ECRTS 2012 in Pisa, Italy
11-13 July 2012

Hardness Results for
Static Priority Real-Time Scheduling

Martin Stigge

Uppsala University, Sweden

Joint work with Wang Yi
Problem Overview

Workload Model
- Task A
- Task B
- Task C

Scheduler Model
- EDF/Static Prio/...

This Work:
Test Complexity

Martin Stigge
Hardness of Static Priority Scheduling
Model/Design Choices

Task Models:
- Periodic (L&L)
- Generalized Multiframe (GMF)
- Digraph Real-Time (DRT)
- ...

Schedulers:
- Dynamic Priorities: EDF
- Static Priorities

Different combinations = Different complexity

(Here: Uniprocessor, preemption, precise tests)
Complexity of Schedulability Test

- Efficient schedulability tests possible?
  - ("Efficient" = "pseudo-polynomial")

<table>
<thead>
<tr>
<th></th>
<th>EDF</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td>L&amp;L</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>GMF</td>
<td>Yes</td>
<td>Yes* No!</td>
</tr>
<tr>
<td>DRT</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>EDRT</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

* = Takada & Sakamura, 1997

Flawed!

Theorem (Our technical result)

For GMF task systems, the schedulability problem for static priority schedulers is strongly coNP-hard.
Problem Overview

Task Models: L&L, GMF, DRT

Analysis Methods
- EDF: Demand Bound Function
- Static Priorities: Maximum Interference Function

Hardness Result
The Liu and Layland (L&L) Task Model
(Liu and Layland, 1973)

- Tasks are *periodic*
  - Job WCET $e$
  - Minimum inter-release delay $p$ (implicit deadline)

Advantages: Well-known model; efficient schedulability tests

However, *not everything is periodic*...
The General Multiframe (GMF) Task Model
(Baruah et al, 1999)

- Tasks *cycle* through job types, “frames”
  - Vector for WCET \((e^{(1)}, \ldots, e^{(n)})\)
  - Vector for deadlines \((d^{(1)}, \ldots, e^{(n)})\)
  - Vector for minimum inter-release delays \((p^{(1)}, \ldots, p^{(n)})\)
The Digraph Real-Time (DRT) Task Model
(S. et al, RTAS 2011)

- Branching, cycles (loops), ...
- *Directed graph* for each task
  - Vertices $J$: jobs to be released (with WCET and deadline)
  - Edges $(J_i, J_j)$: minimum inter-release delays $p(J_i, J_j)$
DRT: Semantics

Path $\pi = (J_4, J_2, J_3)$
Fahrplan

1 Problem Overview

2 Task Models: L&L, GMF, DRT

3 Analysis Methods
   • EDF: Demand Bound Function
   • Static Priorities: Maximum Interference Function

4 Hardness Result

Martin Stigge
Hardness of Static Priority Scheduling
The Demand Bound Function

- Given a time interval length \( t \)
- \( dbf(t) \) bounds the demand for processor time within any \( t \) interval

Theorem

A task system \( \tau \) is schedulable with EDF iff:

\[
\forall t \geq 0 : \sum_{T \in \tau} dbf_T(t) \leq t
\]
## Complexity of Schedulability Test

- **Efficient schedulability tests:**

<table>
<thead>
<tr>
<th></th>
<th>EDF</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td>L&amp;L</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>GMF</td>
<td>Yes</td>
<td>?</td>
</tr>
<tr>
<td>DRT</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>EDRT</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>
Schedulability for Static Priorities

- **L&L tasks:** *Response Time Analysis*
  \[
  R_i = C_i + \sum_{j \in hp(i)} \left\lfloor \frac{R_i}{T_j} \right\rfloor \cdot C_j
  \]

  Interference Term

- **Generalize for GMF:** *Maximum Interference Function (MIF)*
  - \( M_j(t) \): Maximum interference that \( \tau_j \) can cause within \( t \) time units
  \[
  R_i = C_i + \sum_{j \in hp(i)} M_j(R_i)
  \]
  - Efficiently computable for GMF/DRT/...

**BUT:** Inherently overapproximate!
MIF: Example

\[ \tau_1 \]

\[ M_1(t) \]

Martin Stigge
Hardness of Static Priority Scheduling
MIF: Combined Example

\[ \tau_1 \]

\[ t \]

\[ 10 \quad 12 \quad 14 \]

\[ \tau_2 \]

\[ t \]

\[ 10 \quad 12 \quad 14 \]

\[ \tau_1 + \tau_2 \]

\[ t \]

\[ 10 \quad 12 \quad 14 \]

\[ M_1(t) \]

\[ 5 \quad 6 \quad 7 \]

\[ t \]

\[ 10 \quad 12 \quad 14 \]

\[ M_2(t) \]

\[ 5 \quad 6 \quad 7 \]

\[ t \]

\[ 10 \quad 12 \quad 14 \]

\[ \sum M_j(t) \]

\[ 10 \quad 12 \quad 14 \quad 14 \]


class Actual tight MIF

Martin Stigge

Hardness of Static Priority Scheduling

15
Schedulability for Static Priorities

- In summary: MIF is *pessimistic*
- Possible improvement?
  - Define MIF additions better? Precise?
  - Use another abstraction level?
  - ...?
- No!

**Theorem (Our technical result)**

*For GMF task systems, the schedulability problem for static priority schedulers is strongly coNP-hard.*

- Thus: No precise efficient analysis possible.
Hardness Result: Proof sketch

**3-PARTITION instance I:**

- **m bins**
- **3m items**

Possible to (exactly) fit all items? (strongly NP-hard)

**Reduction to GMF schedulability:**

- One GMF task for each item
- Packing possible \( \iff \) Busy interval

Thus: \( \tau(I) \) unsched. \( \iff \) \( I \in 3\text{-PARTITION} \)
Summary and Outlook

<table>
<thead>
<tr>
<th></th>
<th>EDF</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td>L&amp;L</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>GMF</td>
<td>Yes</td>
<td>No!</td>
</tr>
<tr>
<td>DRT</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>EDRT</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Showed *intractability* of static scheduling for GMF

Insight:
- For EDF, “simple” overload test suffices (Local worst cases combine.)
- For static prio: More structure → more complex test (Local worst case unclear.)

Ongoing work:
- Solve anyway? Heuristics?
- Use SAT-/SMT-solvers
Thanks!