An asymptotic 3/2-approximation algorithm for static-priority multiprocessor scheduling of implicit deadline tasks

Andreas Karrenbauer & Thomas Rothvoß

Institute of Mathematics
EPFL, Lausanne

19.02.09
Real-time Scheduling

**Given:** synchronous tasks $\tau_1, \ldots, \tau_n$ where task $\tau_i$
- is periodic with period $p(\tau_i)$
- has running time $c(\tau_i)$
- has implicit deadline

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

**Scheduling policy:**
- multiprocessor
- fixed priority
- pre-emptive
Real-time Scheduling

**Given:** synchronous tasks $\tau_1, \ldots, \tau_n$ where task $\tau_i$
  - is periodic with period $p(\tau_i)$
  - has running time $c(\tau_i)$
  - has implicit deadline

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

**Scheduling policy:**
  - multiprocessor
  - fixed priority
  - pre-emptive

**Definition**

$$u(\tau) = \frac{c(\tau)}{p(\tau)} = \text{utilization of task } \tau$$
Example

\[ c(\tau_1) = 1 \]
\[ p(\tau_1) = 2 \]

\[ c(\tau_2) = 2 \]
\[ p(\tau_2) = 5 \]
Theorem (Liu, Layland ’73)

Optimal priorities: $\frac{1}{p(\tau_i)}$ for $\tau_i$ (Rate-monotonic Schedule)
Theorem (Liu, Layland ’73)

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

\[
\begin{align*}
  c(\tau_1) &= 1 \\
  p(\tau_1) &= 2 \\
  c(\tau_2) &= 2 \\
  p(\tau_2) &= 5
\end{align*}
\]
Theorem (Liu, Layland ’73)

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

Theorem (Liu, Layland ’73)

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

\[
c(\tau_1) = 1 \\
p(\tau_1) = 2
\]

\[
c(\tau_2) = 2 \\
p(\tau_2) = 5
\]

\[u(S) = \sum_{\tau \in S} u(\tau) \leq \ln(2) \approx 0.69 \Rightarrow RM\text{-}schedule \text{ feasible}\]
Response times

Definition

Response time $r(\tau_i)$ of $\tau_i$ := longest time that an instance of task $\tau_i$ waits for accomplishment

Theorem (Lehoczky et al. ’89)

If $p(\tau_1) \leq \ldots \leq p(\tau_n)$ then $r(\tau_i)$ is the smallest value s.t.

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

Tasks are feasible $\Leftrightarrow \forall i : r(\tau_i) \leq p(\tau_i)$
Response times

**Definition**

Response time \( r(\tau_i) \) of \( \tau_i \) := longest time that an instance of task \( \tau_i \) waits for accomplishment

**Theorem (Lehoczky et al. ’89)**

If \( p(\tau_1) \leq \ldots \leq p(\tau_n) \) then \( r(\tau_i) \) is the smallest value s.t.

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

Tasks are feasible \( \iff \forall i : r(\tau_i) \leq p(\tau_i) \)

**Theorem (Eisenbrand & R. - RTSS’08)**

Unless \( \text{NP} = \text{P} \) response times \( r(\tau_i) \) cannot be approximated within any constant.
A sufficient feasibility criterium

Observation:

If periods multiples of each other:
Tasks $\mathcal{S}$ RM-schedulable $\iff u(\mathcal{S}) \leq 1$ ($\rightarrow$ Bin Packing)
A sufficient feasibility criterium

Observation:
If periods multiples of each other:
Tasks $\mathcal{S}$ RM-schedulable $\iff u(\mathcal{S}) \leq 1$ ($\rightarrow$ Bin Packing)

Lemma (Burchard et al ’95)
For tasks $\mathcal{S} = \{\tau_1, \ldots, \tau_n\}$ define

$$S(\tau_i) = \log p(\tau_i) - \lfloor \log p(\tau_i) \rfloor$$

and

$$\beta(\mathcal{S}) = \max_{\tau \in \mathcal{S}} S(\tau) - \min_{\tau \in \mathcal{S}} S(\tau)$$

Then the tasks can be RM-scheduled if

$$u(\mathcal{S}) \leq 1 - \beta(\mathcal{S}) \cdot \ln(2).$$
A simple First Fit algorithm

First Fit:

1. Sort tasks s.t. $0 \leq S(\tau_1) \leq \ldots \leq S(\tau_n) < 1$
2. Distribute tasks via First Fit (using $u(S) \leq 1 - \beta(S) \cdot \ln(2)$ feasibility criterium)

Lemma

Given periodic tasks $S = \{\tau_1, \ldots, \tau_n\}$ and $\varepsilon > 0$.

(1) Let $u(\tau_i) \leq \alpha$ then $\#\text{proc.} \leq \frac{1}{1-\alpha} u(S) + 3$.
(2) Let $u(\tau_i) \leq \frac{1}{2} - \varepsilon$ then $\#\text{proc.} \leq \frac{n}{2} + \frac{1}{\varepsilon}$
(3) First Fit gives an asymptotic 2-approx.
Known results for multiprocessor case

- An asymptotic 7/4-approximation algo [Burchard ’95]
Known results for multiprocessor case

- An asymptotic \(\frac{7}{4}\)-approximation algo [Burchard ’95]
- An asymptotic PTAS under resource augmentation [Eisenbrand & R. - ICALP’08]
Known results for multiprocessor case

- An asymptotic $7/4$-approximation algo [Burchard ’95]
- An asymptotic PTAS under resource augmentation [Eisenbrand & R. - ICALP’08]
- There is no asymptotic FPTAS [Eisenbrand & R. - ICALP’08]
Known results for multiprocessor case

- An asymptotic 7/4-approximation algo [Burchard ’95]
- An asymptotic PTAS under resource augmentation [Eisenbrand & R. - ICALP’08]
- There is no asymptotic FPTAS [Eisenbrand & R. - ICALP’08]

**Theorem (Karrenbauer & R. ’09)**

*There is an asymptotic 3/2-approximation algorithm for Multiprocessor Real-time Scheduling with running time $O(n^3)$.*
Using the power of matchings

\[ \tau_1 \quad \tau_2 \]

\[ \tau_3 \quad \tau_4 \]
Using the power of matchings

\[ \tau_1 \leftrightarrow \{\tau_1, \tau_2\} \text{ feasible} \]
Using the power of matchings
Using the power of matchings

\[ w(\tau_1) \xrightarrow{1} \tau_1 \xrightarrow{1} \tau_2 \xrightarrow{1} \tau_3 \xrightarrow{1} \tau_4 \xrightarrow{1} w(\tau_2) \]

\[ w(\tau) = \lim_{k \to \infty} \frac{\text{#proc. needed to schedule } k \text{ copies of } \tau}{k} \]
Using the power of matchings

\[ w(\tau) = \lim_{k \to \infty} \frac{\# \text{proc. needed to schedule } k \text{ copies of } \tau}{k} \]
Using the power of matchings

\[
1 - w(\tau_1) - w(\tau_2)
\]

\[
1 - w(\tau_1) - w(\tau_3)
\]

\[
1 - w(\tau_3) - w(\tau_4)
\]

\[
w(\tau) = \lim_{k \to \infty} \frac{\text{#proc. needed to schedule } k \text{ copies of } \tau}{k}
\]
Using the power of matchings

\[ w(\tau) = \lim_{k \to \infty} \frac{\# \text{proc. needed to schedule } k \text{ copies of } \tau}{k} \]

1 - w(\tau_1) - w(\tau_3)

1 - w(\tau_3) - w(\tau_4)

1 - w(\tau_1) - w(\tau_2)
The vertex costs

\[ w(\tau) = \lim_{k \to \infty} \frac{\# \text{proc. needed to schedule } k \text{ copies of } \tau}{k} \]

\[
= \begin{cases} 
  u(\tau) \cdot \frac{1}{1-u(\tau)} & u(\tau) \leq \frac{1}{3} \\
  \frac{1}{2} & \frac{1}{3} < u(\tau) \leq \frac{1}{2} - \frac{1}{12k} \\
  1 & u(\tau) > \frac{1}{2} - \frac{1}{12k} 
\end{cases}
\]
The algorithm

1. Construct $G = (S, E)$ with edges $(\tau_1, \tau_2) \in E \iff \{\tau_1, \tau_2\}$ RM-schedulable. Choose edge costs $c_e = 1$ and vertex costs

$$w(\tau) = \begin{cases} 
    u(\tau) \cdot \frac{1}{1-u(\tau)} & u(\tau) \leq \frac{1}{3} \\
    \frac{1}{2} & \frac{1}{3} < u(\tau) \leq \frac{1}{2} - \frac{1}{12k} \\
    1 & u(\tau) > \frac{1}{2} - \frac{1}{12k}
\end{cases}$$

2. Solve

$$\min \{ \sum_{e \in M} c(e) + \sum_{\tau \in S \setminus V(M)} w(\tau) \mid M \text{ is matching} \}$$

3. For all $\{\tau_1, \tau_2\} \in M$ create a processor with $\{\tau_1, \tau_2\}$

4. Define

- $S_i = \{\tau \in S \setminus V(M) \mid \frac{1}{3} \cdot \frac{i-1}{k} \leq u(\tau) < \frac{1}{3} \cdot \frac{i}{k}\} \forall i = 1, \ldots, k$
- $S_{k+1} = \{\tau \in S \setminus V(M) \mid \frac{1}{3} \leq u(\tau) \leq \frac{1}{2} - \frac{1}{12k}\}$
- $S_{k+2} = \{\tau \in S \setminus V(M) \mid u(\tau) > \frac{1}{2} - \frac{1}{12k}\}$

5. Distribute $S_1, \ldots, S_{k+2}$ via First Fit.
The analysis

Lemma

The algorithm yields a solution of cost \((\frac{3}{2} + o(1)) \cdot OPT\).
The analysis

Lemma

The algorithm yields a solution of cost \((\frac{3}{2} + o(1)) \cdot OPT\).

- Matching \(M\) of cost \(\alpha\) \(\Rightarrow\) solution with \((1 + o(1)) \cdot \alpha\) processors
The analysis

Lemma

The algorithm yields a solution of cost \((\frac{3}{2} + o(1)) \cdot OPT\).

- Matching \(M\) of cost \(\alpha\) \(\Rightarrow\) solution with \((1 + o(1)) \cdot \alpha\) processors
  - \(\{\tau_1, \tau_2\} \in M \Rightarrow\) schedule \(\{\tau_1, \tau_2\}\) with 1 proc.
The analysis

**Lemma**

*The algorithm yields a solution of cost* \((\frac{3}{2} + o(1)) \cdot OPT.*

- Matching \(M\) of cost \(\alpha\) \(\Rightarrow\) solution with \((1 + o(1)) \cdot \alpha\) processors
  - \(\{\tau_1, \tau_2\} \in M \Rightarrow\) schedule \(\{\tau_1, \tau_2\}\) with 1 proc.
  - \(\tau\) not covered by \(M\) \(\Rightarrow\) need \(w(\tau) \in [0, 1]\) proc. on average.
The analysis

Lemma

The algorithm yields a solution of cost \((\frac{3}{2} + o(1)) \cdot OPT\).

- Matching \(M\) of cost \(\alpha \Rightarrow\) solution with \((1 + o(1)) \cdot \alpha\) processors
  - \(\{\tau_1, \tau_2\} \in M \Rightarrow\) schedule \(\{\tau_1, \tau_2\}\) with 1 proc.
  - \(\tau\) not covered by \(M\) \(\Rightarrow\) need \(w(\tau) \in [0, 1]\) proc. on average.
- To show: \(\exists\) matching solution of cost \((\frac{3}{2} + o(1))OPT\)
The analysis

**Lemma**

The algorithm yields a solution of cost \((\frac{3}{2} + o(1)) \cdot OPT\).

- Matching \(M\) of cost \(\alpha\) \(\Rightarrow\) solution with \((1 + o(1)) \cdot \alpha\) processors
  - \(\{\tau_1, \tau_2\} \in M \Rightarrow \) schedule \(\{\tau_1, \tau_2\}\) with 1 proc.
  - \(\tau\) not covered by \(M\) \(\Rightarrow\) need \(w(\tau) \in [0, 1]\) proc. on average.
- To show: \(\exists\) matching solution of cost \((\frac{3}{2} + o(1))OPT\)
- Let \(P_1, \ldots, P_m\) optimum solution. Each processor \(P_i\) contributes at most \((\frac{3}{2} + o(1))\) to the matching
The analysis

**Lemma**

The algorithm yields a solution of cost \((\frac{3}{2} + o(1)) \cdot OPT\).

- Matching \(M\) of cost \(\alpha \Rightarrow \text{solution with } (1 + o(1)) \cdot \alpha \text{ processors}
  - \(\{\tau_1, \tau_2\} \in M \Rightarrow \text{schedule } \{\tau_1, \tau_2\} \text{ with 1 proc.}
  - \tau \text{ not covered by } M \Rightarrow \text{need } w(\tau) \in [0, 1] \text{ proc. on average.}
- To show: \(\exists\) matching solution of cost \((\frac{3}{2} + o(1))OPT\)
- Let \(P_1, \ldots, P_m\) optimum solution. Each processor \(P_i\) contributes at most \((\frac{3}{2} + o(1))\) to the matching
- Let \(P_i = \{\tau_1, \ldots, \tau_q\} \text{ with } u(\tau_1) \geq \ldots \geq u(\tau_q)\).
Case split

- Case: $q = 1$. Do not cover task. Contribution: $w(\tau_1) \leq 1$
Case split

- **Case:** $q = 1$. Do not cover task. Contribution: $w(\tau_1) \leq 1$
- **Case:** $u(\{\tau_1, \tau_2\}) \geq 2/3$. Add $\{\tau_1, \tau_2\}$ to matching. Contr.:

\[
1 + \sum_{j=3}^{q} w(\tau_j) = 1 + \sum_{j=3}^{q} u(\tau_j) \cdot \left( \frac{\leq 3/2}{1 - u(\tau_j)} \right) \leq \frac{3}{2}.
\]
Case split

- **Case:** \( q = 1 \). Do not cover task. Contribution: \( w(\tau_1) \leq 1 \)
- **Case:** \( u(\{\tau_1, \tau_2\}) \geq 2/3 \). Add \( \{\tau_1, \tau_2\} \) to matching. Contr.: 
  
  \[
  1 + \sum_{j=3}^{q} w(\tau_j) = 1 + \sum_{j=3}^{q} u(\tau_j) \cdot \frac{1}{1 - u(\tau_j)} \leq \frac{3}{2}.
  \]

- **Case:** \( u(\tau_1) \geq \frac{1}{2} - \frac{1}{12k} \). Add \( \{\tau_1, \tau_2\} \) to \( M \). Contr.: \( \frac{3}{2} + o(1) \)
Case split

- **Case:** $q = 1$. Do not cover task. Contribution: $w(\tau_1) \leq 1$
- **Case:** $u(\{\tau_1, \tau_2\}) \geq 2/3$. Add $\{\tau_1, \tau_2\}$ to matching. Contr.:

\[
1 + \sum_{j=3}^{q} w(\tau_j) = 1 + \sum_{j=3}^{q} u(\tau_j) \cdot \frac{1}{1 - u(\tau_j)} \leq \frac{3}{2}.
\]

- **Case:** $u(\tau_1) \geq \frac{1}{2} - \frac{1}{12k}$. Add $\{\tau_1, \tau_2\}$ to $M$. Contr.: $\frac{3}{2} + o(1)$
- **Case:** $u(\tau_1) \leq 1/3$. Leave tasks uncovered. Contr.:

\[
\sum_{j=1}^{q} w(\tau_j) = \sum_{j=1}^{q} u(\tau_j) \cdot \frac{1}{1 - u(\tau_j)} \leq \frac{3}{2}.
\]
Case split

- **Case:** \( q = 1. \) Do not cover task. Contribution: \( w(\tau_1) \leq 1 \)
- **Case:** \( u(\{\tau_1, \tau_2\}) \geq 2/3. \) Add \( \{\tau_1, \tau_2\} \) to matching. Contr.

\[
1 + \sum_{j=3}^{q} w(\tau_j) = 1 + \sum_{j=3}^{q} u(\tau_j) \cdot \frac{1}{1 - u(\tau_j)} \leq \frac{3}{2}.
\]

- **Case:** \( u(\tau_1) \geq \frac{1}{2} - \frac{1}{12k}. \) Add \( \{\tau_1, \tau_2\} \) to \( M. \) Contr.: \( \frac{3}{2} + o(1) \)
- **Case:** \( u(\tau_1) \leq 1/3. \) Leave tasks uncovered. Contr.

\[
\sum_{j=1}^{q} w(\tau_j) = \sum_{j=1}^{q} u(\tau_j) \cdot \frac{1}{1 - u(\tau_j)} \leq \frac{3}{2}
\]

- **Case:** \( u(\tau_2) \leq \frac{1}{3} < u(\tau_1) \leq \frac{1}{2} - \frac{1}{12k}. \) Leave tasks uncovered. Contr.: \( \leq \frac{3}{2} \)
Average case behavior

Define the **waste** of a solution as the cumulated ratios of idle times \((= APX - u(S))\).

**Theorem (Karrenbauer & R. - ESA’09)**

Generate \(n\) tasks with
- **periods arbitrary**
- draw \(u(\tau_i) \in [0, 1]\) uniformly at random.

Then

\[
E[\text{waste of First Fit}] \leq O(n^{3/4}(\log n)^{3/8}).
\]

**Corollary (Karrenbauer & R. - ESA’09)**

Given \(n\) tasks \(S\) w.r.t. above probability distribution, the best solution in which at most 2 task share a processor, costs in expectation at most

\[
u(S) + O(n^{3/4}(\log n)^{3/8})
\]
Average case behavior (2)

Corollary (Karrenbauer & R. ’09)

Given $n$ tasks w.r.t. above probability distribution then

$$E[\text{waste of matching algo}] \leq O(n^{3/4} (\log n)^{3/8})$$
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