A bicriteria PTAS for Real-time Scheduling with fixed priorities

Thomas Rothvoß
University of Paderborn

8.01.08

This is joint work with Fritz Eisenbrand

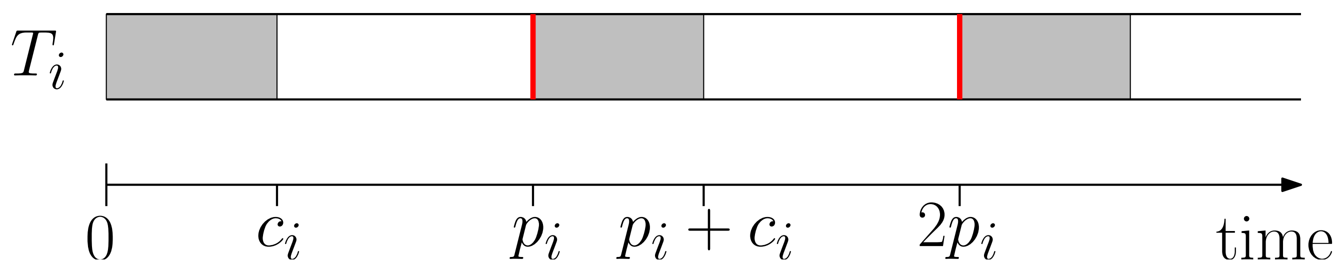
Content of the talk

1. Preliminaries
2. Rounding the instance
3. The algorithm for “large” tasks
4. Dealing with “small” tasks
5. Open problems

Definition

Problem: Given periodic Tasks $T_1, ..., T_n$ with implicit deadlines such that each T_i has running time c_i and period p_i.
Task T_i generates a job of running time c_i each p_i time units, that has to be completed before its period ends.

Goal: Distribute tasks among as few processors as possible using preemptive scheduling.



Dynamic priorities

Theorem. *Dynamic priorities & preemptive Scheduling: Earliest-Deadline First is optimal*

Def.: $\frac{c_i}{p_i} =$ *utilization* of T_i

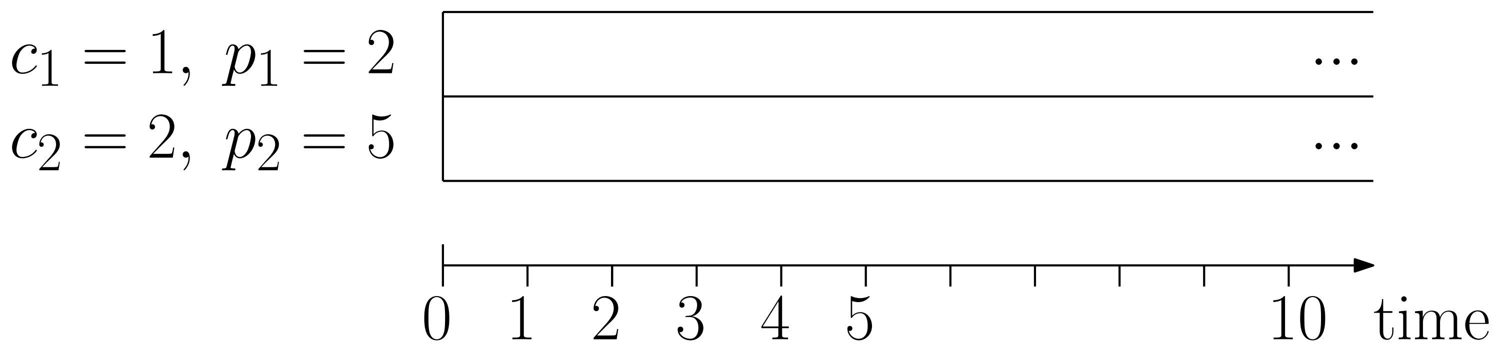
Theorem. *[Liu, Layland '73] If dynamic priorities are allowed: Tasks are feasible on a single processor* $\Leftrightarrow \sum_{i=1}^{n} \frac{c_i}{p_i} \leq 1$

$\Rightarrow$ Bin Packing with item sizes $\frac{c_i}{p_i}$ and bin size 1.

Fixed priorities

Theorem. *[Liu, Layland '73] Optimal priorities:* $\frac{1}{p_i}$ *for* T_i *(Rate-monotonic Schedule)*

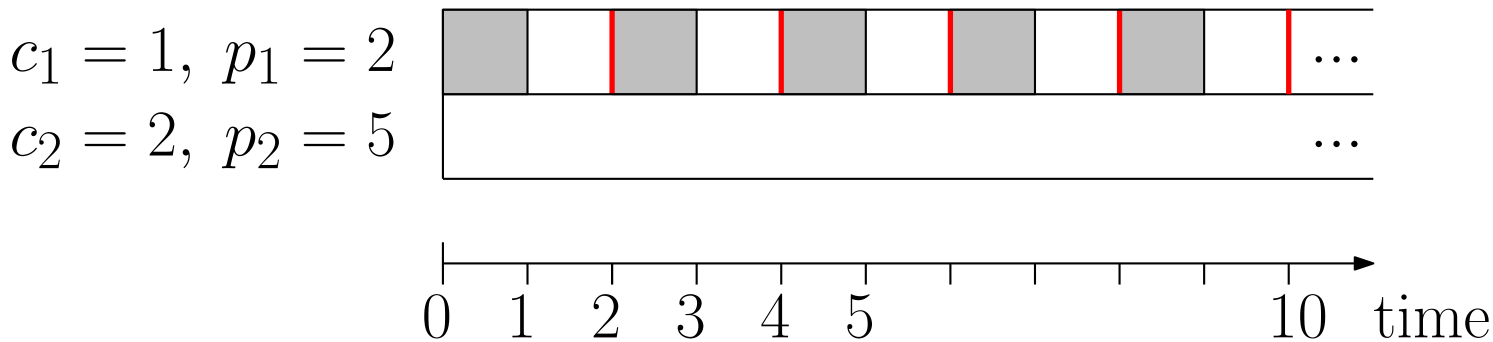
Example



Fixed priorities

Theorem. *[Liu, Layland '73] Optimal priorities:* $\frac{1}{p_i}$ *for* T_i *(Rate-monotonic Schedule)*

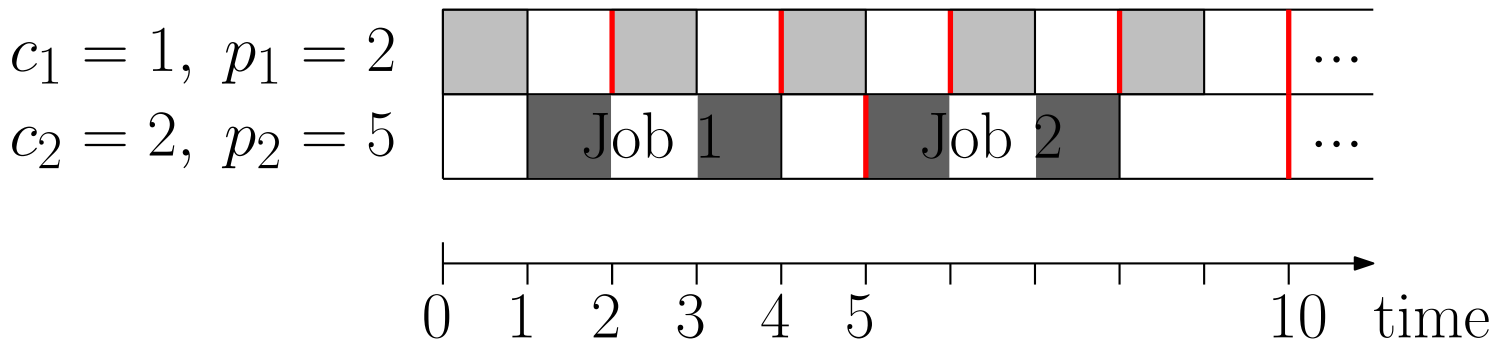
Example



Fixed priorities

Theorem. *[Liu, Layland '73] Optimal priorities:* $\frac{1}{p_i}$ *for* T_i *(Rate-monotonic Schedule)*

Example



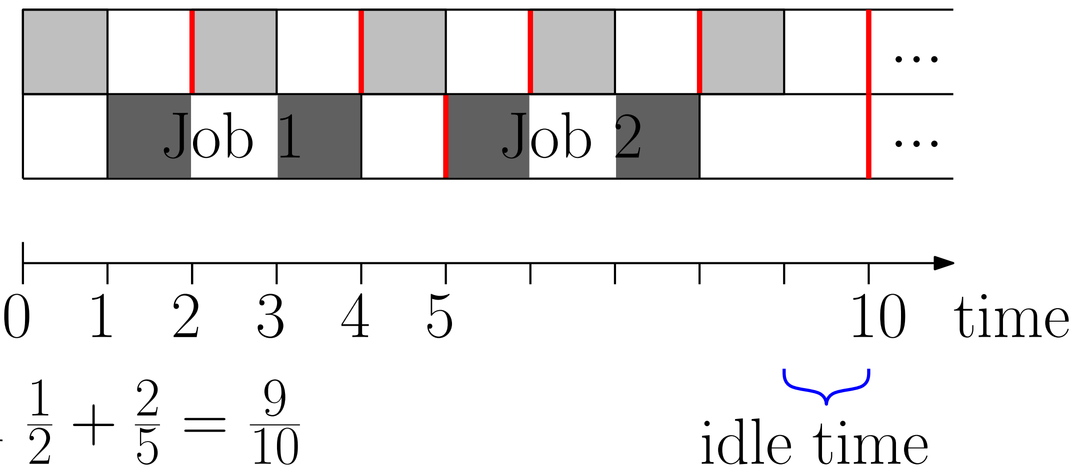
Fixed priorities

Theorem. *[Liu, Layland '73] Optimal priorities:* $\frac{1}{p_i}$ *for* T_i *(Rate-monotonic Schedule)*

Example

$c_1 = 1,\ p_1 = 2$

$c_2 = 2,\ p_2 = 5$



Utilization $\frac{1}{2} + \frac{2}{5} = \frac{9}{10}$

Feasibility

Lemma. *[Liu, Layland '73]* $\sum_{i=1}^{n} \frac{c_i}{p_i} \leq \ln(2) \approx 0.69 \Rightarrow$ *tasks feasible on a single processor*

Def.: $r_i =$ be *Response Time* of $T_i =$ longest time that an instance of task T_i waits for accomplishment

Lemma. *[Lehoczky et al. '89] If* $p_1 \leq ... \leq p_n$ *then* r_i *is the smallest value s.t.*

$$r_i = c_i + \sum_{j<i} \left\lceil \frac{r_i}{p_j} \right\rceil c_j$$

Tasks are feasible $\Leftrightarrow \forall i : r_i \leq p_i$

Local feasibility

Def.: Task T_i is *locally feasible* if $\exists r_i \leq p_i$

$$c_i + \sum_{j<i} \left\lceil \frac{r_i}{p_j} \right\rceil c_j \leq r_i$$

Local feasibility

Def.: Task T_i is *locally feasible* if $\exists r_i \leq p_i$

$$c_i + \sum_{j<i, p_j \leq \varepsilon p_i} \left\lceil \frac{r_i}{p_j} \right\rceil c_j + \sum_{j<i, p_j > \varepsilon p_i} \left\lceil \frac{r_i}{p_j} \right\rceil c_j \leq r_i$$

Local feasibility

Def.: Task T_i is *locally feasible* if $\exists r_i \leq p_i$

$$c_i + \sum_{j<i, p_j \leq \varepsilon p_i} \frac{r_i}{p_j} c_j + \sum_{j<i, p_j > \varepsilon p_i} \left\lceil \frac{r_i}{p_j} \right\rceil c_j \leq r_i$$

Local feasibility

Def.: Task T_i is *locally feasible* if $\exists r_i \leq p_i$

$$c_i + r_i \cdot \sum_{j<i, p_j \leq \varepsilon p_i} \underbrace{\frac{c_j}{p_j}}_{\text{utilization}} + \sum_{j<i, p_j > \varepsilon p_i} \left\lceil \frac{r_i}{p_j} \right\rceil c_j \leq r_i$$

Lemma. *Tasks locally feasible w.r.t.* $\varepsilon > 0 \Rightarrow$ *tasks feasible on a processor of speed* $1 + 2\varepsilon$

Our main result

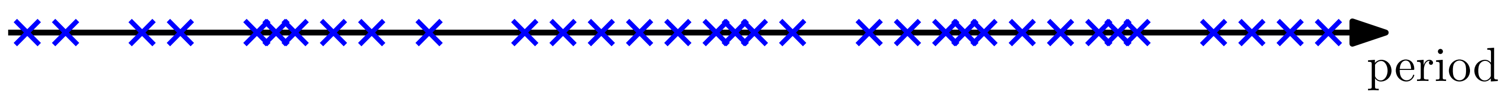
Theorem. *For any* $\varepsilon > 0$ *we can schedule tasks on* $(1+\varepsilon)OPT + O(1)$ *many processors with speed* $1+\varepsilon$ *in polynomial time.*

Recently best algorithm: $\frac{7}{4}$-approximation [Burchard et al. '95]

Rounding the instance

Assume that $\frac{c_i}{p_i} \geq \varepsilon$ and $\frac{1}{\varepsilon} \in \mathbb{Z}$. Round such that

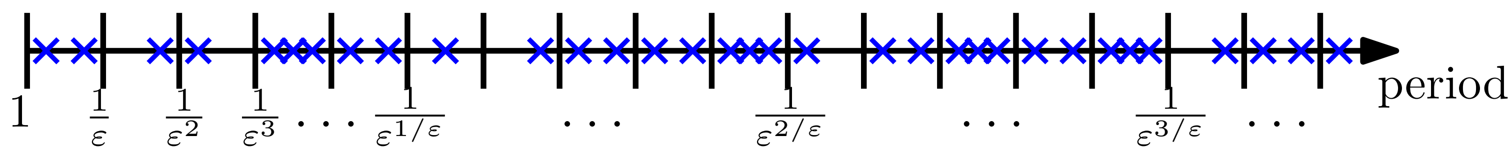
- $p_i = (1+\varepsilon)^{\mathbb{Z}}$
- $\frac{c_i}{p_i} \in \{0, \varepsilon^2, 2\varepsilon^2, ..., 1\}$
- Choose $k \in \{0, ..., 1/\varepsilon - 1\}$ randomly. Remove all tasks having their period in an interval $[1/\varepsilon^i, 1/\varepsilon^{i+1}[$ with $i \equiv_{1/\varepsilon} k$.



Rounding the instance

Assume that $\frac{c_i}{p_i} \geq \varepsilon$ and $\frac{1}{\varepsilon} \in \mathbb{Z}$. Round such that

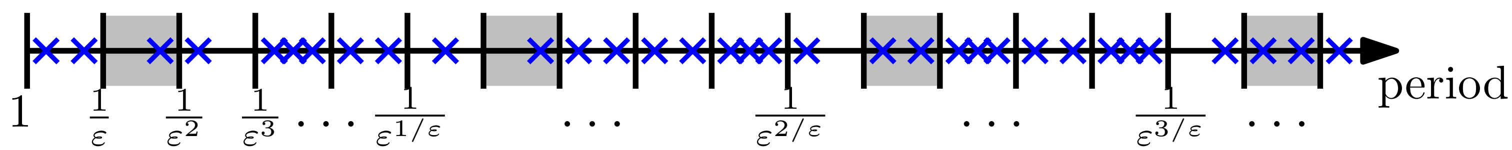
- $p_i = (1+\varepsilon)^{\mathbb{Z}}$
- $\frac{c_i}{p_i} \in \{0, \varepsilon^2, 2\varepsilon^2, ..., 1\}$
- Choose $k \in \{0, ..., 1/\varepsilon - 1\}$ randomly. Remove all tasks having their period in an interval $[1/\varepsilon^i, 1/\varepsilon^{i+1}[$ with $i \equiv_{1/\varepsilon} k$.



Rounding the instance

Assume that $\frac{c_i}{p_i} \geq \varepsilon$ and $\frac{1}{\varepsilon} \in \mathbb{Z}$. Round such that

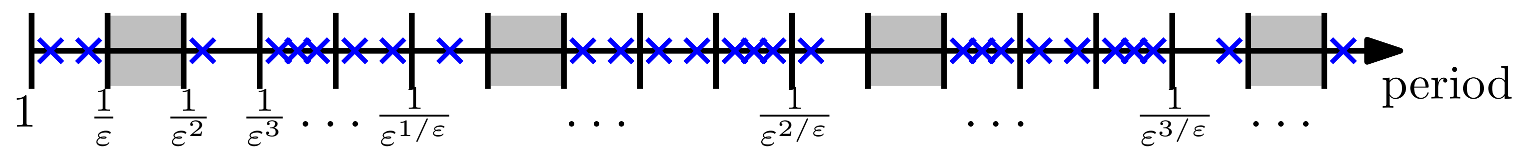
- $p_i = (1+\varepsilon)^{\mathbb{Z}}$
- $\frac{c_i}{p_i} \in \{0, \varepsilon^2, 2\varepsilon^2, ..., 1\}$
- Choose $k \in \{0, ..., 1/\varepsilon - 1\}$ randomly. Remove all tasks having their period in an interval $[1/\varepsilon^i, 1/\varepsilon^{i+1}[$ with $i \equiv_{1/\varepsilon} k$.



Rounding the instance

Assume that $\frac{c_i}{p_i} \geq \varepsilon$ and $\frac{1}{\varepsilon} \in \mathbb{Z}$. Round such that

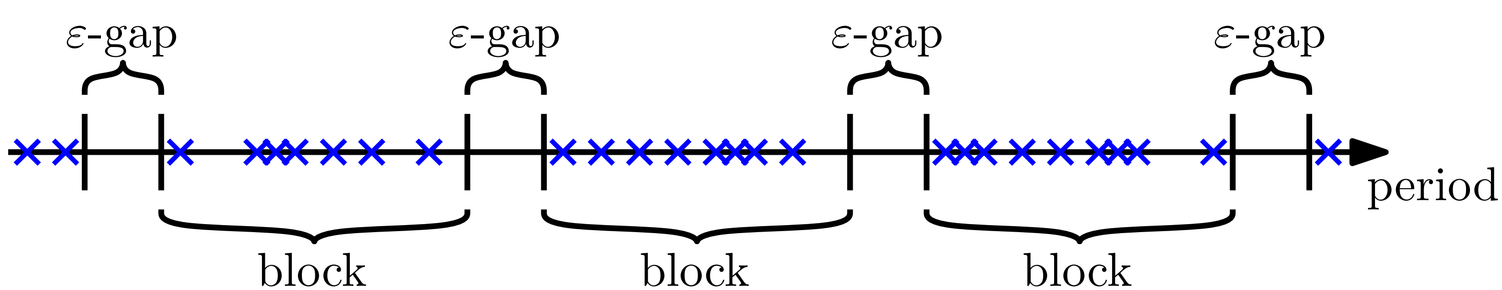
- $p_i = (1+\varepsilon)^{\mathbb{Z}}$
- $\frac{c_i}{p_i} \in \{0, \varepsilon^2, 2\varepsilon^2, ..., 1\}$
- Choose $k \in \{0, ..., 1/\varepsilon - 1\}$ randomly. Remove all tasks having their period in an interval $[1/\varepsilon^i, 1/\varepsilon^{i+1}[$ with $i \equiv_{1/\varepsilon} k$.



Rounding the instance

Assume that $\frac{c_i}{p_i} \geq \varepsilon$ and $\frac{1}{\varepsilon} \in \mathbb{Z}$. Round such that

- $p_i = (1+\varepsilon)^{\mathbb{Z}}$
- $\frac{c_i}{p_i} \in \{0, \varepsilon^2, 2\varepsilon^2, ..., 1\}$
- Choose $k \in \{0, ..., 1/\varepsilon - 1\}$ randomly. Remove all tasks having their period in an interval $[1/\varepsilon^i, 1/\varepsilon^{i+1}[$ with $i \equiv_{1/\varepsilon} k$.



$\Rightarrow$ Blocks $\mathcal{B}_1, ..., \mathcal{B}_k$

Dynamic programming

$$A(a_1, ..., a_n, \ell) \quad = \quad \begin{cases} 1 & \text{if tasks in } \mathcal{B}_1, ..., \mathcal{B}_\ell \text{ can locally feasible} \\ & \text{distributed s.t. processor } i \text{ has util. } \leq a_i \\ 0 & \text{otherwise} \end{cases}$$

Compute

$$A(a_1, ..., a_n, \ell) = 1 \quad \Leftrightarrow \quad \exists 0 \leq b_i \leq a_i : A(b_1, ..., b_n, \ell - 1) \ \&$$

tasks $\mathcal{B}_\ell$ can distributed feasibly

Finally we need

$$\min\{j \mid A(\underbrace{1, ..., 1}_{j-\text{times}}, 0, ..., 0, k) = 1\}$$

many processors.

Distribution of $\mathcal{B}_\ell$

- # of utilization values: $\frac{1}{\varepsilon^2} + 1 = O(1)$
- # of periods in $\mathcal{B}_\ell$: $1 + \log_{1+\varepsilon}(1/\varepsilon)^{1/\varepsilon - 1} = O(1)$
- $\Rightarrow$ # of different task types in $\mathcal{B}_\ell$: $O(1)$
- Utilization $\geq \varepsilon \Rightarrow \leq \frac{1}{\varepsilon}$ tasks per processor
- $\Rightarrow O(1)$ possible packings
- Utilization on processor i can be increased from $b_i \in \varepsilon^2\mathbb{Z}$ to $a_i \in \varepsilon^2\mathbb{Z} \Rightarrow O(1)$ different processor types
- $n^{O(1)}$ many ways to distribute tasks in $\mathcal{B}_\ell$ among the processors

Dealing with small tasks

Partition tasks with util $\leq \varepsilon^6$ in $R_1 \dot{\cup} ... \dot{\cup} R_m$ such that

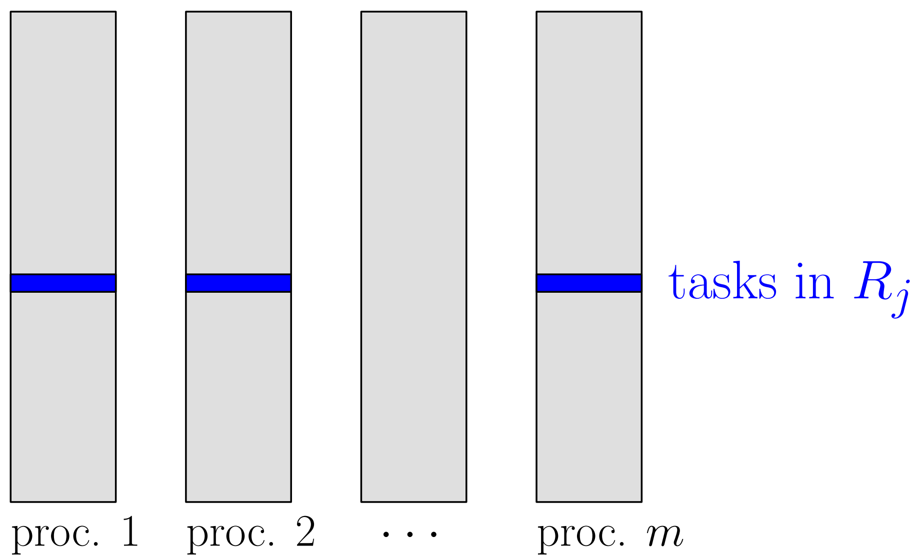
- all tasks in R_i have the same period.
- $\varepsilon^6 \leq$ utilization of $R_i \leq 3\varepsilon^6$

Glue tasks in each R_i together

Merging theorem

Theorem. *$\mathcal{I}'$ merged instance $\Rightarrow \exists$ solution for $\mathcal{I}'$ with $(1+\varepsilon)OPT + O(1)$ processors of speed $1+\varepsilon$*

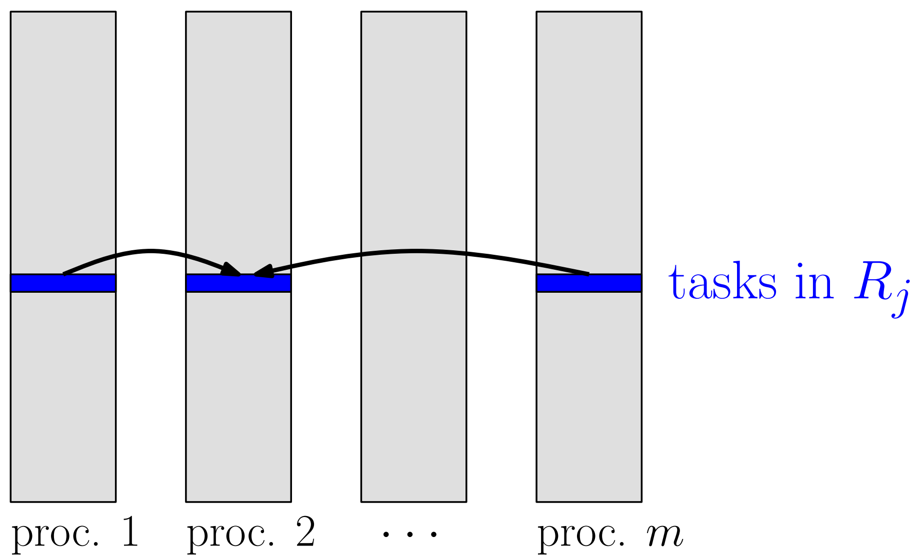
Let $\mathcal{S}_1 \dot{\cup} \ldots \dot{\cup} \mathcal{S}_m$ optimal solution for $\mathcal{I}$. Consider group R_j choose a task $T \in R_j$ randomly with prob $\frac{\text{utilization of } T}{\text{utilization of } R_j}$. Put new task for R_j on T's processor.



Merging theorem

Theorem. *$\mathcal{I}'$ merged instance $\Rightarrow \exists$ solution for $\mathcal{I}'$ with $(1+\varepsilon)OPT + O(1)$ processors of speed $1+\varepsilon$*

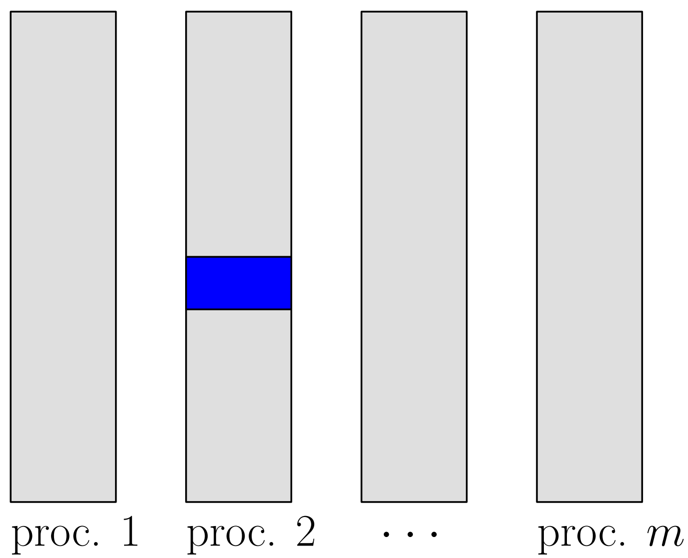
Let $\mathcal{S}_1 \dot\cup ... \dot\cup \mathcal{S}_m$ optimal solution for $\mathcal{I}$. Consider group R_j choose a task $T \in R_j$ randomly with prob $\frac{\text{utilization of } T}{\text{utilization of } R_j}$. Put new task for R_j on T's processor.



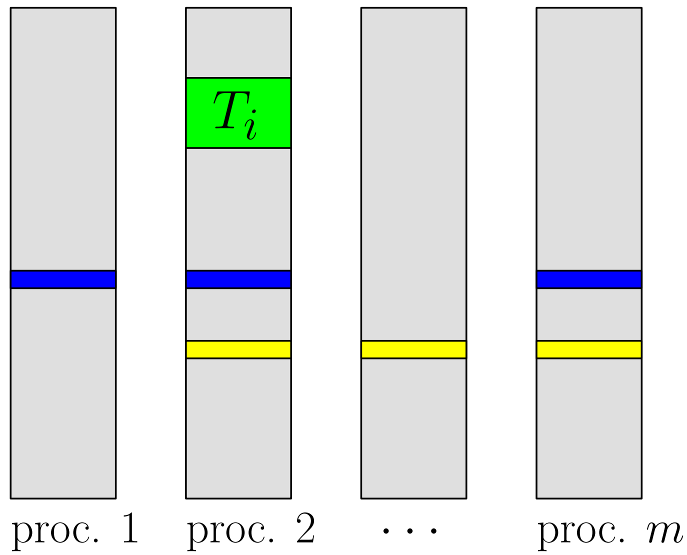
Merging theorem

Theorem. $\mathcal{I}'$ *merged instance* $\Rightarrow \exists$ *solution for* $\mathcal{I}'$ *with* $(1+\varepsilon)OPT + O(1)$ *processors of speed* $1+\varepsilon$

Let $\mathcal{S}_1 \dot\cup ... \dot\cup \mathcal{S}_m$ optimal solution for $\mathcal{I}$. Consider group R_j choose a task $T \in R_j$ randomly with prob $\frac{\text{utilization of } T}{\text{utilization of } R_j}$. Put new task for R_j on T's processor.



Merging theorem



Task T_i still feasible on a processor of speed $1+\varepsilon \Leftrightarrow \exists r_i \leq p_i$:

$$c_i + \sum_{j<i, T_i \text{ large}} \left\lceil \frac{r_i}{p_j} \right\rceil c_j + r_i \sum_{j<i, T_i \text{ small}} \underbrace{\left\lceil \frac{r_i}{p_j} \right\rceil \frac{p_j}{r_i}}_{\in[1,2]} \cdot \underbrace{\frac{c_j}{p_j}}_{\text{utilization}} \leq (1+\varepsilon) r_i$$

Via Chernoff bounds: $\Pr[T_i \text{ is not feasible}] \leq \varepsilon$

If T_i gets infeasible $\Rightarrow$ remove T_i

Open problems

- What about a *real* (asymptotic) PTAS?
- Now running time $n^{g(\varepsilon)}$. Is a bicriteria FPTAS possible or at least running time $f(\varepsilon) \cdot n^{O(1)}$
- Absolutely inefficient in practice! Is there a practicable algorithm (better then First-Fit)?