Selfish scheduling with setup times

Laurent Gourvès[1] Jérôme Monnot[1] Orestis A. Telelis[2]

1. CNRS - LAMSADE Université Paris Dauphine
2. Computer Science Department, Aarhus University

September 18, 2009

Table of contents

1. Strategic games
2. Scheduling with setup
   - `Makespan` mechanism
   - Type ordering mechanisms
3. Concluding remarks

Strategic games

- a set of players $P = \{1, 2, \cdots, p\}$
- a *strategy set* Σ_i for every $i \in P$
  - a *pure state* is a vector $S = (S_1, S_2, \cdots, S_p)$ in $\Sigma = \Sigma_1 \times \Sigma_2 \times \cdots \times \Sigma_p$ where S_i is the action of player i
- a function $f_i : \Sigma \rightarrow \mathbb{Z}$ for every $i \in P$

prisoner dilemma

2 players
$\Sigma_1 = \{Silent, Betray\}$
$\Sigma_2 = \{Silent, Betray\}$

	Silent	Betray
Silent	1 1	0 10
Betray	10 0	3 3

Solution concept

Nash equilibrium

State where no player can *unilaterally* change his strategy and benefit

	Silent	Betray
Silent	1 1	0 10
Betray	10 0	3 ⋆ 3

(*Betray*, *Betray*) is the only **pure** Nash equilibrium

Nash equilibrium

Existence of a **pure** Nash equilibrium **not** guaranteed

football

players : goal keeper and stricker

	Left	Right
Left	1 0	0 1
Right	0 1	1 0

Price of Anarchy

How far from socially optimal states are Nash equilibria ?

prisoner dilemma

	Silent	Betray
Silent	1 1 1	0 10 10
Betray	10 10 0	3 3 3

red$= \max\{f_1(S), f_2(S)\}$

Price of Anarchy (PoA)

Worst case ratio between the social cost of a Nash eq. and the socially optimal state

PoA=3 in the example

Analogy with the approximation ratio

Selfish scheduling

Each job is controlled by a player who chooses on which machine his job will be executed

- $P =$ the set of jobs, $\Sigma_i =$ the set of machines

Selfish scheduling

Each job is controlled by a player who chooses on which machine his job will be executed

- $P =$ the set of jobs, $\Sigma_i =$ the set of machines

Each machine has a *public* scheduling policy (algorithm) which, ideally, does not depend on the jobs executed on the other machines

- **Mechanism =** a set of scheduling policies, one per machine

Selfish scheduling

Each job is controlled by a player who chooses on which machine his job will be executed

- $P =$ the set of jobs, $\Sigma_i =$ the set of machines

Each machine has a *public* scheduling policy (algorithm) which, ideally, does not depend on the jobs executed on the other machines

- **Mechanism =** a set of scheduling policies, one per machine

Every player wants to minimize the completion of his own job, no matter how bad the whole schedule can be

- f_i to be minimized

Selfish scheduling

Price of Anarchy for selfish scheduling

$\sup \frac{\text{makespan of Nash eq}}{\text{optimal makespan}}$ over all instances of the game

Remark: an optimum is not necessarily a Nash equilibrium

Selfish scheduling

Price of Anarchy for selfish scheduling

sup $\frac{\text{makespan of Nash eq}}{\text{optimal makespan}}$ over all instances of the game

Remark: an optimum is not necessarily a Nash equilibrium

Questions

1. Which mechanism guarantees that a pure Nash eq. exists ?
2. What is the price of anarchy of these mechanisms ?

bibliography

E. Koutsoupias and C. Papadimitriou, Worst Case Equilibria, STACS '99

T. Roughgarden and E. Tardos, How Bad is Selfish Routing?, JACM '02

and many others

Table of contents

1 Strategic games

2 Scheduling with setup

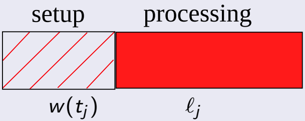
- Makespan mechanism
- Type ordering mechanisms

3 Concluding remarks

Instance

m identical machines, n jobs, k job-types

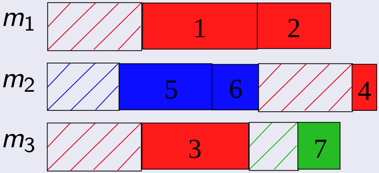
- every job j has a **type** t_j and a **processing length** ℓ_j
- jobs of type θ incur a **setup overhead** of $w(\theta)$



setup = loading packages, running an application, etc a setup is run once on a machine for all jobs of the same type

example

3 machines, 3 job-types (red, blue, green), 7 jobs



$S_1 = 1$ $S_2 = 1$ $S_3 = 3$ $S_4 = 2$ $S_5 = 2$ $S_6 = 2$ $S_7 = 3$

Table of contents

1. Strategic games

2. Scheduling with setup
   - Makespan mechanism
   - Type ordering mechanisms

3. Concluding remarks

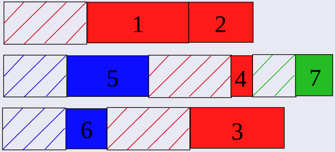
Makespan mechanism

any job's completion time = load of its machine

notation

For a state S:

- $c_j(S)$ = completion time of job j
- $C_i(S)$ = completion time of machine i
- $C(S)$ = makespan



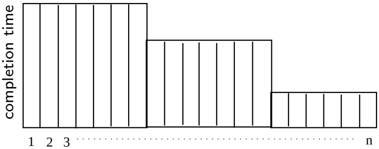
$$c_1(S) = c_2(S) = C_1(S)$$

$$c_4(S) = c_5(S) = c_7(S) = C_2(S)$$

$$c_3(S) = c_6(S) = C_3(S)$$

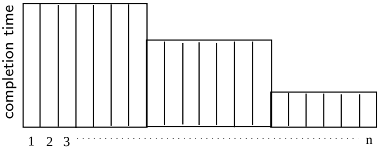
Existence of a pure Nash equilibrium

Associate a vector of length n to every state S such that each coordinate is the completion of a job (sorted by non increasing value)



Existence of a pure Nash equilibrium

Associate a vector of length n to every state S such that each coordinate is the completion of a job (sorted by non increasing value)



Each time a player moves, the vector decreases lexicographically ⇒ A state with lexicographically smallest vector is a pure Nash equilibrium

PoA of the Makespan mechanism

notations

- n jobs ; $\mathcal{J}$ = set of all jobs
- k different job-types ; $\mathcal{T}$ = set of all types
- S = state at Nash equilibrium ; S^* = optimal state

Lower bounds on $C(S^*)$

1. $mC(S^*) \geq \sum_{\theta \in \mathcal{T}} w(\theta) + \sum_{j \in \mathcal{J}} \ell_j$
2. $C(S^*) \geq w(t_j) + \ell_j$ for all $j \in \mathcal{J}$
3. $(k-1)C(S^*) \geq \sum_{\theta \in \mathcal{T} \setminus \{\xi\}} w(\theta)$ for all $\xi \in \mathcal{T}$

PoA of the Makespan mechanism : case $m \leq k$

upper bound

$$C(S) \leq \sum_{\theta \in \mathcal{T}} w(\theta) + \sum_{j \in \mathcal{J}} \ell_j \leq mC(S^*) \Rightarrow PoA = \frac{C(S)}{C(S^*)} \leq m$$

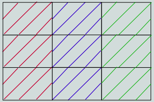
lower bound

Suppose that $m = k$.

For each type θ: $w(\theta) = 1$, m jobs of length 0

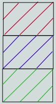
Nash eq. with makespan m

One job of each type on every machine



Optimum with makespan 1

Same type jobs on a dedicated machine



PoA of the Makespan mechanism : case $m > k$

Assume $C(S) = C_1(S)$ w.l.o.g.
For any job j on machine 1, and a machine $i \neq 1$:

- $c_j(S) \leq C_i(S) + w(t_j) + \ell_j$ if t_j does not appear on machine i
- $c_j(S) \leq C_i(S) + \ell_j$ if t_j already appears on machine i

$$\begin{aligned}
(m-1)c_j(S) &\leq \sum_{i \neq 1} C_i(S) + \alpha w(t_j) + (m-1)\ell_j \\
C_1(S) + (m-1)c_j(S) &\leq \sum_{i=1}^{m} C_i(S) + \alpha w(t_j) + (m-1)\ell_j \\
m\, C_1(S) &\leq m \sum_{\theta \in \mathcal{T}} w(\theta) + \sum_{j \in \mathcal{J}} \ell_j + (m-1)\ell_j
\end{aligned}$$

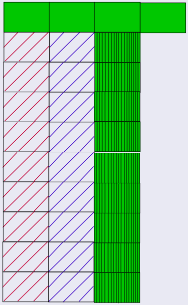
PoA of the Makespan mechanism : case $m > k$

$$\begin{aligned}
m\, C_1(S) &\leq m \sum_{\theta \in \mathcal{T}} w(\theta) + \sum_{j \in \mathcal{J}} \ell_j + (m-1)\ell_j \\
m\, C_1(S) &\leq (m-1)\Big(\sum_{\theta \in \mathcal{T} \setminus \{t_j\}} w(\theta) + w(t_j) + \ell_j \Big) + \sum_{\theta \in \mathcal{T}} w(\theta) + \sum_{j \in \mathcal{J}} \ell_j \\
C_1(S) &\leq \frac{m-1}{m}\Big(\sum_{\theta \in \mathcal{T} \setminus \{t_j\}} w(\theta) + w(t_j) + \ell_j \Big) + \frac{1}{m}\Big(\sum_{\theta \in \mathcal{T}} w(\theta) + \sum_{j \in \mathcal{J}} \ell_j \Big) \\
C(S) &\leq \frac{m-1}{m}((k-1)C(S^*) + C(S^*)) + C(S^*) = (k+1-\frac{k}{m})C(S^*) \\
PoA &\leq k+1-\frac{k}{m}
\end{aligned}$$

PoA of the Makespan mechanism : case $m > k$

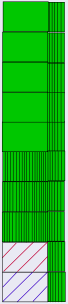
Lower bound when $k = 3$

Nash Equilibrium



makespan $k + 1$

Optimum



makespan $1 + \epsilon$

PoA

Theorem

Under the makespan mechanism, the PoA of the scheduling game with setup times is $\min\{m, k+1-\epsilon\}$

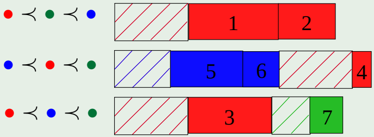
Table of contents

1. Strategic games
2. Scheduling with setup
   - Makespan mechanism
   - Type ordering mechanisms
3. Concluding remarks

Type ordering mechanisms

The scheduling policy of every machine i is as follows:

- batch scheduling of same type jobs
- preemptive execution of all jobs in a batch s.t. completion of a job = completion time of its batch
- type batches are executed serially, following an order $\prec_i$ on the type indexes



$$c_1(S) = c_2(S)$$
$$c_5(S) = c_6(S)$$

Existence

Theorem

A pure Nash equilibrium exists for every type ordering mechanism

Constructive proof

$\prec := \prec_1$

Start from an empty solution and repeat until all jobs are assigned

1. Find the earliest type θ according to $\prec$, with at least one unassigned job.
2. Let j be the largest length unassigned job with $t_j = \theta$.
3. Pick $i \in \mathcal{M}$ minimizing completion time of j (break ties in favor of $i \in M_\prec$).
4. **If** $i \in M_\prec$ set $S_j = i$ **else** $\prec := \prec_i$.

A general Lower bound

Lemma Erdos-Szekeres (1935), Seidenberg (1959)

Every sequence of x distinct numbers possesses a monotone subsequence of size at least $\sqrt{x}$

1 2 3 4 5 6 7 8 9

3 2 1 6 5 4 9 8 7

A general Lower bound

Lemma Erdos-Szekeres (1935), Seidenberg (1959)

Every sequence of x distinct numbers possesses a monotone subsequence of size at least $\sqrt{x}$

1 2 3 4 5 6 7 8 9

3 2 1 6 5 4 9 8 7

Corollary

If $k \geq x^{2^{m-1}}$ then there is a subset of x types $\theta_1, \cdots, \theta_x$ such that

$$\theta_1 \prec_i \theta_2 \prec_i \cdots \prec_i \theta_x \text{ or } \theta_x \prec_i \theta_{x-1} \prec_i \cdots \prec_i \theta_1$$

for all machine i

A general Lower bound

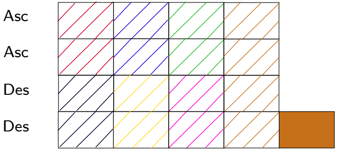
If there are $2m-1$ types $\theta_1, \cdots, \theta_{2m-1}$ such that

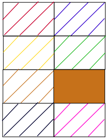
- $\theta_1 \prec \theta_2 \prec \cdots \prec \theta_{2m-1}$ holds on α machines
- $\theta_{2m-1} \prec \theta_{2m-2} \prec \cdots \prec \theta_1$ holds on δ machines

then one can build an instance with PoA $\geq \frac{m+1}{2}$

Ascending order ● $\prec$ ● $\prec$ ● $\prec$ ● $\prec$ ● $\prec$ ● $\prec$ ●

Descending order ● $\prec$ ● $\prec$ ● $\prec$ ● $\prec$ ● $\prec$ ● $\prec$ ●





Nash eq. with makespan $m+1$

Optimum with makespan 2

A general Lower bound

Theorem

when $k \geq (2m-1)^{2^{m-1}}$, every type ordering mechanism has a PoA$\geq \frac{m+1}{2}$

An optimal mechanism

mechanism `A-D`

Half of the machines schedules the batches by **ascending** index
Half of the machines schedules the batches by **descending** index

Theorem

Under the `A-D` mechanism, the PoA of the scheduling game with setup times is $\min\{\frac{m+1}{2}, \frac{k+3}{2} - \epsilon\}$

Table of contents

Strong equilibrium

No group of players can change their strategy and reach a state where they **all** benefit

- existence of strong equilibria for `makespan` and `A-D`
- PoA for strong equilibria

Open problems

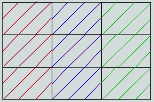
- Better coordination mechanisms for identical machines
- Other machine environments

Thank you!

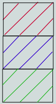
A general Lower bound

If there are m types $\theta_1, \cdots, \theta_m$ such that $\theta_1 \prec \theta_2 \prec \cdots \prec \theta_m$ on every machine then one can build an instance with PoA $\geq m$

- $w(\theta_1) = w(\theta_2) = \ldots = w(\theta_m) = 1$
- m jobs with length 0 per type



Nash eq. with makespan m



Optimum with makespan 1

Strong Equilibria

Strong equilibrium

No group of players can change their strategy and reach a state where they **all** benefit

mechanism `Makespan`

A strong equilibrium always exists

The PoA for strong equilibria is

- $3/2$ when $m = 2$
- 2 when $m \geq 3$

mechanism `A-D`

A strong equilibrium exists when $m = 2$, open when $m \geq 3$

PoA for strong equilibria $=$ PoA for Nash equilibria