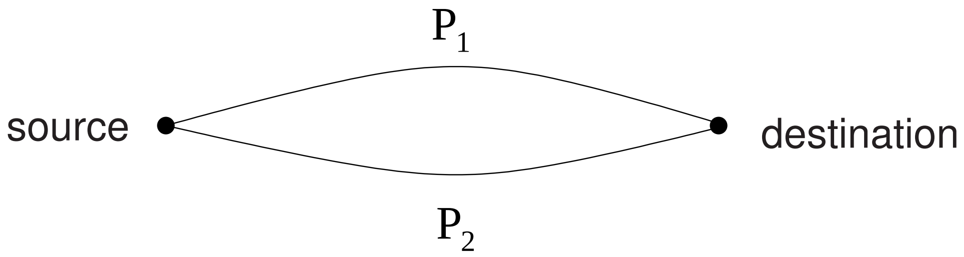
The price of approximate stability for a scheduling problem

Eric Angel, Evripidis Bampis, Fanny Pascual

IBISC, University of Evry

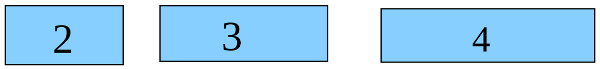
Introduction

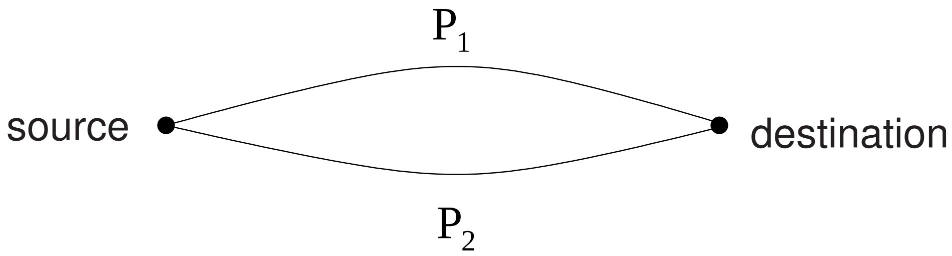
Context : Optimisation of the performances of a network used by independant and selfish users.



Introduction

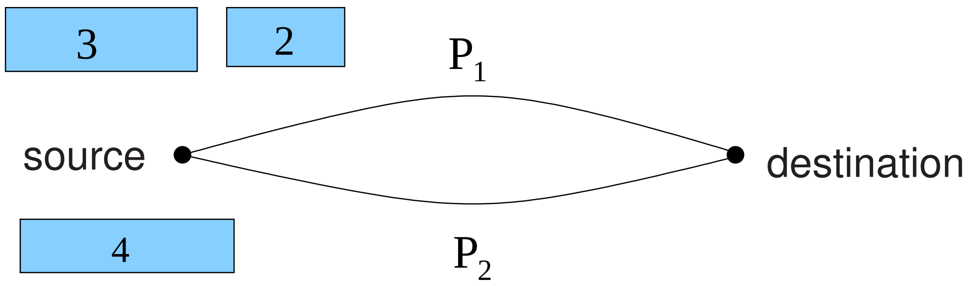
Context : Optimisation of the performances of a network used by independant and selfish users.

a set of packets : 



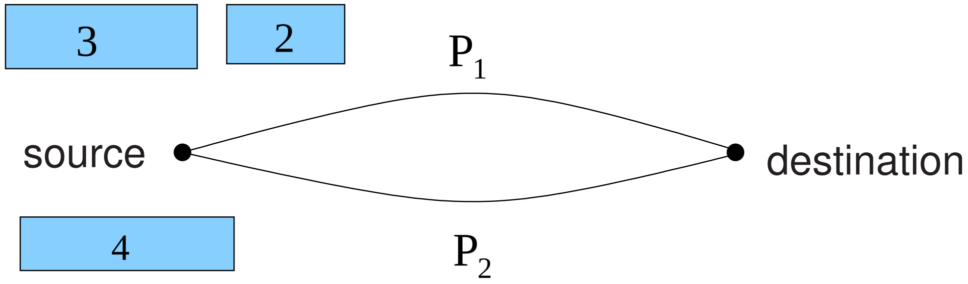
Introduction

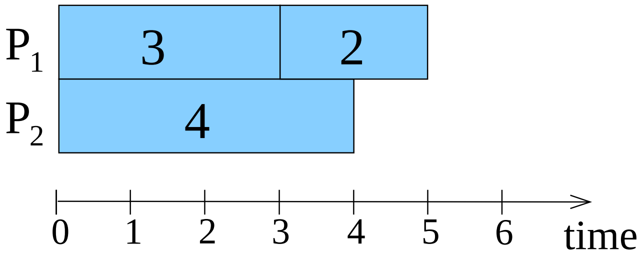
Context : Optimisation of the performances of a network used by independant and selfish users.



Introduction

Context : Optimisation of the performances of a network used by independant and selfish users.





Introduction

A non-cooperative game :

- A set of agents
- a set of possible strategies for each agent
- a function which, depending of the strategies chosen by the agents, associates a cost c_i to each agent i

Nash equilibrium (NE) : Situation in which no agent can decrease its cost by unilaterally changing strategy

Introduction

A non-cooperative game :

- A set of agents
  $\rightarrow n$ tasks
- a set of possible strategies for each agent
- a function which, depending of the strategies chosen by the agents, associates a cost c_i to each agent i

Nash equilibrium (NE) : Situation in which no agent can decrease its cost by unilaterally changing strategy

Introduction

A non-cooperative game :

- A set of agents
  $\rightarrow n$ tasks
- a set of possible strategies for each agent
  $\rightarrow$ strategy 1 = go on P1 ; strategy 2 = go on P2
- a function which, depending of the strategies chosen by the agents, associates a cost c_i to each agent i

Nash equilibrium (NE) : Situation in which no agent can decrease its cost by unilaterally changing strategy

Introduction

A non-cooperative game :

- A set of agents
  $\rightarrow n$ tasks
- a set of possible strategies for each agent
  $\rightarrow$ strategy 1 = go on P1 ; strategy 2 = go on P2
- a function which, depending of the strategies chosen by the agents, associates a cost c_i to each agent i
  $\rightarrow c_i$ = completion time of tash (agent) i

Nash equilibrium (NE) : Situation in which no agent can decrease its cost by unilaterally changing strategy

Introduction

A non-cooperative game :

- A set of agents
  $\rightarrow n$ tasks
- a set of possible strategies for each agent
  $\rightarrow$ strategy 1 = go on P1 ; strategy 2 = go on P2
- a function which, depending of the strategies chosen by the agents, associates a cost c_i to each agent i
  $\rightarrow c_i$ = completion time of tash (agent) i

Nash equilibrium (NE) : Situation in which no agent can decrease its cost by unilaterally changing strategy
$\rightarrow$ NE : no task can decrease its completion time by going on the other machine.
= Stability

Introduction

n tasks (agents) to schedule on 2 machines.

- *Individual objective function :*

  Goal of each task : to minimize its completion time.

- *Global objective function :*

  Our goal : to minimize the makespan (the maximum completion time).

Introduction

- Each task knows the length and strategies chosen by the other tasks. It also knows the order in which each machine schedules its tasks.
- no cooperation between the tasks.

Each task chooses on which machine it will be scheduled.

Goal : Obtain a schedule which minimizes the makespan.

Outline

- Introduction
- How to cope with selfish tasks
  - Without protocol : the price of anarchy
  - With a protocol : the price of stability
  - The price of α-approximate stability

Outline

- Introduction
- How to cope with selfish tasks
  - Without protocol : the price of anarchy
  - With a protocol : the price of stability
  - The price of α-approximate stability
- Our work : Study of the price of α-approximate stability
- Conclusion and future work

Without protocol

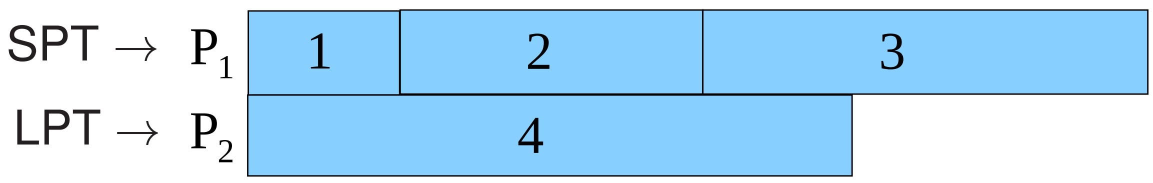
A policy per machine (a local algo. which says how to schedule the tasks) = *coordination mechanism*.

Without protocol

A policy per machine (a local algo. which says how to schedule the tasks) = *coordination mechanism*.

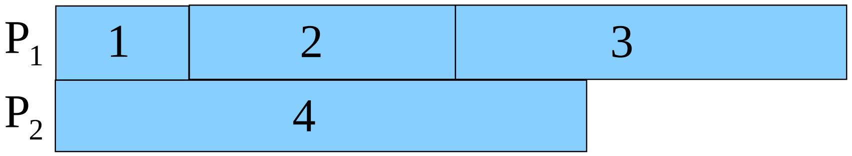
Example : [Christodoulou et al, ICALP 2004]
Policy of the 1st machine = SPT (from the smallest task to the largest task).

Policy of the 2nd machine = LPT (from the largest task to the smallest task).

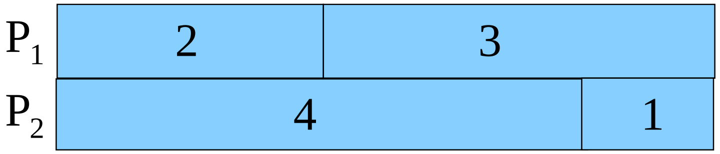


Nash equilibrium

Without protocol

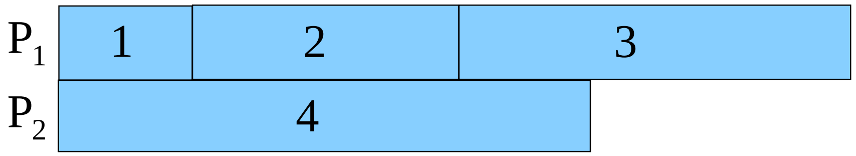


Nash equilibrium

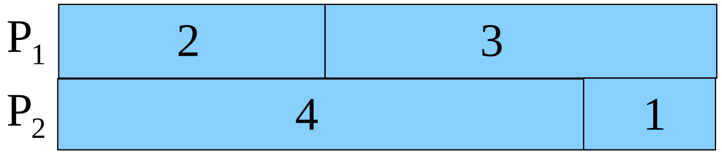


OPT

Without protocol



Nash equilibrium



OPT

Definition : [Koutsoupias and Papadimitriou, STACS 1999]

$$\text{Price of anarchy} = \frac{\text{Makespan in the worst NE}}{\text{Makespan in an optimal solution}}.$$

Example :

Policy of the P_1 = SPT ; policy of the P_2 = LPT

Price of anarchy = 4/3.

Without protocol

Best known coordination mechanism :
Policy of each machine = LPT (+ small delays between tasks).
$\rightarrow$ Price of anarchy = 7/6.

Conjecture : [Christodoulou et al, ICALP 2004]
There is no coord. mechanism with a price of anarchy smaller than 7/6.

With a protocol :

We have :

- A policy per machine.
- A protocol which suggests an assignment of the tasks on the machines.

The tasks accept or refuse this assignment.

With a protocol :

We have :

- A policy per machine.
- A protocol which suggests an assignment of the tasks on the machines.

The tasks accept or refuse this assignment.

Goal : A protocol which returns a solution :

- which minimizes the makespan
- and which is stable.

With a protocol :

We have :

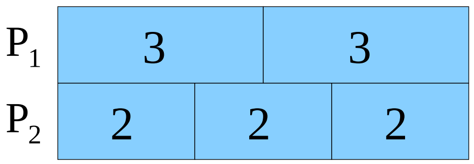
- A policy per machine.
- A protocol which suggests an assignment of the tasks on the machines.

The tasks accept or refuse this assignment.

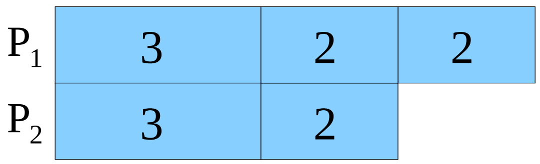
Goal : A protocol which returns a solution :

- which minimizes the makespan
- and which is stable.

Example : If the policy of each machine is LPT :



Not stable



Stable

Price of stability

Introduced in [Schultz et al, SODA 2003] and [Anshelevich et al, FOCS 2004].

Definition :

$$\text{Price of stability} = \frac{\text{Makespan in the best NE}}{\text{Makespan in an optimal solution}}.$$

Recall :

$$\text{Price of anarchy} = \frac{\text{Makespan in the worst NE}}{\text{Makespan in an optimal solution}}.$$

Example : If the policy of each machine is LPT, then the price of stability is 7/6.

Price of α-approximate stability

α-approximate Nash equilibrium = situation in which no agent decreases its cost by a factor greater than α by changing strategy.

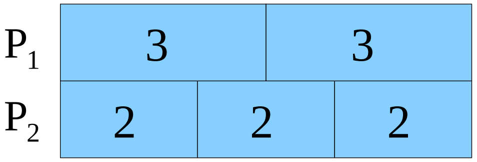
= situation in which no task can decrease its completion time by more than α by changing machine.

Price of α-approximate stability

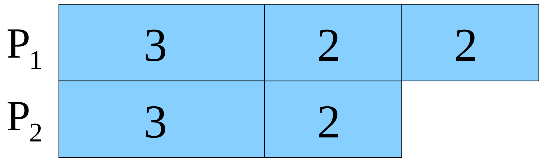
α-approximate Nash equilibrium = situation in which no agent decreases its cost by a factor greater than α by changing strategy.

= situation in which no task can decrease its completion time by more than α by changing machine.

Example : Policy of each machine = LPT.



2-approximate NE



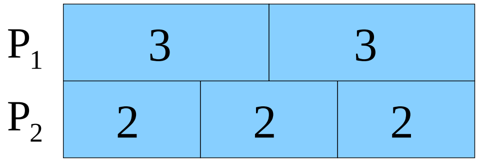
NE

Price of α-approximate stability

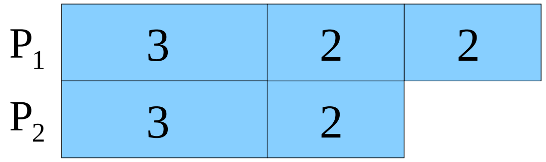
α-approximate Nash equilibrium = situation in which no agent decreases its cost by a factor greater than α by changing strategy.

= situation in which no task can decrease its completion time by more than α by changing machine.

Example : Policy of each machine = LPT.



2-approximate NE

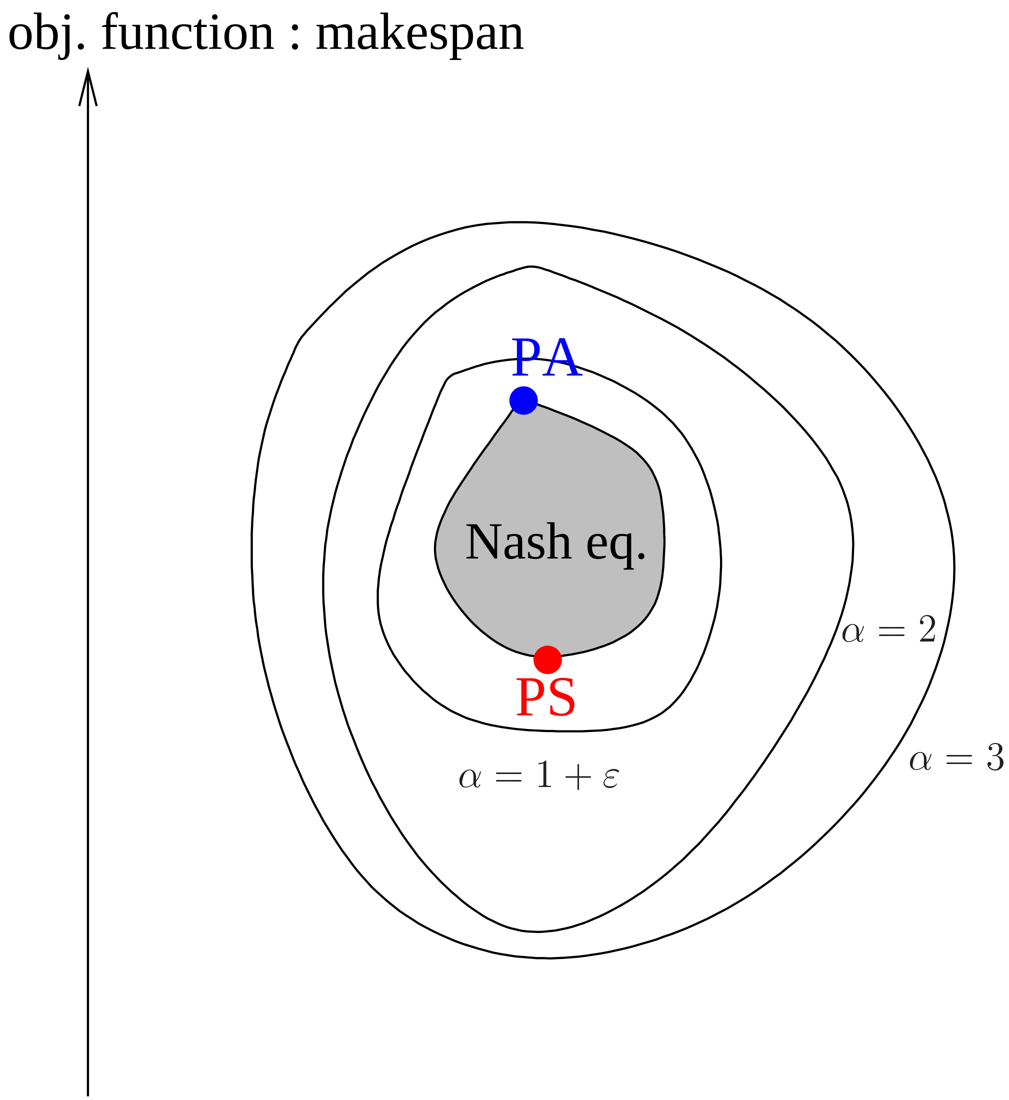


NE

Price of α-approximate stability [Chen and Roughgarden, SPAA 2006]

$$= \frac{\text{Makespan in the best } \alpha\text{-approx. NE}}{\text{Makespan in an optimal solution}}$$

Price of α-approximate stability



Outline

- Introduction
- How to cope with selfish tasks
  - Without protocol : price of anarchy $\longrightarrow$ Ratio $\geq$ 7/6 (LPT)
  - Protocol : price of stability $\longrightarrow$ Ratio = 7/6 (LPT)
  - Price of α-approximate stability $\longrightarrow$ Ratio = $f(\alpha)$?
- Study of the price of α-approximate stability
- Conclusion and future work

Our goal

Policy of the machines = LPT.

What is the price of α-approximate stability ?

Given (r, α), is there a r-approximate algorithm which returns α-approximate NE ?

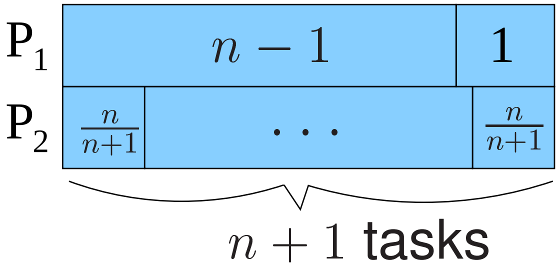
Lower bound (policy = LPT)

Theorem : Let $n > 5$. There is no algorithm with an approx. ratio $< (1 + \frac{1}{n(n+1)})$ which returns α-approximate NE with $\alpha < n$.

Lower bound (policy = LPT)

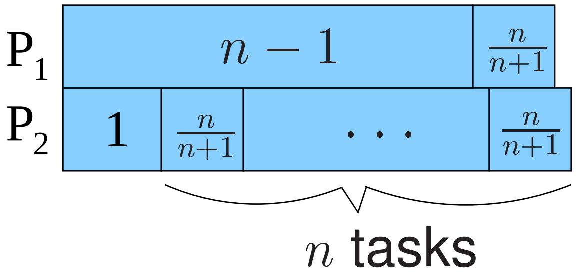
Theorem : Let $n > 5$. There is no algorithm with an approx. ratio $< (1 + \frac{1}{n(n+1)})$ which returns α-approximate NE with $\alpha < n$.

Sketch of the proof :



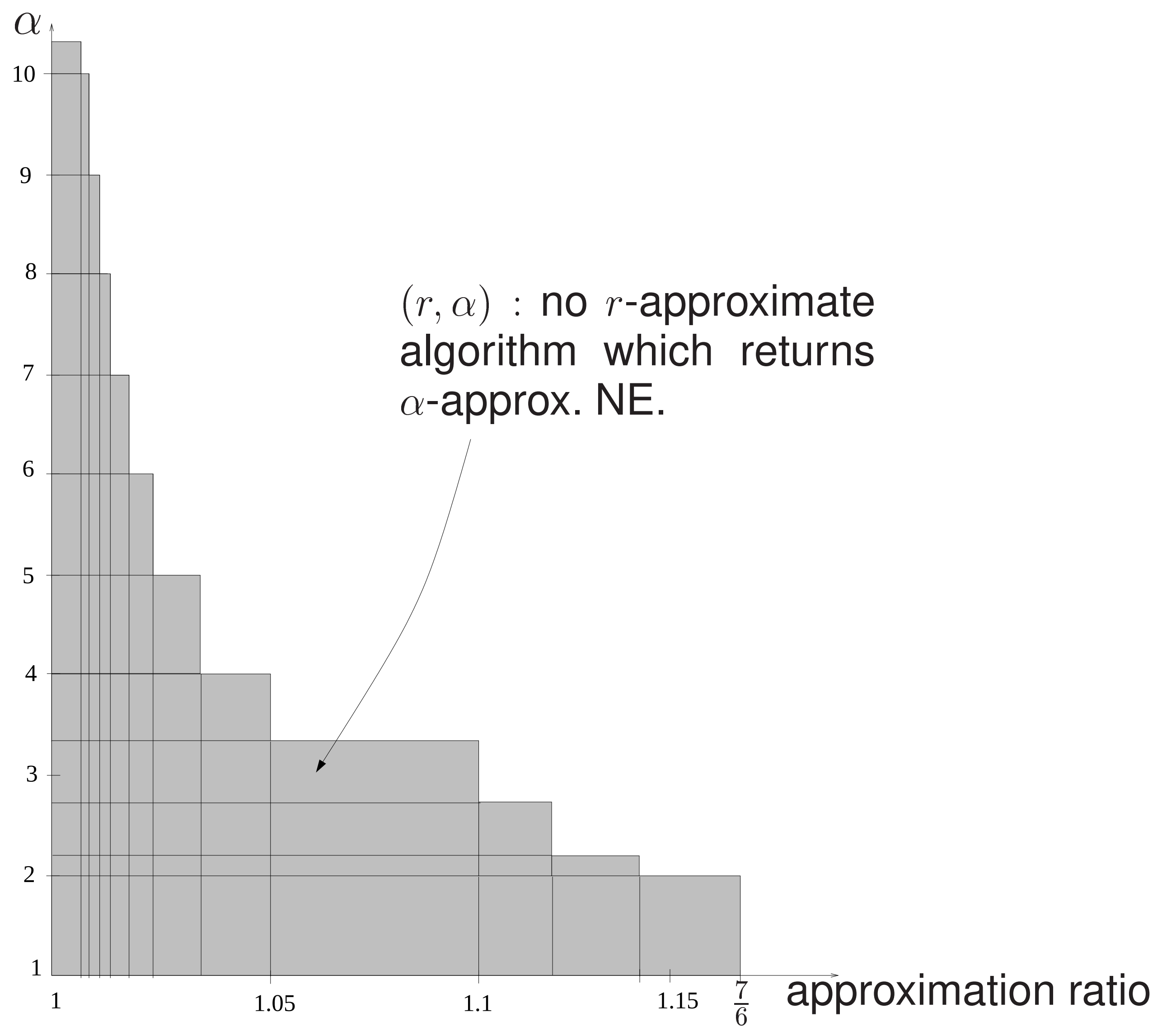
approx. ratio < $1 + \frac{1}{n(n+1)}$

n-approximate NE



approx ratio = $1 + \frac{1}{n(n+1)}$

Lower bound (policy = LPT)



Upper bound (policy = LPT)

Theorem : There is a $(1 + \frac{1}{\alpha})$-approximate algorithm which returns α-approximate NE.

Upper bound (policy = LPT)

Theorem : There is a $(1 + \frac{1}{\alpha})$-approximate algorithm which returns α-approximate NE.

Algorithm OPT-LPT(k) : PTAS of Graham [Graham, 1966]

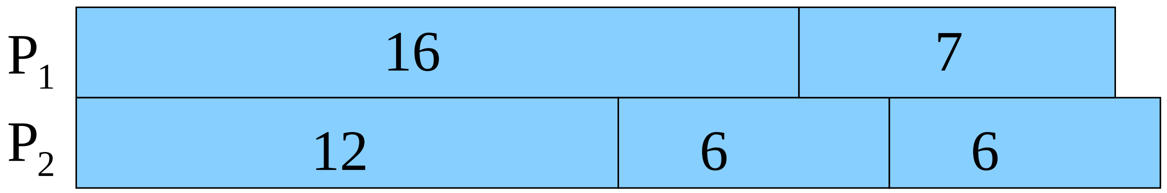
- Let $k > 0$
- Get an optimal schedule of the k largest tasks such that :
  - On each machine tasks are scheduled with the LPT policy.
  - If two tasks have the same length, the one with the smallest ID is scheduled first.
- Schedule the $n - k$ remaining tasks with the LPT rule.

Upper bound (policy = LPT)

Example : Tasks of lengths $16,\ 12,\ 7,\ 6,\ 6,\ 5,\ 3,\ 3,\ 1$.
Schedule returned by OPT-LPT(5) :

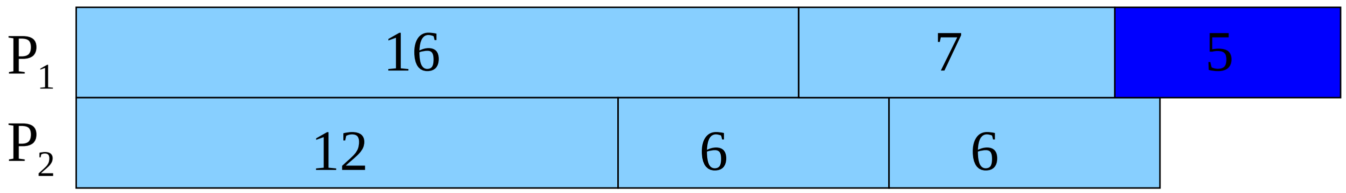
Upper bound (policy = LPT)

Example : Tasks of lengths $16, \ 12, \ 7, \ 6, \ 6, \ 5, \ 3, \ 3, \ 1$.
Schedule returned by OPT-LPT(5) :



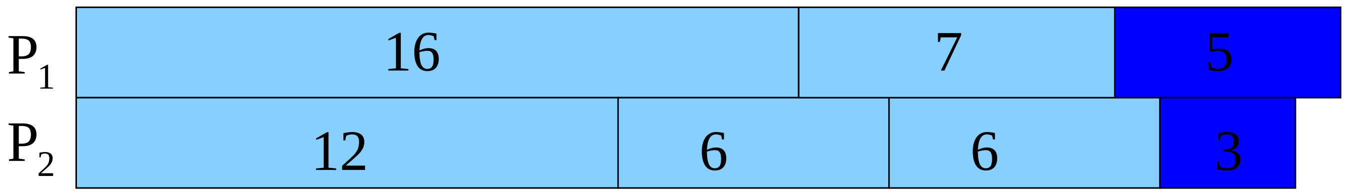
Upper bound (policy = LPT)

Example : Tasks of lengths $16,\ 12,\ 7,\ 6,\ 6,\ 5,\ 3,\ 3,\ 1$.
Schedule returned by OPT-LPT(5) :



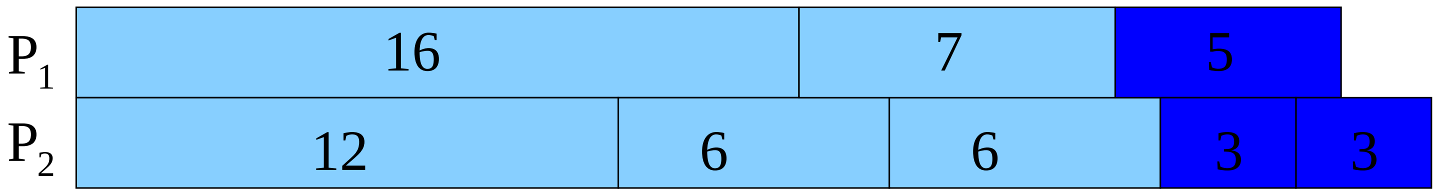
Upper bound (policy = LPT)

Example : Tasks of lengths $16,\ 12,\ 7,\ 6,\ 6,\ 5,\ 3,\ 3,\ 1$.
Schedule returned by OPT-LPT(5) :



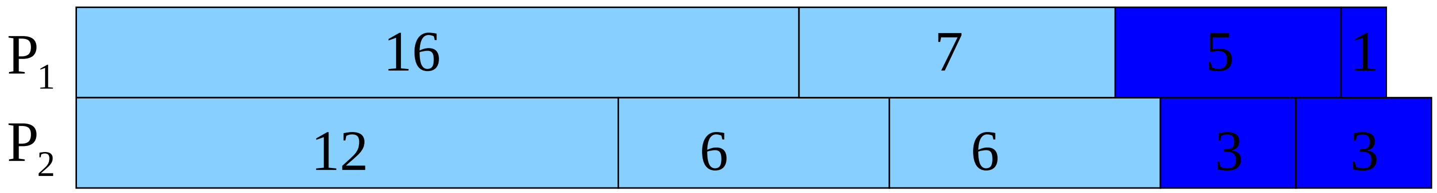
Upper bound (policy = LPT)

Example : Tasks of lengths $16,\ 12,\ 7,\ 6,\ 6,\ 5,\ 3,\ 3,\ 1$.
Schedule returned by OPT-LPT(5) :



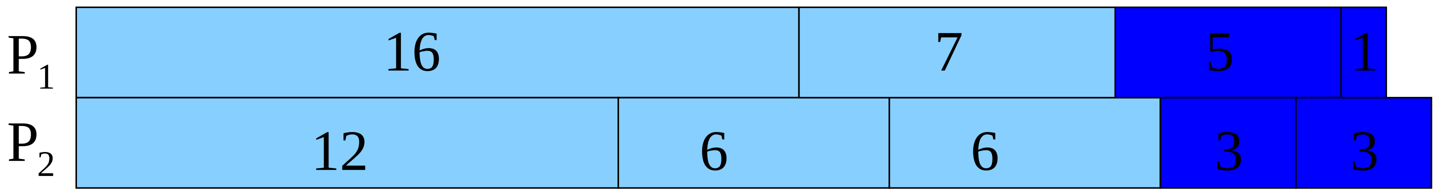
Upper bound (policy = LPT)

Example : Tasks of lengths $16,\ 12,\ 7,\ 6,\ 6,\ 5,\ 3,\ 3,\ 1$.
Schedule returned by OPT-LPT(5) :



Upper bound (policy = LPT)

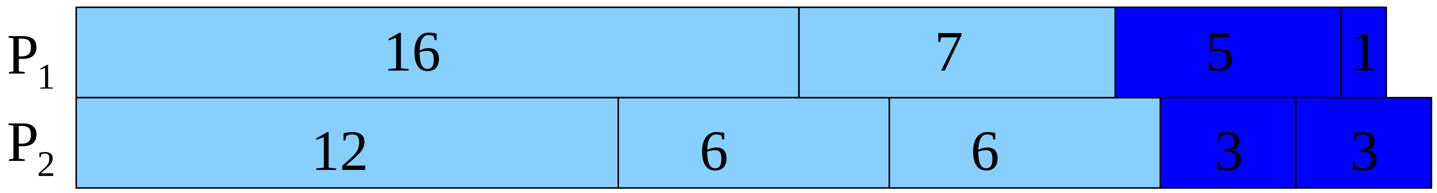
Example : Tasks of lengths $16,\ 12,\ 7,\ 6,\ 6,\ 5,\ 3,\ 3,\ 1$.
Schedule returned by OPT-LPT(5) :



OPT-LPT(k) returns $(k-2)$-approximate Nash equilibria.

Upper bound (policy = LPT)

Example : Tasks of lengths $16,\ 12,\ 7,\ 6,\ 6,\ 5,\ 3,\ 3,\ 1$.
Schedule returned by OPT-LPT(5) :



OPT-LPT(k) returns $(k-2)$-approximate Nash equilibria.

Sketch of the proof :

- The small tasks do not have incentive to change machine.
- If there is $k-1$ large tasks on the same machine, schedule = NE.
  Otherwise a large task can overtake at most $k-2$ tasks and these tasks are smaller than it.

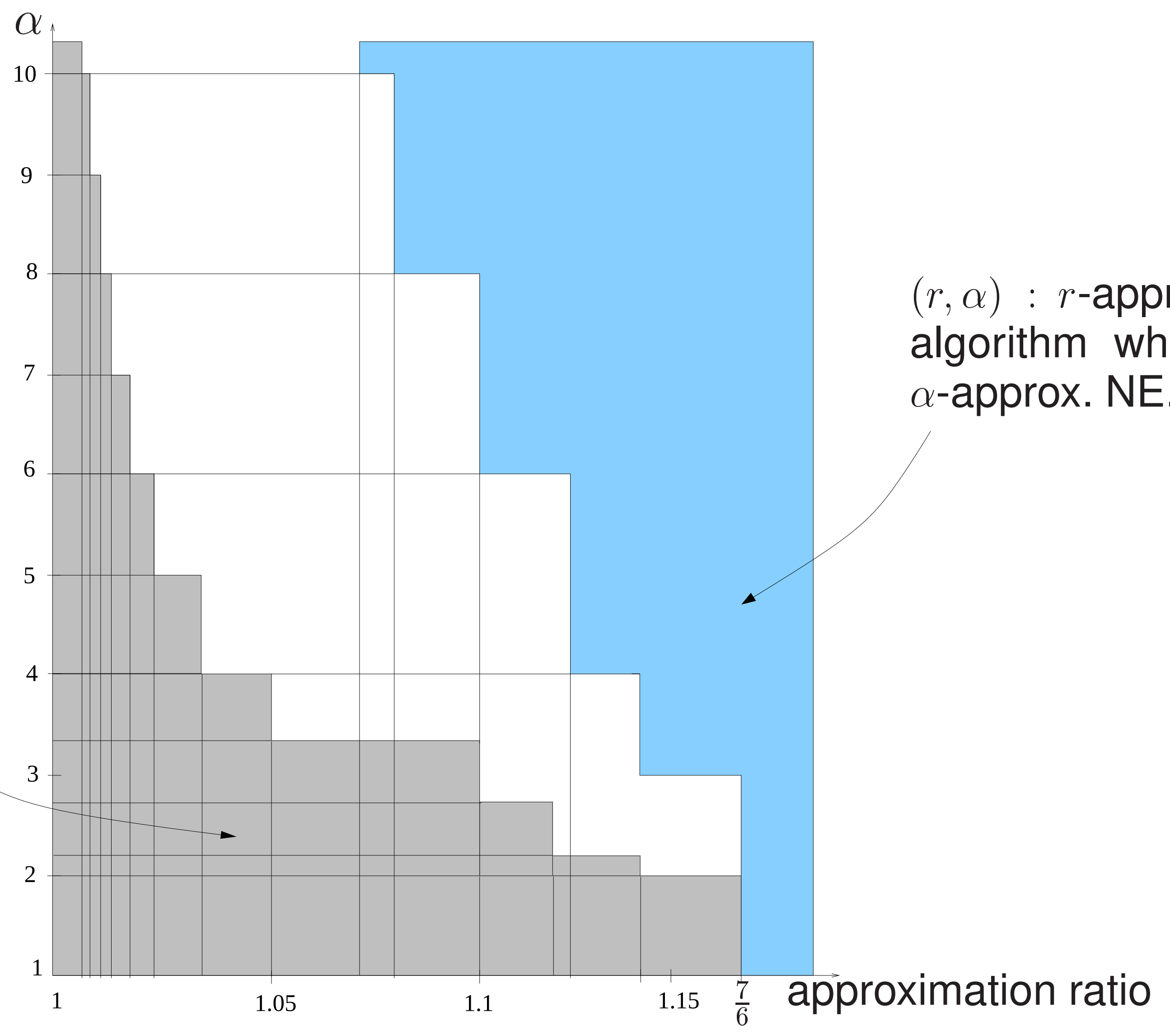
Upper bound (policy = LPT)

- Approximation ratio of OPT-LPT(k) = $1+\frac{1}{2+2\lfloor\frac{k}{2}\rfloor} < 1+\frac{1}{k-2}$. [Graham, 1969]
- OPT-LPT(k) returns $(k-2)$-approximate Nash equilibria.

$\Rightarrow$ There is a $(1+\frac{1}{\alpha})$-approximate algorithm which returns α-approximate NE.

Results : (policy = LPT)

(r, α) : no r-approximate algorithm which returns α-approx. NE.



(r, α) : r-approximate algorithm which returns α-approx. NE.

Conclusion and future work

Conclusion :

We studied the tradeoff between the approx. ratio and the stability of a schedule.

- We bounded the price of α-approx. stability with the LPT policy.
- The SPT policy is worse than the LPT policy.

Conclusion and future work

Conclusion :

We studied the tradeoff between the approx. ratio and the stability of a schedule.

- We bounded the price of α-approx. stability with the LPT policy.
- The SPT policy is worse than the LPT policy.

Future work :

- Fill the gap.
- Is there a deterministic policy better than LPT ?
- Study the price of α-approximate stability for other problems.