Combinatorial Abstraction Refinement for Feasibility Analysis

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/...

Schedulable?
- Task A
- Task B

Our Setting:
- DRT tasks
- Static Priorities
- Precise Test
Problem Overview

Workload Model

- High priority
  - Task A
  - Task B

- Medium priority
  - Task A
  - Task B

- Low priority
  - Task A

Scheduler Model

EDF/Static Prio/...

Our Setting:
- DRT tasks
- Static Priorities
- Precise Test
The Digraph Real-Time (DRT) Task Model
(S. et al, RTAS 2011)

- Generalizes periodic, sporadic, GMF, RRT, ...
- 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 \( \pi = (J_1, J_2) \)
Path \( \pi = (J_1, J_2, J_3) \)
DRT: Semantics

Path \( \pi = (J_1) \)
Path $\pi = (J_1, J_2)$
DRT: Semantics

Path \( \pi = (J_1, J_2, J_3) \)
Complexity Results for DRT Schedulability

**EDF**
- *Pseudo-polynomial*
- Dbf-based analysis
  [RTAS 2011]
- Equivalent to Feasibility

**Static Priorities**
- Strongly \textit{coNP-hard}
- Already for trees or cycles
  [ECRTS 2012]
- Efficient solution?
## Complexity Results for DRT Schedulability

<table>
<thead>
<tr>
<th>EDF</th>
<th>Static Priorities</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Pseudo-polynomial</em></td>
<td><em>Strongly coNP-hard</em></td>
</tr>
<tr>
<td>Dbf-based analysis</td>
<td><em>Already for trees or cycles</em></td>
</tr>
<tr>
<td>[RTAS 2011]</td>
<td>[ECRTS 2012]</td>
</tr>
<tr>
<td>Equivalent to Feasibility</td>
<td><em>Efficient solution?</em></td>
</tr>
</tbody>
</table>
Fahrplan

1 Problem Introduction
   - Digraph Real-Time Tasks
   - Complexity Results

2 Analysis Approach
   - Request Functions
   - Rf-based Test

3 Combinatorial Abstraction Refinement
   - Abstraction Trees
   - Refinement Procedure

4 Evaluation
Fahrplan

1 Problem Introduction
   - Digraph Real-Time Tasks
   - Complexity Results

2 Analysis Approach
   - Request Functions
   - Rf-based Test

3 Combinatorial Abstraction Refinement
   - Abstraction Trees
   - Refinement Procedure

4 Evaluation
Testing the Scheduling Window

High priority

Medium priority

Low priority

Is C schedulable?

Scheduling window of C
Testing the Scheduling Window

Is C schedulable?

Scheduling window of C
Request Functions

1. $J_1 \langle 6, 10 \rangle \rightarrow J_2 \langle 5, 25 \rangle$
2. $J_2 \langle 5, 25 \rangle \rightarrow J_3 \langle 1, 10 \rangle$
3. $J_3 \langle 1, 10 \rangle \rightarrow J_4 \langle 2, 12 \rangle$
4. $J_4 \langle 2, 12 \rangle \rightarrow J_1 \langle 6, 10 \rangle$
5. $J_4 \langle 2, 12 \rangle \rightarrow J_5 \langle 10, 50 \rangle$

$rf(t)$

$t$
Lemma

A job $J$ is schedulable iff for all combinations of request functions $rf^{(T)}$ of higher priority tasks:

$$\exists t \leq d(J) : e(J) + \sum_{T \in \tau} rf^{(T)}(t) \leq t.$$  \hspace{1cm} (1)

![Diagram showing the scheduling window and the condition for schedulability](image)
Lemma

A job $J$ is schedulable iff for all combinations of request functions $rf^{(T)}$ of higher priority tasks:

\[ \exists t \leq d(J) : e(J) + \sum_{T \in \tau} rf^{(T)}(t) \leq t. \]  

(1)
Lemma

A job $J$ is schedulable iff for all combinations of request functions $rf(T)$ of higher priority tasks:

$$\exists t \leq d(J) : e(J) + \sum_{T \in \tau} rf(T)(t) \leq t.$$  \hfill (1)

Problem: Naive test *double exponential!*

1. Number of paths per task
2. Number of combinations
Request Functions: Domination

\[ \langle 6, 10 \rangle, \langle 5, 25 \rangle, \langle 10, 50 \rangle, \langle 1, 10 \rangle, \langle 2, 12 \rangle \]

\[ rf(t) \]

\[ rf(J_1, J_2, J_3) \]

Martin Stigge
Combinatorial Abstraction Refinement
Request Functions: Domination

\[ J_1 \langle 6, 10 \rangle \quad J_2 \langle 5, 25 \rangle \quad J_3 \langle 1, 10 \rangle \quad J_4 \langle 2, 12 \rangle \quad J_5 \langle 10, 50 \rangle \]

\[ rf(t) \]

\[ rf(J_1, J_2, J_3) \quad rf(J_3, J_4, J_2) \]
Request Functions: Domination

\[ J_1 \langle 6, 10 \rangle \rightarrow J_2 \langle 5, 25 \rangle \rightarrow J_3 \langle 1, 10 \rangle \rightarrow J_5 \langle 10, 50 \rangle \]

\[ J_2 \langle 13 \rangle \rightarrow J_4 \langle 12 \rangle \rightarrow J_3 \langle 25 \rangle \rightarrow J_5 \langle 50 \rangle \]

\[ rf(t) \]

\[ rf(J_1, J_2, J_3) \rightarrow rf(J_3, J_4, J_2) \]
Combinatorial Explosion

Lemma

A job $J$ is schedulable if for all combinations of request functions $rf^{(T)}$ of higher priority tasks:

$$\exists t \leq d(J) : e(J) + \sum_{T \in \tau} rf^{(T)}(t) \leq t.$$  (1)

What about the Combinatorial Explosion?
Overapproximation: \textit{mrf}

- **Approach:** Define an overapproximation
- \textit{mrf}^{(T)}(t): \textit{Maximum} of all \textit{rf}^{(T)}(t) for a task \textit{T}
  - “Request-Bound Function”
  - “Workload-Arrival Function”
- **New test:**
  \[ \exists t \leq d(J) : e(J) + \sum_{T \in \tau} \textit{mrf}^{(T)}(t) \leq t. \]
- **Efficient:** Only \textit{one} test, no combinatorial explosion
Overapproximation: \( mrf \)

- **Approach:** Define an overapproximation
- \( mrf^{(T)}(t) \): *Maximum* of all \( rf^{(T)}(t) \) for a task \( T \)
  - “Request-Bound Function”
  - “Workload-Arrival Function”
- **New test:**
  \[ \exists t \leq d(J) : e(J) + \sum_{T \in \tau} mrf^{(T)}(t) \leq t. \]
- **Efficient:** Only one test, no combinatorial explosion
- **Problem:** Imprecise!
Overapproximation: mrf

- Approach: Define an overapproximation
- $mrf^T(t)$: Maximum of all $rf^T(t)$ for a task $T$
  - “Request-Bound Function”
  - “Workload-Arrival Function”
- New test:
  \[
  \exists t \leq d(J) : e(J) + \sum_{T \in \tau} mrf^T(t) \leq t.
  \]
- **Efficient**: Only one test, no combinatorial explosion
- Problem: Imprecise!

**How can we get efficiency and precision?**
Define an *abstraction tree* per task:

- Leaves are concrete *rf*
- Each node: maximum function of child nodes
- Root is *mrf*, maximum of *all rf*
Define an *abstraction tree* per task:

- Leaves are concrete *rf*
- Each node: maximum function of child nodes
- Root is *mrf*, maximum of all *rf*
Abstraction Tree

Define an *abstraction tree* per task:

- Leaves are concrete *rf*
- Each node: maximum function of child nodes
- Root is *mrf*, maximum of all *rf*
Abstraction Tree

Define an *abstraction tree* per task:

- Leaves are concrete *rf*
- Each node: maximum function of child nodes
- Root is *mrf*, maximum of *all* *rf*
Define an *abstraction tree* per task:

- Leaves are concrete *rf*
- Each node: maximum function of child nodes
- Root is *mrf*, maximum of *all* *rf*
New Algorithm:

- Test *one* combination of all \( mrf \).
- If schedulable: done
- Otherwise: Replace *one* \( mrf \) with all child nodes,
  - 2 new combinations to test
- Repeat until:
  - All combinations show schedulability, or
  - A combination of leaves shows non-schedulability
Combinatorial Abstraction Refinement: Example

Task $T_1$

Task $T_3$

Task $T_3$

Task $T_4$

Testing $rf$ tuples:

$$(A, A, A, A)$$

Result: Schedulable!

Total combinations: $3 \cdot 2 \cdot 4 \cdot 3 = 72$; Tested: 5 (!)
Combinatorial Abstraction Refinement: Example

Testing rf tuples:

\((A, A, A, A)\) ?

Test: \(\exists t \leq d(J) : e(J) + \sum_{T \in \tau} rf(T)(t) \leq t\)
Combinatorial Abstraction Refinement: Example

Task $T_1$

Task $T_3$

Task $T_3$

Task $T_4$

Testing $rf$ tuples:

$$(A, A, A, A)$$

UNSCHED
Testing \( rf \) tuples:

\[
\begin{align*}
(A, A, A, A) && \text{UNSCHED} \\
(B, A, A, A) \\
(C, A, A, A)
\end{align*}
\]
Combinatorial Abstraction Refinement: Example

Testing rf tuples:

- \((A, A, A, A)\) UNSCHED
- \((B, A, A, A)\)
- \((C, A, A, A)\)

Test: \(\exists t \leq d(J) : e(J) + \sum_{T \in \tau} rf(T)(t) \leq t\)
Combinatorial Abstraction Refinement: Example

Testing $rf$ tuples:

- $(A, A, A, A)$: UNSCHED
- $(B, A, A, A)$: SCHED
- $(C, A, A, A)$

Result: Schedulable!

Total combinations: $3 \cdot 2 \cdot 4 \cdot 3 = 72$; Tested: 5!
Combinatorial Abstraction Refinement: Example

Testing $rf$ tuples:

- $(A, A, A, A, A)$: UNSCHED
- $(B, A, A, A, A)$: SCHED
- $(C, A, A, A, A)$: ?

Test: $\exists t \leq d(J) : e(J) + \sum_{T \in \tau} rf(T)(t) \leq t$
Combinatorial Abstraction Refinement: Example

Task $T_1$

Task $T_3$

Task $T_3$

Task $T_4$

Testing $rf$ tuples:

- $(A, A, A, A)$: UNSCHED
- $(B, A, A, A)$: SCHED
- $(C, A, A, A)$: UNSCHED
- $(C, A, A, A)$: UNSCHED
Combinatorial Abstraction Refinement: Example

Task $T_1$

Testing $rf$ tuples:

- $(A, A, A, A)$: UNSCHED
- $(B, A, A, A)$: SCHED
- $(C, A, A, A)$: UNSCHED
- $(C, A, B, A)$
- $(C, A, C, A)$

Result: Schedulable!

Total combinations: $3 \cdot 2 \cdot 4 \cdot 3 = 72$; Tested: 5
Combinatorial Abstraction Refinement: Example

Testing rf tuples:

- $(A, A, A, A)$ \textbf{UNSCHED}
- $(B, A, A, A, A)$ \textbf{SCHED}
- $(C, A, A, A)$ \textbf{UNSCHED}
- $(C, A, B, A)$ ?
- $(C, A, C, A)$

\textbf{Test: } $\exists t \leq d(J) : e(J) + \sum_{T \in \tau} rf(T)(t) \leq t$
Combinatorial Abstraction Refinement: Example

Testing rf tuples:

- $(A, A, A, A)$: UNSCHED
- $(B, A, A, A)$: SCHED
- $(C, A, A, A)$: UNSCHED
- $(C, A, B, A)$: SCHED
- $(C, A, C, A)$:

Result: Schedulable!

Total combinations: $3 \cdot 2 \cdot 4 \cdot 3 = 72$; Tested: 5
Combinatorial Abstraction Refinement: Example

Testing rf tuples:

\[(A, A, A, A) \quad \text{UNSCHED} \]
\[(B, A, A, A) \quad \text{SCHED} \]
\[(C, A, A, A) \quad \text{UNSCHED} \]
\[(C, A, B, A) \quad \text{SCHED} \]
\[(C, A, C, A) \quad ? \]

Test: \( \exists t \leq d(J) : e(J) + \sum_{T \in \tau} rf(T)(t) \leq t \)
Combinatorial Abstraction Refinement: Example

Task $T_1$

Testing $rf$ tuples:

- $(A, A, A, A)$ UNSCHED
- $(B, A, A, A)$ SCHED
- $(C, A, A, A)$ UNSCHED
- $(C, A, B, A)$ SCHED
- $(C, A, C, A)$ SCHED

Result: *Schedulable!*
Combinatorial Abstraction Refinement: Example

Task \( T_1 \)

- A
- B
- C
- D
- E

Task \( T_3 \)

- A
- B
- C

Task \( T_3 \)

- A
- B
- C
- D
- E
- F
- G

Task \( T_4 \)

- A
- B
- C
- D
- E

Testing \( rf \) tuples:

- \((A, A, A, A)\) \(\text{UNSCHEDE}\)
- \((B, A, A, A)\) \(\text{SCHED}\)
- \((C, A, A, A)\) \(\text{UNSCHEDE}\)
- \((C, A, B, A)\) \(\text{SCHED}\)
- \((C, A, C, A)\) \(\text{SCHED}\)

Result: \(\text{Schedulable!}\)

Total combinations: \(3 \cdot 2 \cdot 4 \cdot 3 = 72\); Tested: 5 (!)
Fahrplan

1 Problem Introduction
   - Digraph Real-Time Tasks
   - Complexity Results

2 Analysis Approach
   - Request Functions
   - Rf-based Test

3 Combinatorial Abstraction Refinement
   - Abstraction Trees
   - Refinement Procedure

4 Evaluation
Fahrplan

1 Problem Introduction
   - Digraph Real-Time Tasks
   - Complexity Results

2 Analysis Approach
   - Request Functions
   - Rf-based Test

3 Combinatorial Abstraction Refinement
   - Abstraction Trees
   - Refinement Procedure

4 Evaluation
Evaluation: Runtime vs. Utilization

Comparing runtimes of

- EDF-test using dbf (pseudo-polynomial)
- SP-test based on *Combinatorial Abstraction Refinement*
Evaluation: Tested vs. Total Combinations

10^5 samples of single-job tests.
- Executed tests: in 99.9% of all cases, less than 100
- Total combinations possible: 10^{12} or more
Summary and Outlook

- Solve coNP-hard problem
  - Previously unsolved
  - Efficient method
- Abstraction refinement
  - General method
  - Combinatorial problems
  - Needs abstraction lattice

- Ongoing work:
  - Response-Time Analysis (submitted)
  - Apply to other problems
Q & A

Thanks!