An Average-Case Analysis for Rate-Monotonic Multiprocessor Real-time Scheduling

Andreas Karrenbauer & Thomas Rothvoß

Institute of Mathematics
EPFL, Lausanne

ESA 2009

DISCRETE OPTIMIZATION

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Real-time Scheduling

Given: synchronous tasks $\tau_1, \ldots, \tau_n$ where task τ_i

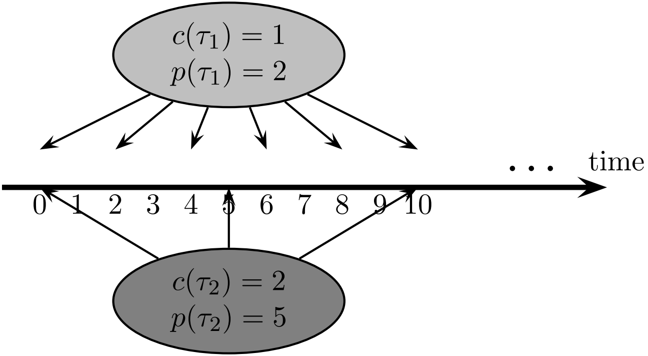
- is periodic with period $p(\tau_i)$
- has running time $c(\tau_i)$
- has implicit deadline

W.l.o.g.: Task τ_i releases job of length $c(\tau_i)$ at $0, p(\tau_i), 2p(\tau_i), \ldots$

Scheduling policy:

- multiprocessor
- fixed priority
- pre-emptive

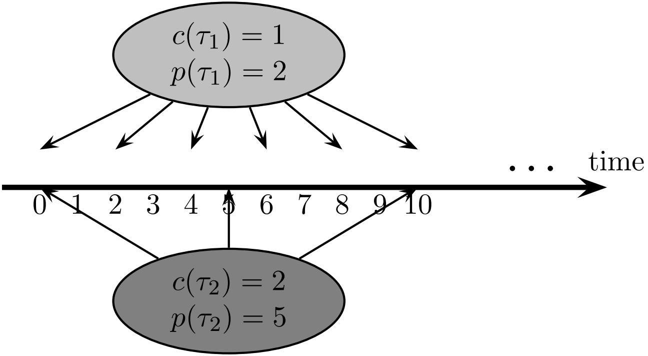
Example



Example

Theorem (Liu & Layland '73)

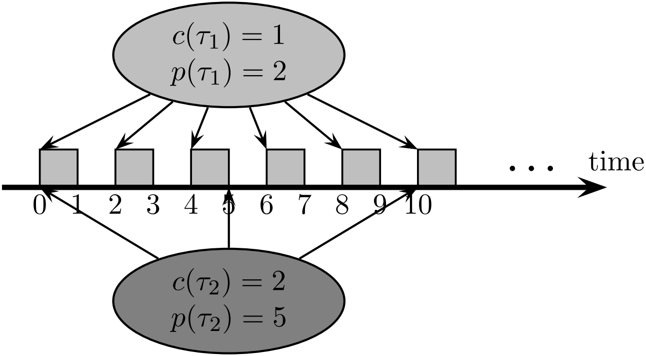
Optimal priorities: $\frac{1}{p(\tau_i)}$ *for* τ_i *(Rate-monotonic Schedule)*



Example

Theorem (Liu & Layland '73)

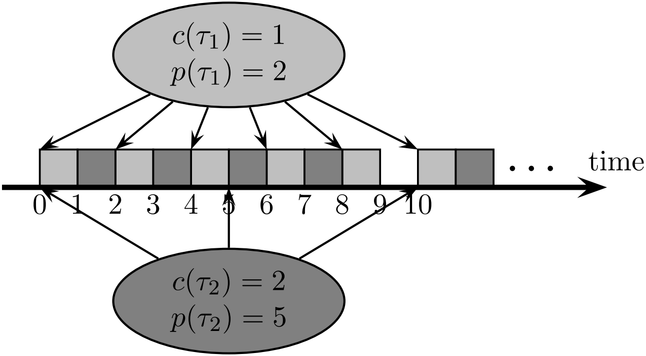
Optimal priorities: $\frac{1}{p(\tau_i)}$ *for* τ_i *(Rate-monotonic Schedule)*



Example

Theorem (Liu & Layland '73)

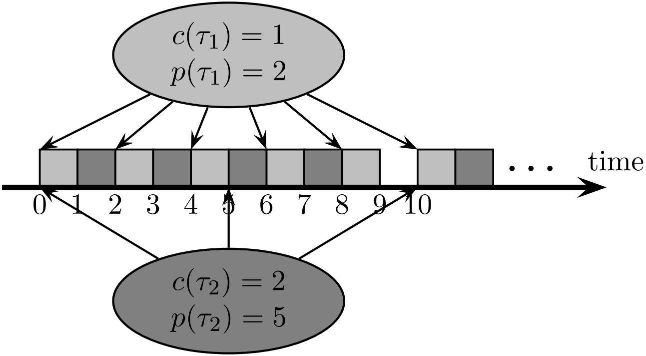
Optimal priorities: $\frac{1}{p(\tau_i)}$ *for* τ_i *(Rate-monotonic Schedule)*



Example

Theorem (Liu & Layland '73)

Optimal priorities: $\frac{1}{p(\tau_i)}$ *for* τ_i *(Rate-monotonic Schedule)*



Definition

$u(\tau) = \frac{c(\tau)}{p(\tau)}$ = **utilization** of task τ

Feasibility test

Theorem (Lehoczky et al. '89)

If $p(\tau_1) \leq \ldots \leq p(\tau_n)$ *then the* **response time** $r(\tau_i)$ *in a RM-schedule is the smallest value s.t.*

$$c(\tau_i) + \sum_{j<i} \left\lceil \frac{r(\tau_i)}{p(\tau_j)} \right\rceil c(\tau_j) \leq r(\tau_i)$$

1 machine suffices $\Leftrightarrow \forall i : r(\tau_i) \leq p(\tau_i)$.

Feasibility test

Theorem (Lehoczky et al. '89)

If $p(\tau_1) \leq \ldots \leq p(\tau_n)$ *then the* **response time** $r(\tau_i)$ *in a RM-schedule is the smallest value s.t.*

$$c(\tau_i) + \sum_{j<i} \left\lceil \frac{r(\tau_i)}{p(\tau_j)} \right\rceil c(\tau_j) \leq r(\tau_i)$$

1 machine suffices $\Leftrightarrow \forall i : r(\tau_i) \leq p(\tau_i)$.

Theorem (Eisenbrand & R. '08)

Computing a response time $r(\tau_i)$ is **NP**-hard.

Feasibility test

Theorem (Lehoczky et al. '89)

If $p(\tau_1) \leq \ldots \leq p(\tau_n)$ *then the* **response time** $r(\tau_i)$ *in a RM-schedule is the smallest value s.t.*

$$c(\tau_i) + \sum_{j<i} \left\lceil \frac{r(\tau_i)}{p(\tau_j)} \right\rceil c(\tau_j) \leq r(\tau_i)$$

1 machine suffices $\Leftrightarrow \forall i : r(\tau_i) \leq p(\tau_i)$.

Theorem (Eisenbrand & R. '08)

Computing a response time $r(\tau_i)$ is **NP**-hard.

Theorem (Liu & Layland '73)

$u(\mathcal{S}) = \sum_{\tau \in \mathcal{S}} u(\tau) \leq \ln(2) \approx 69\% \Rightarrow$ *1 machine suffices*

Feasibility test

Theorem (Lehoczky et al. '89)

If $p(\tau_1) \leq \ldots \leq p(\tau_n)$ *then the* **response time** $r(\tau_i)$ *in a RM-schedule is the smallest value s.t.*

$$c(\tau_i) + \sum_{j<i} \left\lceil \frac{r(\tau_i)}{p(\tau_j)} \right\rceil c(\tau_j) \leq r(\tau_i)$$

1 machine suffices $\Leftrightarrow \forall i : r(\tau_i) \leq p(\tau_i)$.

Theorem (Eisenbrand & R. '08)

Computing a response time $r(\tau_i)$ is **NP**-hard.

Theorem (Liu & Layland '73)

$u(\mathcal{S}) = \sum_{\tau \in \mathcal{S}} u(\tau) \leq \ln(2) \approx 69\% \Rightarrow$ *1 machine suffices*

Observation

If $p(\tau_1) \mid p(\tau_2) \mid \ldots \mid p(\tau_n)$: 1 machine suffices $\Leftrightarrow u(\mathcal{S}) \leq 1$

Feasibility test (2)

Lemma (Burchard et al. '95)

For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

Feasibility test (2)

Lemma (Burchard et al. '95)

For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

$$\alpha(\tau_i) := \log_2 p(\tau_i) - \lfloor \log_2 p(\tau_i) \rfloor$$

Feasibility test (2)

Lemma (Burchard et al. '95)

For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

$$\alpha(\tau_i) := \log_2 p(\tau_i) - \lfloor \log_2 p(\tau_i) \rfloor$$

Feasibility test (2)

Lemma (Burchard et al. '95)

For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

$$\alpha(\tau_i) := \log_2 p(\tau_i) - \lfloor \log_2 p(\tau_i) \rfloor$$

Feasibility test (2)

Lemma (Burchard et al. '95)

For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

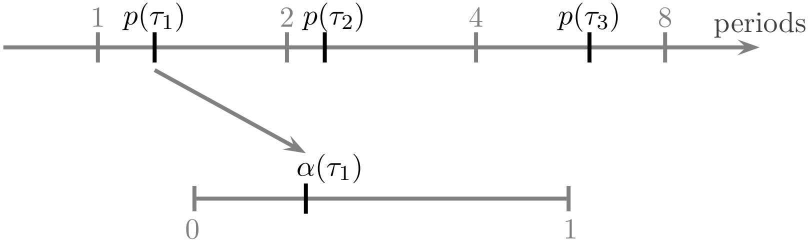
$$\alpha(\tau_i) := \log_2 p(\tau_i) - \lfloor \log_2 p(\tau_i) \rfloor$$

Feasibility test (2)

Lemma (Burchard et al. '95)

For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

$$\alpha(\tau_i) := \log_2 p(\tau_i) - \lfloor \log_2 p(\tau_i) \rfloor$$

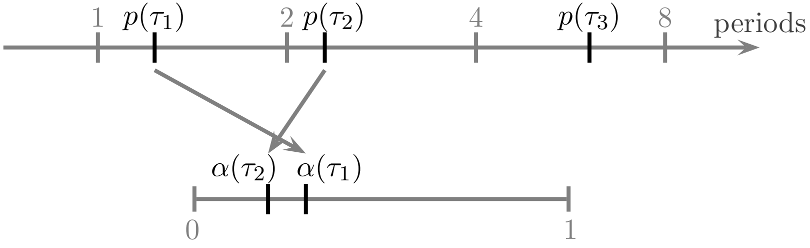


Feasibility test (2)

Lemma (Burchard et al. '95)

For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

$$\alpha(\tau_i) := \log_2 p(\tau_i) - \lfloor \log_2 p(\tau_i) \rfloor$$

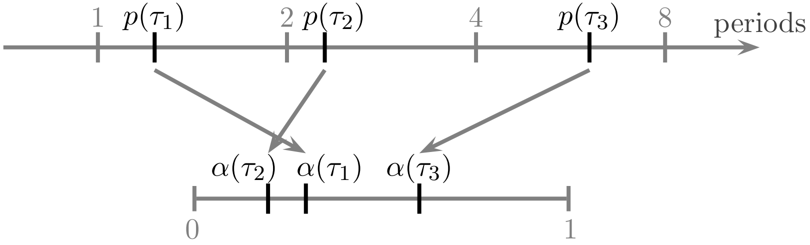


Feasibility test (2)

Lemma (Burchard et al. '95)

For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

$$\alpha(\tau_i) := \log_2 p(\tau_i) - \lfloor \log_2 p(\tau_i) \rfloor$$



Feasibility test (2)

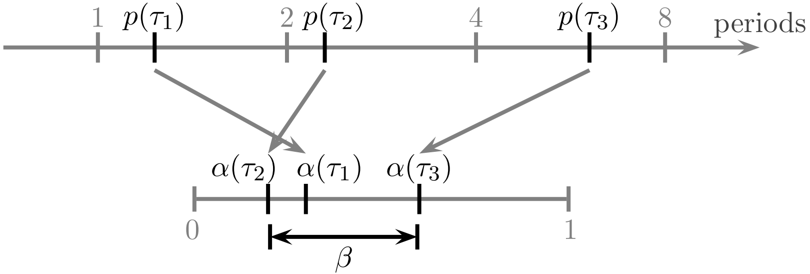
Lemma (Burchard et al. '95)

For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

$$\alpha(\tau_i) := \log_2 p(\tau_i) - \lfloor \log_2 p(\tau_i) \rfloor$$

and

$$\beta(\mathcal{S}) := \max_{i,j} |\alpha(\tau_i) - \alpha(\tau_j)|$$



Feasibility test (2)

Lemma (Burchard et al. '95)

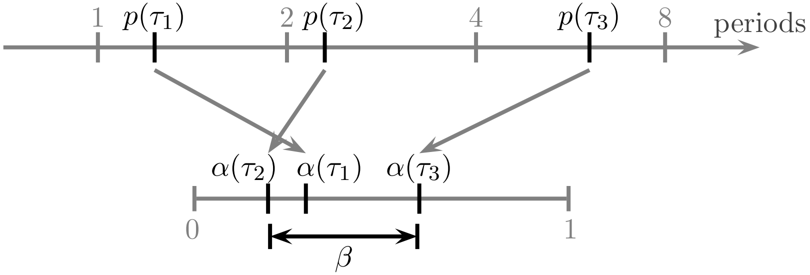
For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ *define*

$$\alpha(\tau_i) := \log_2 p(\tau_i) - \lfloor \log_2 p(\tau_i) \rfloor$$

and

$$\beta(\mathcal{S}) := \max_{i,j} |\alpha(\tau_i) - \alpha(\tau_j)|$$

One machine suffices if $u(\mathcal{S}) \leq 1 - \beta(\mathcal{S})$.



Algorithms for Multiprocessor Scheduling

Algorithms for Multiprocessor Scheduling

Known results:

- **NP**-hard (generalization of Bin Packing)

Algorithms for Multiprocessor Scheduling

Known results:

- **NP**-hard (generalization of Bin Packing)
- asymp. 1.75-apx [Burchard et al. '95]

Algorithms for Multiprocessor Scheduling

Known results:

- **NP**-hard (generalization of Bin Packing)
- asymp. 1.75-apx [Burchard et al. '95]
- asymp. $(1+\varepsilon)$-apx (with resource augmentation) in $\mathcal{O}_\varepsilon(1) \cdot n^{(1/\varepsilon)^{\mathcal{O}(1)}}$ [Eisenbrand & R. '08]

Algorithms for Multiprocessor Scheduling

Known results:

- **NP**-hard (generalization of Bin Packing)
- asymp. 1.75-apx [Burchard et al. '95]
- asymp. $(1 + \varepsilon)$-apx (with resource augmentation) in $\mathcal{O}_\varepsilon(1) \cdot n^{(1/\varepsilon)^{\mathcal{O}(1)}}$ [Eisenbrand & R. '08]

Some papers:

1. Sort tasks w.r.t. some criteria (e.g. $p(\tau_i), u(\tau_i), \alpha(\tau_i)$)

Algorithms for Multiprocessor Scheduling

Known results:

- **NP**-hard (generalization of Bin Packing)
- asymp. 1.75-apx [Burchard et al. '95]
- asymp. $(1+\varepsilon)$-apx (with resource augmentation) in $\mathcal{O}_\varepsilon(1) \cdot n^{(1/\varepsilon)^{\mathcal{O}(1)}}$ [Eisenbrand & R. '08]

Some papers:

1. Sort tasks w.r.t. some criteria (e.g. $p(\tau_i), u(\tau_i), \alpha(\tau_i)$)
2. Distribute tasks in a First Fit/Next Fit way using a sufficient feasibility criterion

Algorithms for Multiprocessor Scheduling

Known results:

- **NP**-hard (generalization of Bin Packing)
- asymp. 1.75-apx [Burchard et al. '95]
- asymp. $(1+\varepsilon)$-apx (with resource augmentation) in $\mathcal{O}_\varepsilon(1) \cdot n^{(1/\varepsilon)^{\mathcal{O}(1)}}$ [Eisenbrand & R. '08]

Some papers:

1. Sort tasks w.r.t. some criteria (e.g. $p(\tau_i), u(\tau_i), \alpha(\tau_i)$)
2. Distribute tasks in a First Fit/Next Fit way using a sufficient feasibility criterion
3. Perform experiments with randomly chosen tasks

Algorithms for Multiprocessor Scheduling

Known results:

- **NP**-hard (generalization of Bin Packing)
- asymp. 1.75-apx [Burchard et al. '95]
- asymp. $(1+\varepsilon)$-apx (with resource augmentation) in $\mathcal{O}_\varepsilon(1) \cdot n^{(1/\varepsilon)^{\mathcal{O}(1)}}$ [Eisenbrand & R. '08]

Some papers:

1. Sort tasks w.r.t. some criteria (e.g. $p(\tau_i), u(\tau_i), \alpha(\tau_i)$)
2. Distribute tasks in a First Fit/Next Fit way using a sufficient feasibility criterion
3. Perform experiments with randomly chosen tasks

Definition

Probability distribution:

- Adversary gives $p(\tau_i)$
- $u(\tau_i) \in [0,1]$ uniformly at random

Waste

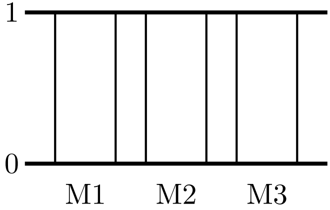
Definition

Waste of a solution: #machines $- u(\mathcal{S})$.

Waste

Definition

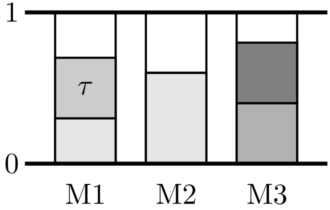
Waste of a solution: #machines $- u(\mathcal{S})$.



Waste

Definition

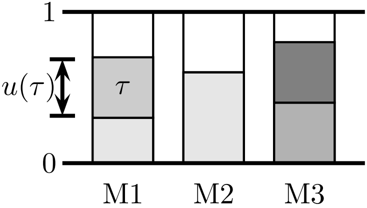
Waste of a solution: #machines $- u(\mathcal{S})$.



Waste

Definition

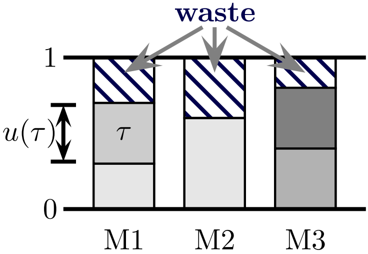
Waste of a solution: $\#\text{machines} - u(\mathcal{S})$.



Waste

Definition

Waste of a solution: #machines $- u(\mathcal{S})$.



Our algorithm

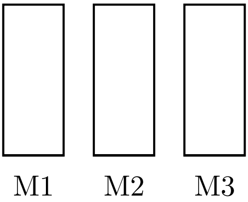
First Fit Matching Periods (FFMP):

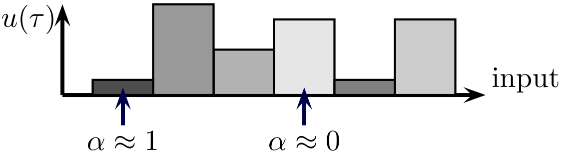
1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P with $u(P + \tau_i) \leq 1 - \beta(P + \tau_i)$

Our algorithm

First Fit Matching Periods (FFMP):

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P with $u(P + \tau_i) \leq 1 - \beta(P + \tau_i)$

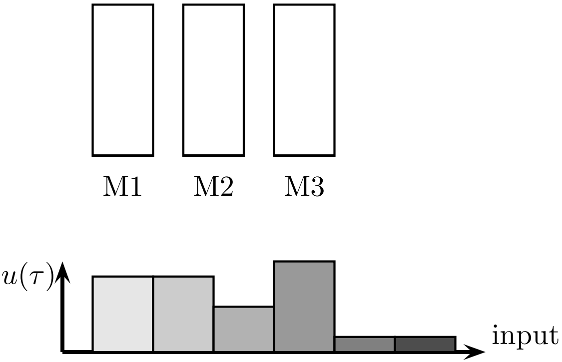




Our algorithm

First Fit Matching Periods (FFMP):

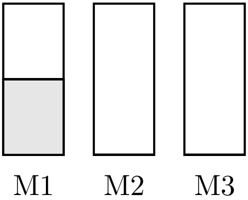
1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P with $u(P + \tau_i) \leq 1 - \beta(P + \tau_i)$

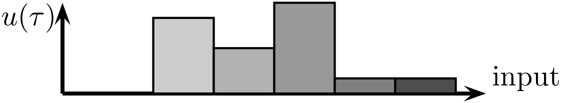


Our algorithm

First Fit Matching Periods (FFMP):

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P with $u(P + \tau_i) \leq 1 - \beta(P + \tau_i)$

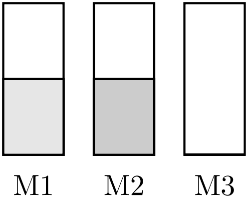


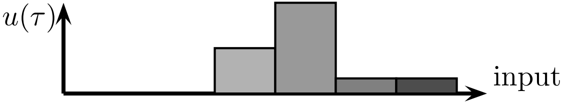


Our algorithm

First Fit Matching Periods (FFMP):

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P with $u(P + \tau_i) \leq 1 - \beta(P + \tau_i)$

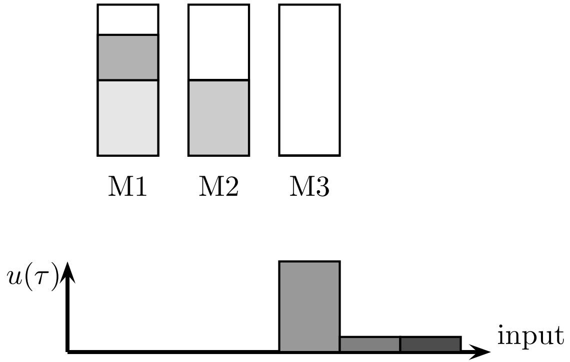




Our algorithm

First Fit Matching Periods (FFMP):

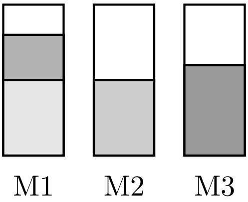
1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P with $u(P + \tau_i) \leq 1 - \beta(P + \tau_i)$

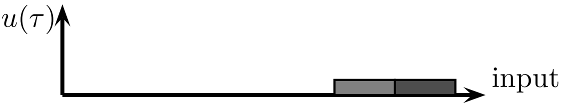


Our algorithm

First Fit Matching Periods (FFMP):

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P with $u(P + \tau_i) \leq 1 - \beta(P + \tau_i)$

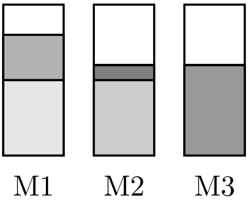




Our algorithm

First Fit Matching Periods (FFMP):

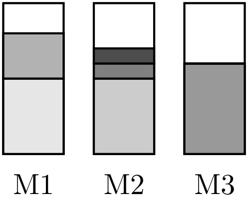
1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P with $u(P + \tau_i) \leq 1 - \beta(P + \tau_i)$



Our algorithm

First Fit Matching Periods (FFMP):

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
    - Assign τ_i to first machine P with $u(P + \tau_i) \leq 1 - \beta(P + \tau_i)$



Our algorithm

First Fit Matching Periods (FFMP):

1. **Sort:** $0 \le \alpha(\tau_1) \le \alpha(\tau_2) \le \ldots \le \alpha(\tau_n) < 1$
2. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P with $u(P + \tau_i) \le 1 - \beta(P + \tau_i)$

Running time: $\mathcal{O}(n \log n)$

Theorem

For n tasks with $u(\tau_i) \in [0, 1]$ uniformly at random

$$E[\textit{waste of FFMP}] \le \mathcal{O}(n^{3/4} (\log n)^{3/8})$$

(average utilization $\to$ 100%)

Bin Packing

Given: n items with **sizes** $a_i \in [0, 1]$
Find: Assignment to minimum number of **bins** of size ≤ 1

Bin Packing

Given: n items with **sizes** $a_i \in [0, 1]$
Find: Assignment to minimum number of **bins** of size ≤ 1

(Restricted) First Fit:

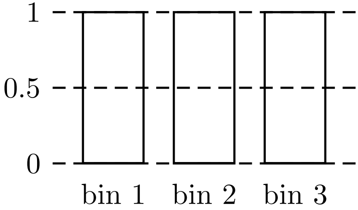
- FOR $i = 1, \ldots, n$ DO: Assign i to first bin B with
  - B is empty **or**
  - B contains exactly 1 item of size $\geq 1/2$ and $\text{size}(B) + a_i \leq 1$

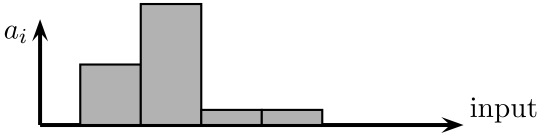
Bin Packing

Given: n items with **sizes** $a_i \in [0, 1]$
Find: Assignment to minimum number of **bins** of size ≤ 1

(Restricted) First Fit:

- FOR $i = 1, \ldots, n$ DO: Assign i to first bin B with
  - B is empty **or**
  - B contains exactly 1 item of size $\geq 1/2$ and $\text{size}(B) + a_i \leq 1$



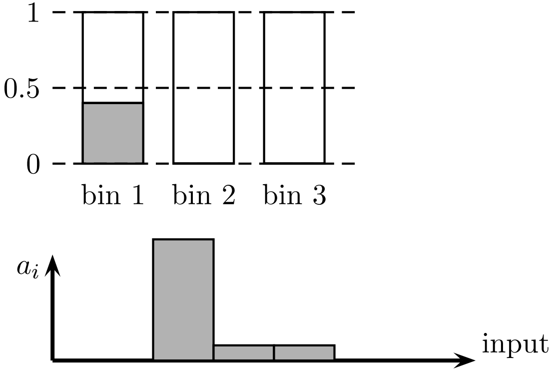


Bin Packing

Given: n items with **sizes** $a_i \in [0,1]$
Find: Assignment to minimum number of **bins** of size ≤ 1

(Restricted) First Fit:

- FOR $i = 1, \ldots, n$ DO: Assign i to first bin B with
  - B is empty **or**
  - B contains exactly 1 item of size $\geq 1/2$ and $\text{size}(B) + a_i \leq 1$

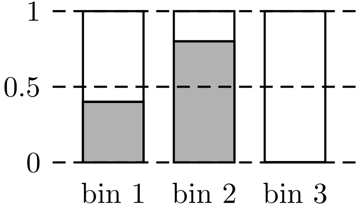


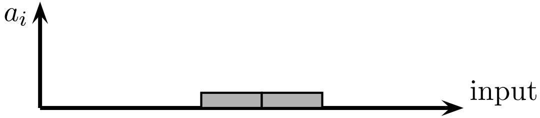
Bin Packing

Given: n items with **sizes** $a_i \in [0, 1]$
Find: Assignment to minimum number of **bins** of size ≤ 1

(Restricted) First Fit:

- FOR $i = 1, \ldots, n$ DO: Assign i to first bin B with
  - B is empty **or**
  - B contains exactly 1 item of size $\geq 1/2$ and $\text{size}(B) + a_i \leq 1$



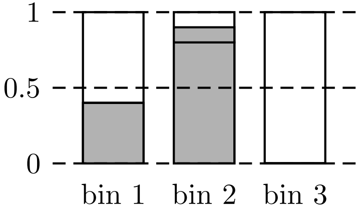


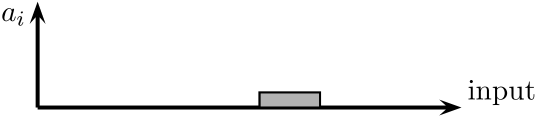
Bin Packing

Given: n items with **sizes** $a_i \in [0, 1]$
Find: Assignment to minimum number of **bins** of size ≤ 1

(Restricted) First Fit:

- FOR $i = 1, \ldots, n$ DO: Assign i to first bin B with
  - B is empty **or**
  - B contains exactly 1 item of size $\geq 1/2$ and $\text{size}(B) + a_i \leq 1$



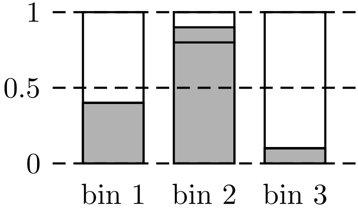


Bin Packing

Given: n items with **sizes** $a_i \in [0,1]$
Find: Assignment to minimum number of **bins** of size ≤ 1

(Restricted) First Fit:

- FOR $i = 1, \ldots, n$ DO: Assign i to first bin B with
  - B is empty **or**
  - B contains exactly 1 item of size $\geq 1/2$ and $\text{size}(B) + a_i \leq 1$



Bin Packing

Given: n items with **sizes** $a_i \in [0, 1]$
Find: Assignment to minimum number of **bins** of size ≤ 1

(Restricted) First Fit:

- FOR $i = 1, \ldots, n$ DO: Assign i to first bin B with
  - B is empty **or**
  - B contains exactly 1 item of size $\geq 1/2$ and $\text{size}(B) + a_i \leq 1$

Theorem (Shor '84)

Given n items with $a_i \in [0, 1]$ uniformly at random

$$E[\textit{waste of First Fit}] \leq \mathcal{O}(n^{2/3}\sqrt{\log n}).$$

Bin Packing

Given: n items with **sizes** $a_i \in [0, 1]$
Find: Assignment to minimum number of **bins** of size ≤ 1

(Restricted) First Fit:

- FOR $i = 1, \ldots, n$ DO: Assign i to first bin B with
  - B is empty **or**
  - B contains exactly 1 item of size $\geq 1/2$ and $\text{size}(B) + a_i \leq 1$

Theorem (Shor '84)

Given n items with $a_i \in [0, 1]$ uniformly at random

$$E[\textit{waste of First Fit}] \leq \mathcal{O}(n^{2/3}\sqrt{\log n}).$$

Theorem (Shor '84)

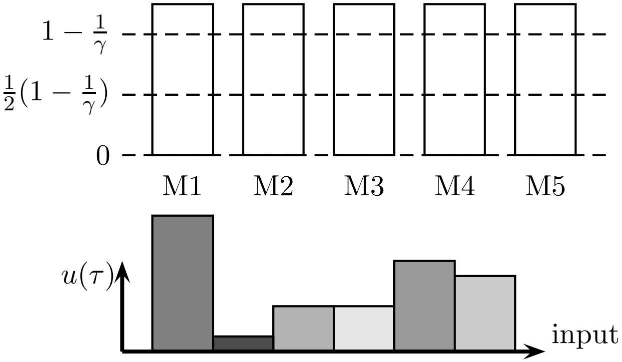
First Fit is **monotone**.

An auxiliary algorithm: FFMP*

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \leq \alpha(\tau_i) < \frac{j}{\gamma}\} \rightarrow$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \geq \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \leq 1 - \frac{1}{\gamma}$

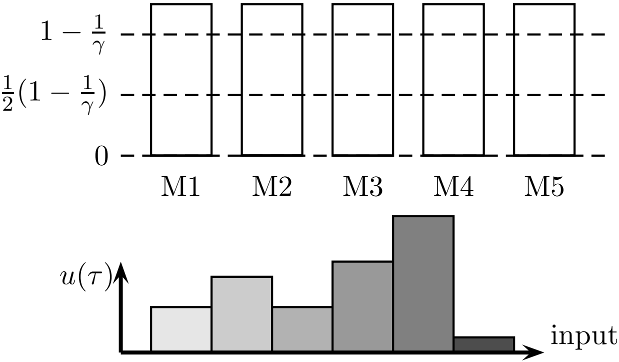
An auxiliary algorithm: FFMP*

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \leq \alpha(\tau_i) < \frac{j}{\gamma}\} \rightarrow$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \geq \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \leq 1 - \frac{1}{\gamma}$



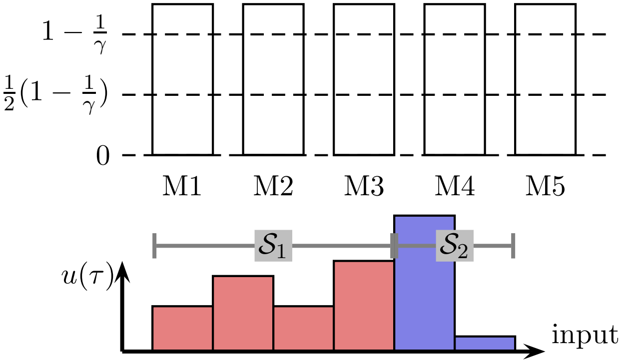
An auxiliary algorithm: FFMP*

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \leq \alpha(\tau_i) < \frac{j}{\gamma}\} \rightarrow$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \geq \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \leq 1 - \frac{1}{\gamma}$



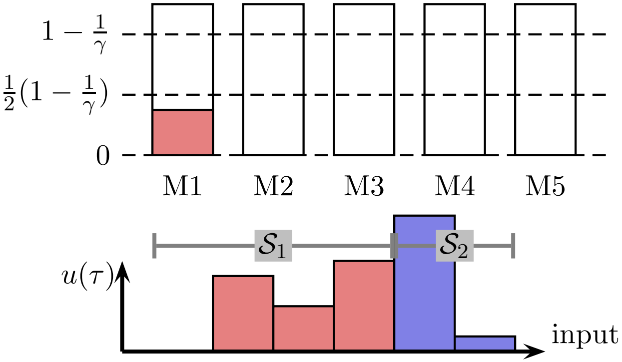
An auxiliary algorithm: FFMP*

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \leq \alpha(\tau_i) < \frac{j}{\gamma}\} \rightarrow$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \geq \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \leq 1 - \frac{1}{\gamma}$



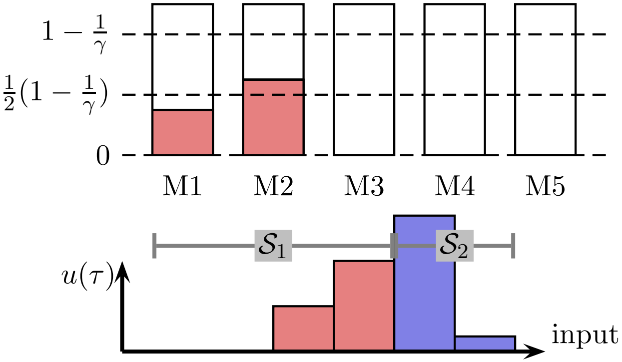
An auxiliary algorithm: FFMP*

1. **Sort:** $0 \le \alpha(\tau_1) \le \alpha(\tau_2) \le \ldots \le \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \le \alpha(\tau_i) < \frac{j}{\gamma}\} \to$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \ge \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \le 1 - \frac{1}{\gamma}$



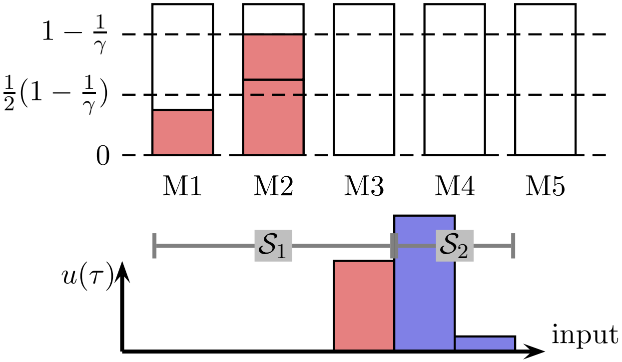
An auxiliary algorithm: FFMP*

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \leq \alpha(\tau_i) < \frac{j}{\gamma}\} \rightarrow$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \geq \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \leq 1 - \frac{1}{\gamma}$



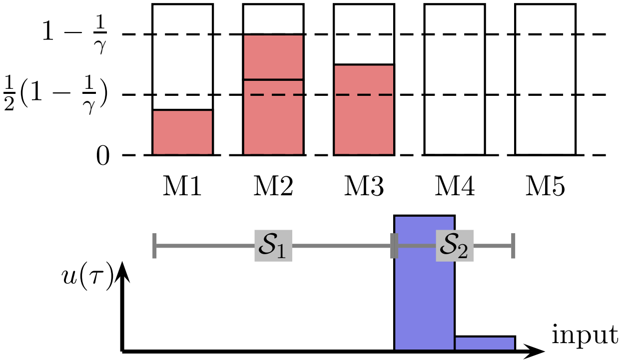
An auxiliary algorithm: FFMP*

1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \leq \alpha(\tau_i) < \frac{j}{\gamma}\} \rightarrow$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \geq \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \leq 1 - \frac{1}{\gamma}$



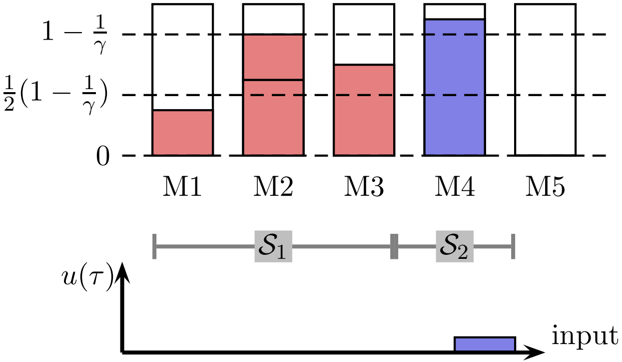
An auxiliary algorithm: FFMP*

1. **Sort:** $0 \le \alpha(\tau_1) \le \alpha(\tau_2) \le \ldots \le \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \le \alpha(\tau_i) < \frac{j}{\gamma}\} \rightarrow$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \ge \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \le 1 - \frac{1}{\gamma}$



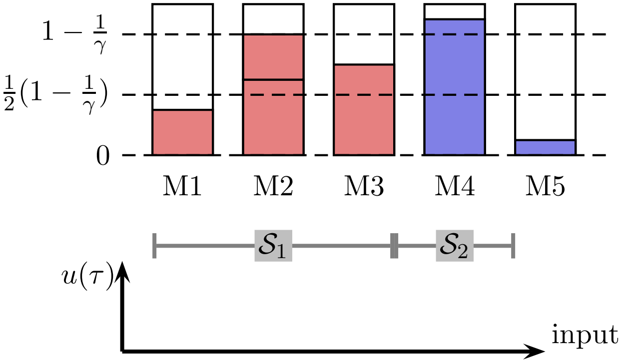
An auxiliary algorithm: FFMP*

1. **Sort:** $0 \le \alpha(\tau_1) \le \alpha(\tau_2) \le \ldots \le \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \le \alpha(\tau_i) < \frac{j}{\gamma}\} \rightarrow$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \ge \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \le 1 - \frac{1}{\gamma}$

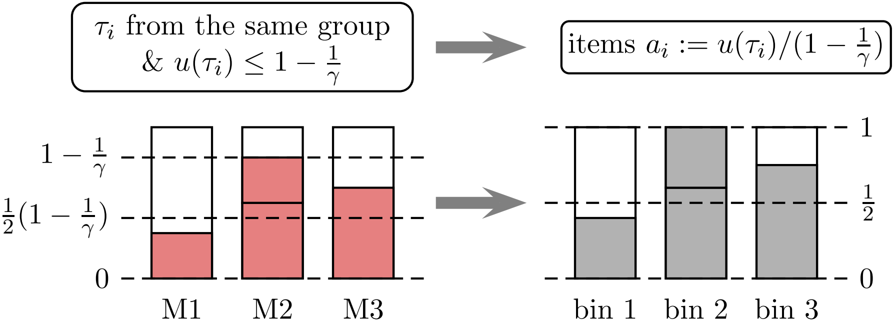


An auxiliary algorithm: FFMP*

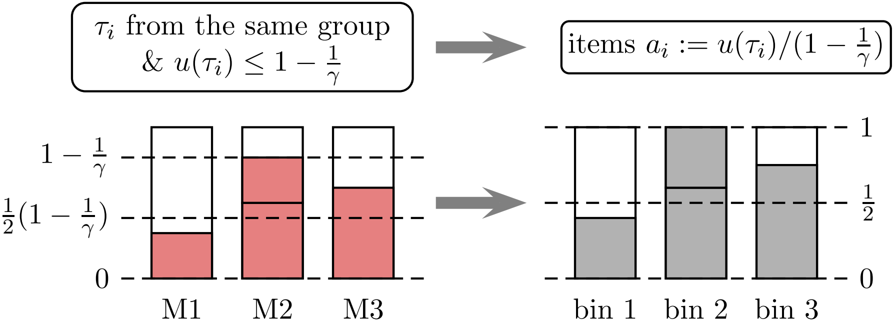
1. **Sort:** $0 \leq \alpha(\tau_1) \leq \alpha(\tau_2) \leq \ldots \leq \alpha(\tau_n) < 1$
2. **Partition:** $\mathcal{S}_j = \{\tau_i \mid \frac{j-1}{\gamma} \leq \alpha(\tau_i) < \frac{j}{\gamma}\} \rightarrow$ **groups**
3. **First Fit:** FOR $i = 1, \ldots, n$ DO
   - Assign τ_i to first machine P that's empty **or**
     - P contains exactly one task τ_j **and**
     - τ_j is from the same group & $u(\tau_j) \geq \frac{1}{2}(1 - \frac{1}{\gamma})$ **and**
     - $u(P + \tau_i) \leq 1 - \frac{1}{\gamma}$



FFMP* vs. First Fit

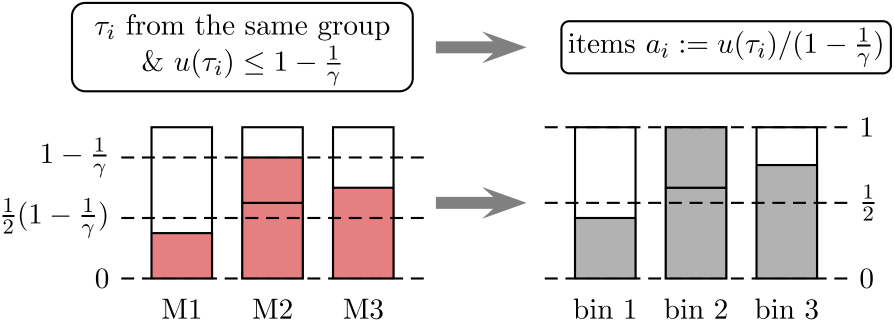


FFMP* vs. First Fit



- FFMP* = First Fit

FFMP* vs. First Fit



- FFMP* = First Fit

Corollary

FFMP is monotone.*

Consequences

Lemma

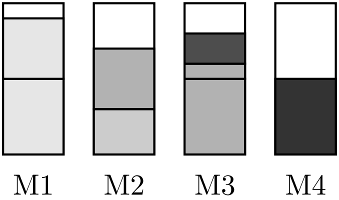
For any task set $\mathcal{S}$: $FFMP(\mathcal{S}) \leq FFMP^*(\mathcal{S})$

Consequences

Lemma

For any task set $\mathcal{S}$: $FFMP(\mathcal{S}) \leq FFMP^*(\mathcal{S})$

- Consider schedule of $\mathcal{S}$ w.r.t. FFMP.

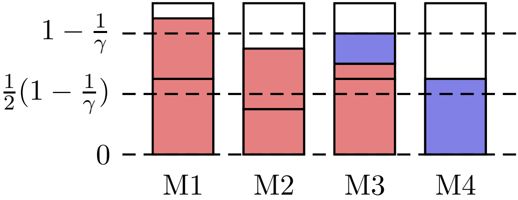


Consequences

Lemma

For any task set $\mathcal{S}$: $FFMP(\mathcal{S}) \leq FFMP^*(\mathcal{S})$

- Consider schedule of $\mathcal{S}$ w.r.t. FFMP.
- Delete tasks that FFMP* would have scheduled differently

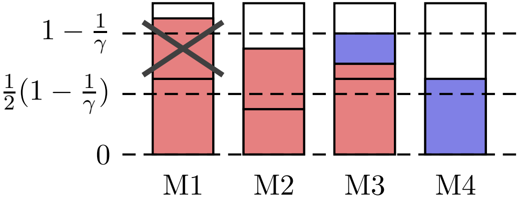


Consequences

Lemma

For any task set $\mathcal{S}$: $FFMP(\mathcal{S}) \leq FFMP^*(\mathcal{S})$

- Consider schedule of $\mathcal{S}$ w.r.t. FFMP.
- Delete tasks that FFMP* would have scheduled differently

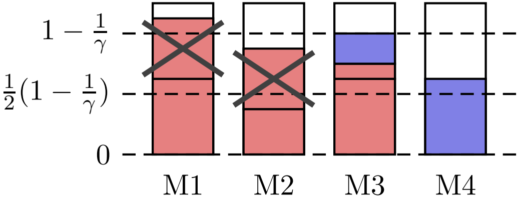


Consequences

Lemma

For any task set $\mathcal{S}$: $FFMP(\mathcal{S}) \leq FFMP^*(\mathcal{S})$

- Consider schedule of $\mathcal{S}$ w.r.t. FFMP.
- Delete tasks that FFMP* would have scheduled differently

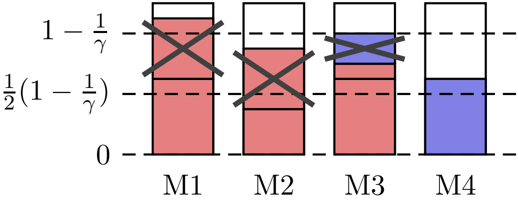


Consequences

Lemma

For any task set $\mathcal{S}$: $FFMP(\mathcal{S}) \leq FFMP^*(\mathcal{S})$

- Consider schedule of $\mathcal{S}$ w.r.t. FFMP.
- Delete tasks that FFMP* would have scheduled differently

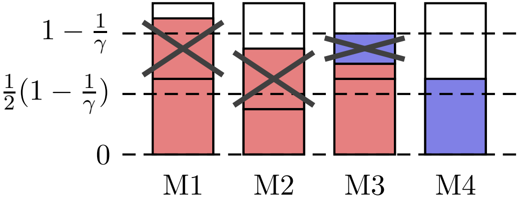


Consequences

Lemma

For any task set $\mathcal{S}$: $FFMP(\mathcal{S}) \leq FFMP^*(\mathcal{S})$

- Consider schedule of $\mathcal{S}$ w.r.t. FFMP.
- Delete tasks that FFMP* would have scheduled differently $\rightarrow \mathcal{S}' \subseteq \mathcal{S}$



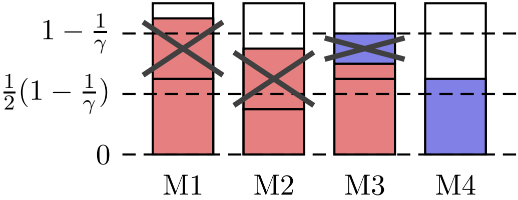
Consequences

Lemma

For any task set $\mathcal{S}$: $FFMP(\mathcal{S}) \leq FFMP^*(\mathcal{S})$

- Consider schedule of $\mathcal{S}$ w.r.t. FFMP.
- Delete tasks that FFMP* would have scheduled differently $\rightarrow \mathcal{S}' \subseteq \mathcal{S}$

$$\text{FFMP}(\mathcal{S}) = \text{FFMP}^*(\mathcal{S}')$$



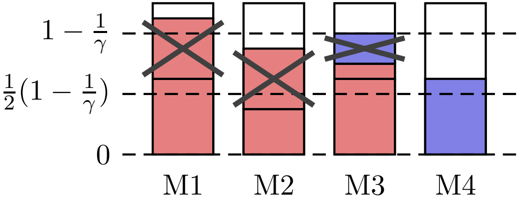
Consequences

Lemma

For any task set $\mathcal{S}$: FFMP$(\mathcal{S}) \leq$ FFMP$^*(\mathcal{S})$

- Consider schedule of $\mathcal{S}$ w.r.t. FFMP.
- Delete tasks that FFMP* would have scheduled differently $\rightarrow \mathcal{S}' \subseteq \mathcal{S}$

$$\text{FFMP}(\mathcal{S}) = \text{FFMP}^*(\mathcal{S}') \overset{\text{monotonicity}}{\leq} \text{FFMP}^*(\mathcal{S})$$



□

Main result

Theorem

$E[waste\ of\ FFMP^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

Main result

Theorem

$E[waste\ of\ FFMP^{*}] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit

Main result

Theorem

$E[waste\ of\ FFMP^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks

Main result

Theorem

$E[waste\ of\ FFMP^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$

Main result

Theorem

$E[\textit{waste of FFMP}^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$
- $\forall \tau_i \in \mathcal{S}'_j : u(\tau_i) \in [0, 1 - \frac{1}{\gamma}]$ uniform $\Rightarrow a_i \in [0, 1]$ uniform

Main result

Theorem

$E[\textit{waste of FFMP}^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$
- $\forall \tau_i \in \mathcal{S}'_j : u(\tau_i) \in [0, 1 - \frac{1}{\gamma}]$ uniform $\Rightarrow a_i \in [0, 1]$ uniform

$E[\text{waste of FFMP}^*(\mathcal{S}_j)]$

Main result

Theorem

$E[\textit{waste of FFMP}^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$
- $\forall \tau_i \in \mathcal{S}'_j : u(\tau_i) \in [0, 1 - \frac{1}{\gamma}]$ uniform $\Rightarrow a_i \in [0,1]$ uniform

$$E[\text{waste of FFMP}^*(\mathcal{S}_j)] \quad \leq \quad |\mathcal{S}_j| \cdot \frac{1}{\gamma} + E[\text{waste of FirstFit}(I_j)]$$

Main result

Theorem

$E[\textit{waste of FFMP}^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$
- $\forall \tau_i \in \mathcal{S}'_j : u(\tau_i) \in [0, 1 - \frac{1}{\gamma}]$ uniform $\Rightarrow a_i \in [0,1]$ uniform

$$\begin{aligned} E[\text{waste of FFMP}^*(\mathcal{S}_j)] &\leq |\mathcal{S}_j| \cdot \frac{1}{\gamma} + E[\text{waste of FirstFit}(I_j)] \\ &\leq \frac{|\mathcal{S}_j|}{\gamma} + f(|I_j|) \end{aligned}$$

Main result

Theorem

$E[\textit{waste of FFMP}^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$
- $\forall \tau_i \in \mathcal{S}'_j : u(\tau_i) \in [0, 1 - \frac{1}{\gamma}]$ uniform $\Rightarrow a_i \in [0,1]$ uniform

$$\begin{aligned} E[\text{waste of FFMP}^*(\mathcal{S}_j)] &\leq |\mathcal{S}_j| \cdot \frac{1}{\gamma} + E[\text{waste of FirstFit}(I_j)] \\ &\leq \frac{|\mathcal{S}_j|}{\gamma} + f(|I_j|) \leq \frac{|\mathcal{S}_j|}{\gamma} + f(|\mathcal{S}_j|) \end{aligned}$$

Main result

Theorem

$E[\textit{waste of FFMP}^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$
- $\forall \tau_i \in \mathcal{S}'_j : u(\tau_i) \in [0, 1 - \frac{1}{\gamma}]$ uniform $\Rightarrow a_i \in [0,1]$ uniform

$$\begin{aligned} E[\text{waste of FFMP}^*(\mathcal{S}_j)] &\leq |\mathcal{S}_j| \cdot \frac{1}{\gamma} + E[\text{waste of FirstFit}(I_j)] \\ &\leq \frac{|\mathcal{S}_j|}{\gamma} + f(|I_j|) \leq \frac{|\mathcal{S}_j|}{\gamma} + f(|\mathcal{S}_j|) \end{aligned}$$

$E[\text{waste of FFMP}^*(\mathcal{S})]$

Main result

Theorem

$E[\textit{waste of FFMP}^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$
- $\forall \tau_i \in \mathcal{S}'_j : u(\tau_i) \in [0, 1 - \frac{1}{\gamma}]$ uniform $\Rightarrow a_i \in [0,1]$ uniform

$$E[\text{waste of FFMP}^*(\mathcal{S}_j)] \leq |\mathcal{S}_j| \cdot \frac{1}{\gamma} + E[\text{waste of FirstFit}(I_j)]$$

$$\leq \frac{|\mathcal{S}_j|}{\gamma} + f(|I_j|) \leq \frac{|\mathcal{S}_j|}{\gamma} + f(|\mathcal{S}_j|)$$

$$E[\text{waste of FFMP}^*(\mathcal{S})] \leq \sum_{j=1}^{\gamma} \left(\frac{|\mathcal{S}_j|}{\gamma} + f(|\mathcal{S}_j|) \right)$$

Main result

Theorem

$E[\textit{waste of FFMP}^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$
- $\forall \tau_i \in \mathcal{S}'_j : u(\tau_i) \in [0, 1 - \frac{1}{\gamma}]$ uniform $\Rightarrow a_i \in [0,1]$ uniform

$$\begin{aligned} E[\text{waste of FFMP}^*(\mathcal{S}_j)] &\leq |\mathcal{S}_j| \cdot \frac{1}{\gamma} + E[\text{waste of FirstFit}(I_j)] \\ &\leq \frac{|\mathcal{S}_j|}{\gamma} + f(|I_j|) \leq \frac{|\mathcal{S}_j|}{\gamma} + f(|\mathcal{S}_j|) \end{aligned}$$

$$E[\text{waste of FFMP}^*(\mathcal{S})] \leq \sum_{j=1}^{\gamma} \left(\frac{|\mathcal{S}_j|}{\gamma} + f(|\mathcal{S}_j|) \right)$$

$$\stackrel{\text{Jensen}}{\leq} \frac{n}{\gamma} + \gamma \cdot f(n/\gamma)$$

Main result

Theorem

$E[\textit{waste of FFMP}^*] \leq \mathcal{O}(n^{3/4}(\log n)^{3/8})$

- $f(m) := \mathcal{O}(m^{2/3}\sqrt{\log m})$ expected waste of First Fit
- $\mathcal{S}'_j := \{\tau_i \in \mathcal{S}_j \mid u(\tau_i) \leq 1 - \frac{1}{\gamma}\}$ not too big tasks
- $I_j := \{a_i := u(\tau_i)/(1 - \frac{1}{\gamma}) \mid \tau_i \in \mathcal{S}'_j\}$
- $\forall \tau_i \in \mathcal{S}'_j : u(\tau_i) \in [0, 1 - \frac{1}{\gamma}]$ uniform $\Rightarrow a_i \in [0,1]$ uniform

$$E[\text{waste of FFMP}^*(\mathcal{S}_j)] \leq |\mathcal{S}_j| \cdot \frac{1}{\gamma} + E[\text{waste of FirstFit}(I_j)]$$

$$\leq \frac{|\mathcal{S}_j|}{\gamma} + f(|I_j|) \leq \frac{|\mathcal{S}_j|}{\gamma} + f(|\mathcal{S}_j|)$$

$$E[\text{waste of FFMP}^*(\mathcal{S})] \leq \sum_{j=1}^{\gamma} \left(\frac{|\mathcal{S}_j|}{\gamma} + f(|\mathcal{S}_j|) \right)$$

$$\overset{\text{Jensen}}{\leq} \frac{n}{\gamma} + \gamma \cdot f(n/\gamma) \overset{\gamma := n^{1/4}/(\log n)^{3/8}}{\leq} \mathcal{O}(n^{3/4}(\log n)^{3/8}) \quad \square$$

Open problems

- **Problem:** Is there an asymptotic PTAS for Multiprocessor Real-time scheduling?

Open problems

- **Problem:** Is there an asymptotic PTAS for Multiprocessor Real-time scheduling?

Thanks for your attention