguish 4 cases:

  - Case 1:

i1	A_1	i2	B_1
i3	A_2		

□Cj= i1+i2+2
If B_1 goes on M_2 : □Cj= i1+1
=> This is not a Nash equilibrium.

  - Case 2:

i1	B_1	i2	A_2
i3	A_1		

□Cj= i1+i2+i3+3
Exchange A_1 □ A_2 : □Cj= i1+i3+2
=> This is not a Nash equilibrium.

  - Case 3:

i1	A_1	i2	A_2
i3	B_1		

□Cj= 2 * i1+i2+3
Nash equilibrium only if
i1+i3+2 □ 2 * i1+i2+3,
i.e. if i3 □ 1+i1+i2
If i3 □ 1 then price of anarchy □ 2.

  - Case 4: the 3 tasks are together

By contradiction. If the price of anarchy is < 2 □ i3 < 1.
With other properties, we can deduce than there is an instance without Nash equilibrium.

=> With identical deterministic policies, the price of anarchy is □ 2.

Outline

1. Problem

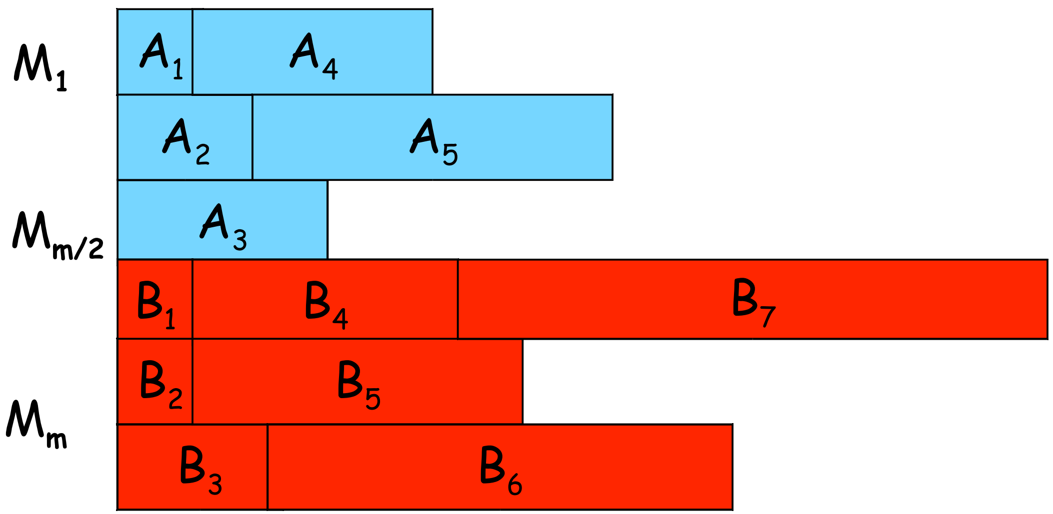
2. Properties of coordination mechanisms

3. Stability of classical mechanisms
   - SPT
   - LPT and Random

1. Conclusion and future work

The SPT policy

- Property : If all the machines use the SPT policy, then there exist always a 3-approximate Nash equilibrium.
- Proof : (m is even)



Sum of the completion times on a set of m/2 machines ≤ 2 x Sum of the completion times of the same tasks on m machines.

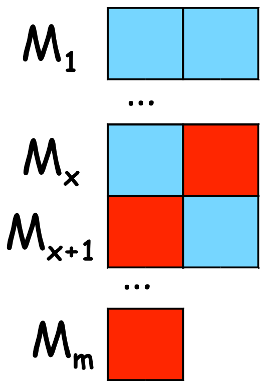
The SPT policy : lower bound

- Property : If all the machines use the SPT policy, then there does not always exist $(3/2-\int)$-approximate Nash equilibrium, for all $\int$.
- Proof: 2m-1 tasks of length 1 s.t.

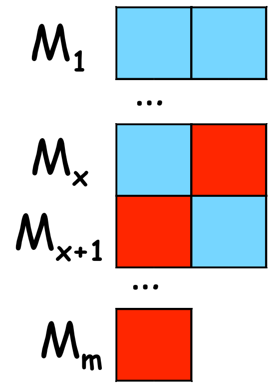
$$A_1 < B_1 < A_2 < B_2 < \dots < A_{m-1} < B_{m-1} < A_m$$

Let S be the most stable schedule.

- At most 2 tasks per machine in S.
- Each agent can, by moving its tasks, make them start at time 0.



Let x be the number of tasks of A in 1st position in S.



- $\sum C_j = x + 2(m-x)$.
  A could gain a factor $\frac{x+2(m-x)}{m} = 2 - \frac{x}{m}$ by moving.

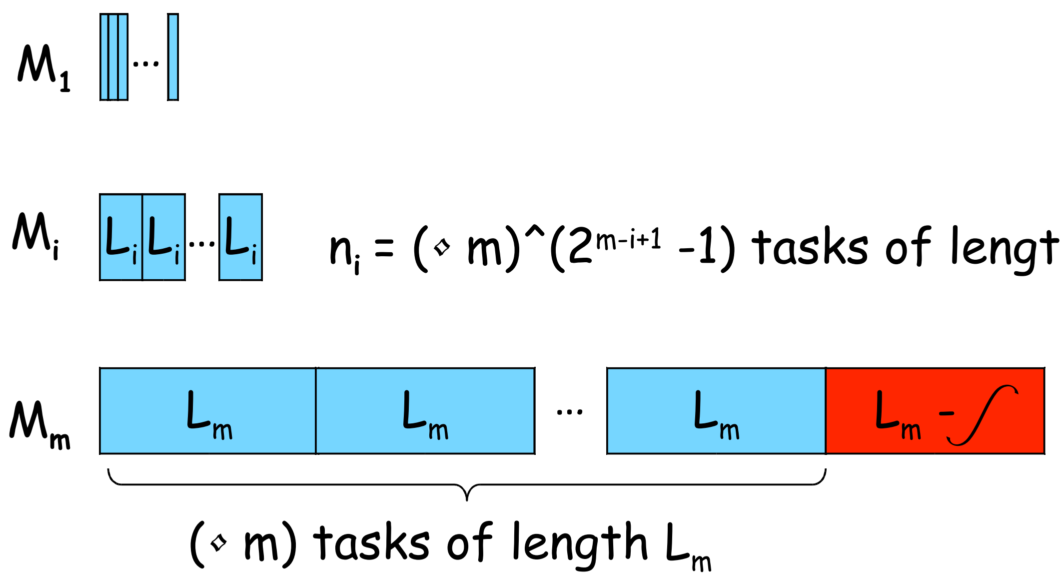
- B could gain a factor $1 + \frac{x}{m-1}$ by moving.

- S is a max($2 - \frac{x}{m}$, $1 + \frac{x}{m-1}$)-approximate Nash equilibrium.

- There is no ⋄-approximate Nash equilibrium with ⋄ < min max($2 - \frac{x}{m}$, $1 + \frac{x}{m-1}$) = $\frac{3}{2}$. $(x \in \{1,\dots,m\})$

The LPT and Random policies

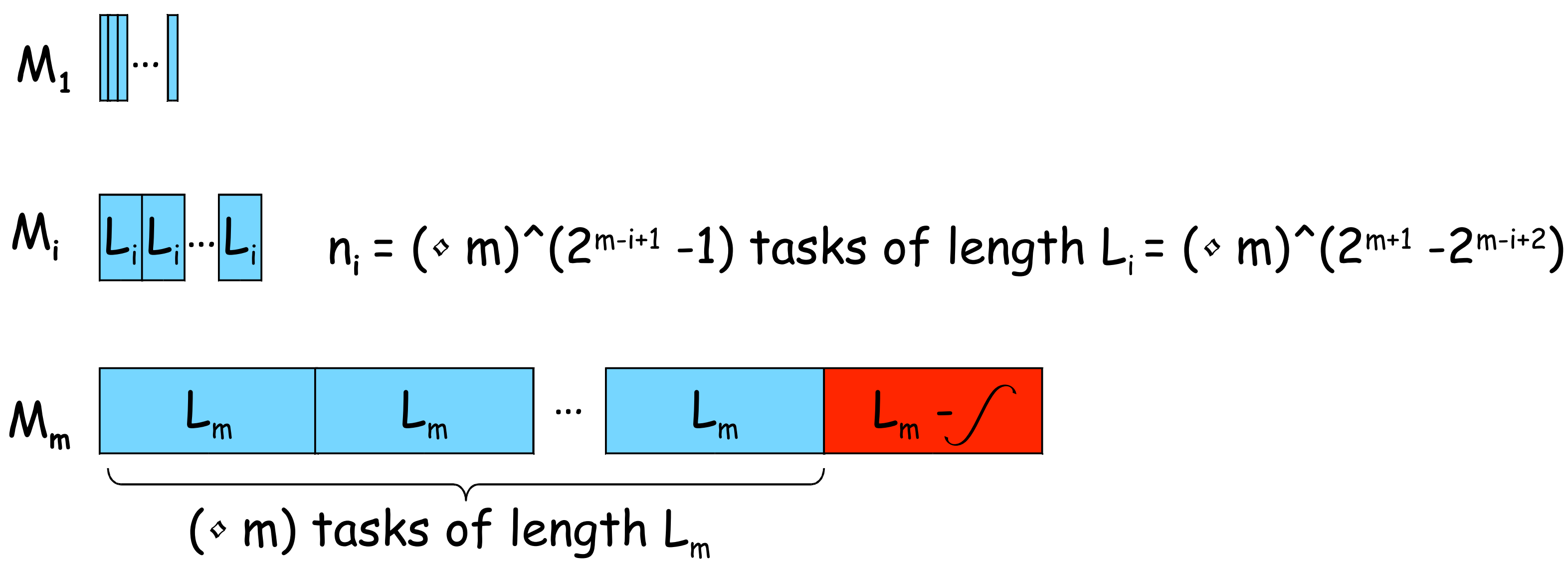
- Property : If the machines use the policy LPT or Random, then there does not always exist an ⋄-approximate Nash equilibrium, for all ⋄.
- Proof for LPT:



n_i = (⋄ m)^(2^{m-i+1} -1) tasks of length L_i = (⋄ m)^(2^{m+1} -2^{m-i+2})

1- In S, agent B decreases its completion time with a factor larger than ⋄ by going on another machine.

2- In any other schedule, agent *A* decreases its sum of completion times by a factor > ⋄ by moving its tasks.



- Sum of the completion times of *A* in S :
  □Cj(S) < m (⋄ m)^(2^{m+1} - 2)
- Sum of the completion times of the tasks of length L_i with a longest task > L_{i+1} n_i = (⋄ m)^(2^{m+1} - 1) > ⋄ □Cj(S)

□ There is no ⋄-approximate Nash equilibrium in this game.

Conclusion

- Machines with deterministic identical policies
  - without idle times : instances without Nash equilibrium
  - with idle times : price of anarchy at least 2 (social cost = sum of completion times).

- Classical policies :
  - LPT, Random induce schedules as instable as wanted
  - SPT induces ⋄-approximate Nash equilibria with ⋄ between 3/2 et 3.

Future work

- Tight bound for SPT
- Complexity for an agent to compute its best response (for a given coordination mechanism) ? Convergence time to obtain a Nash equilibria?
- Does there exist a coordination mechanism which induces Nash equilibria for this problem?

  For example : one machine uses SPT, and another one LPT?