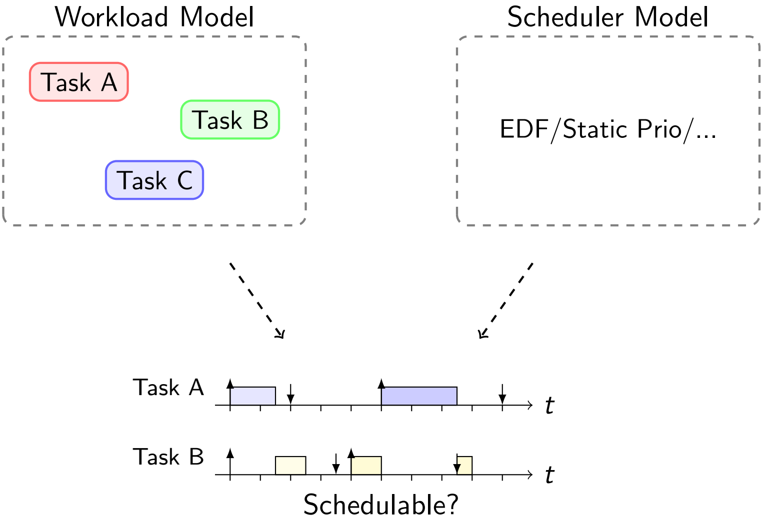
Combinatorial Abstraction Refinement for Feasibility Analysis

Martin Stigge

Uppsala University, Sweden

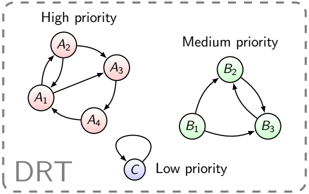
Joint work with Wang Yi

Problem Overview



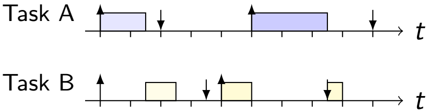
Problem Overview

Workload Model



Scheduler Model

EDF/Static Prio/...



Schedulable?

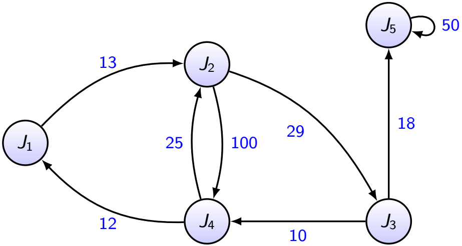
Our Setting:

- DRT tasks
- Static Priorities
- Precise Test

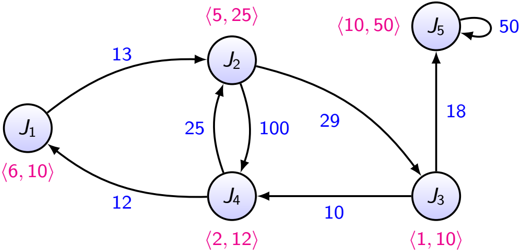
The Digraph Real-Time (DRT) Task Model

(S. et al, RTAS 2011)

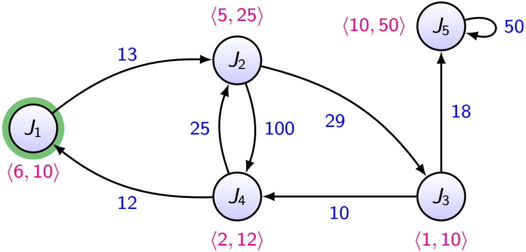
- Generalizes periodic, sporadic, GMF, RRT, ...
- *Directed graph* for each task
  - Vertices J: jobs to be released (with WCET and deadline)
  - Edges (J_i, J_j): minimum inter-release delays $p(J_i, J_j)$



DRT: Semantics

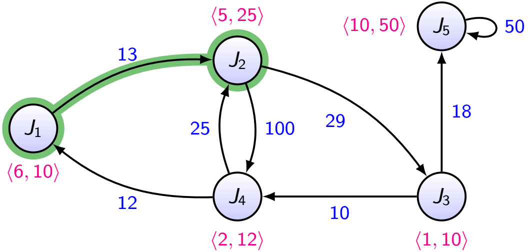


DRT: Semantics

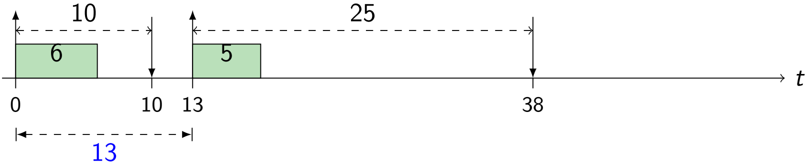


Path $\pi = (J_1)$

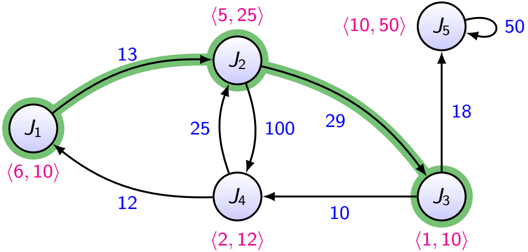
DRT: Semantics



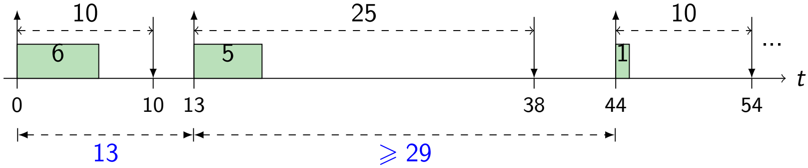
Path $\pi = (J_1, J_2)$



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 *coNP-hard*
- Already for trees or cycles [ECRTS 2012]
- Efficient solution?

Complexity Results for DRT Schedulability

EDF

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

Static Priorities

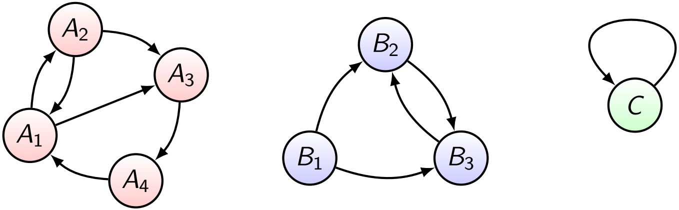
- Strongly *coNP-hard*
- Already for trees or cycles [ECRTS 2012]
- Efficient solution?

Fahrplan

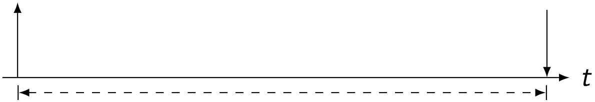
Fahrplan

Testing the Scheduling Window

High priority Medium priority Low priority

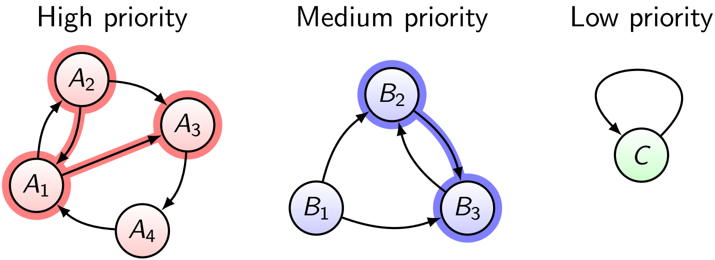


Is C schedulable?

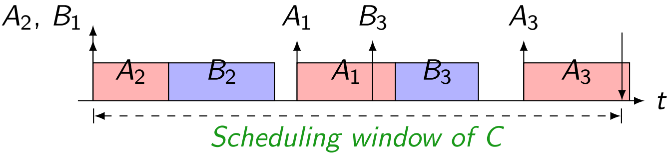


Scheduling window of C

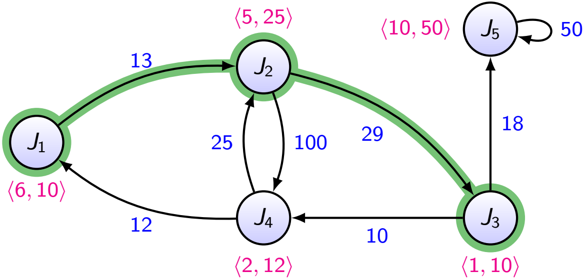
Testing the Scheduling Window

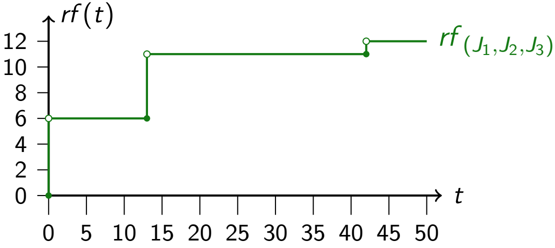


Is C schedulable?



Request Functions



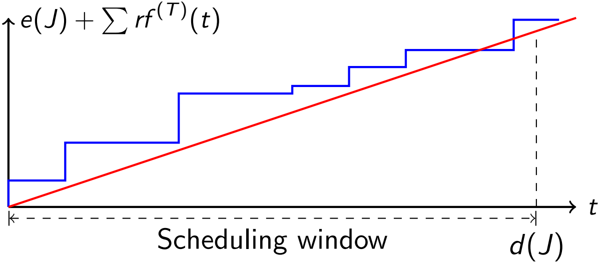


Request Functions: Schedulability Test

Lemma

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

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

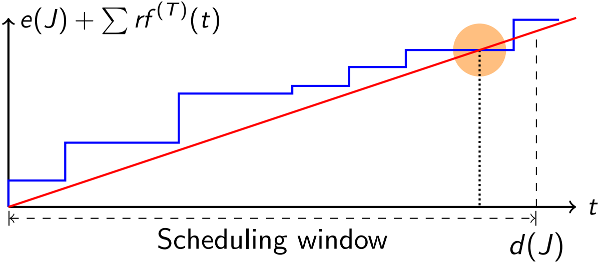


Request Functions: Schedulability Test

Lemma

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

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

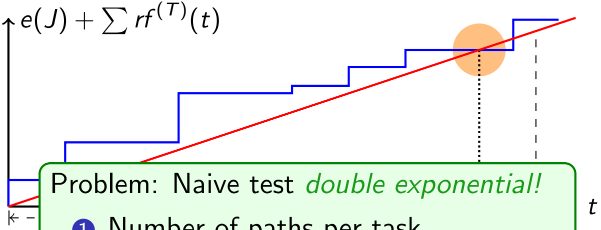


Request Functions: Schedulability Test

Lemma

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

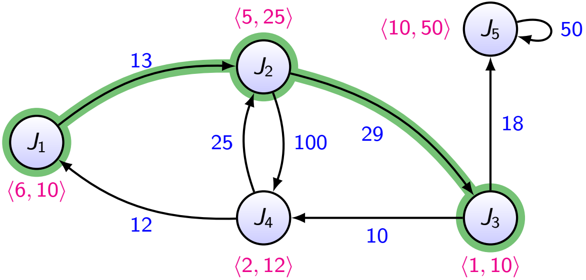
$$\exists t \leqslant d(J) : e(J) + \sum_{T \in \tau} rf^{(T)}(t) \leqslant t. \tag{1}$$

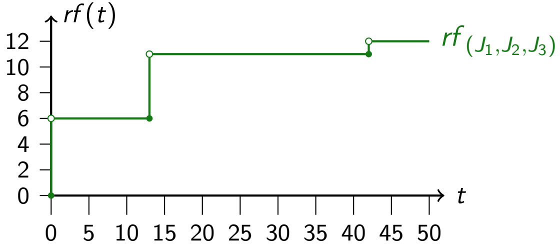


Problem: Naive test *double exponential!*

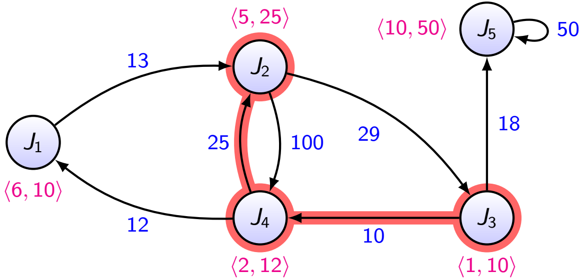
1. Number of paths per task
2. Number of combinations

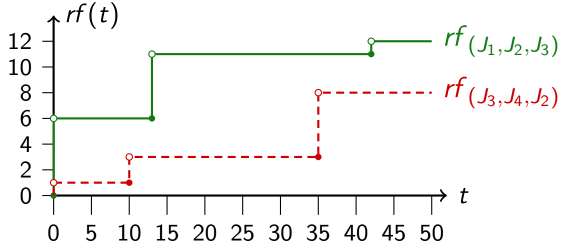
Request Functions: Domination



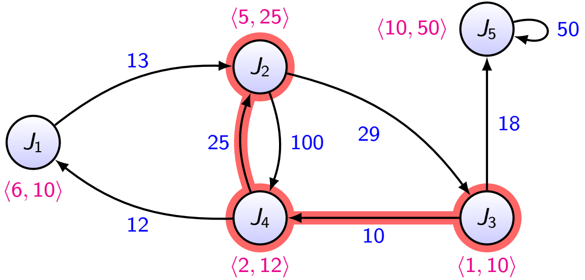


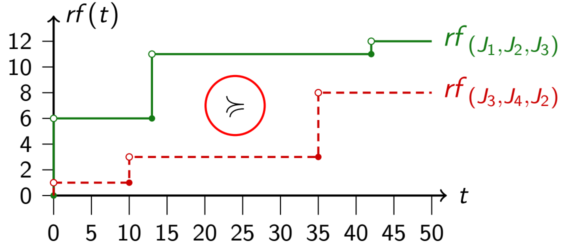
Request Functions: Domination





Request Functions: Domination





Combinatorial Explosion

Lemma

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

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

What about the *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:

$$\exists t \leqslant d(J) : e(J) + \sum_{T \in \tau} mrf^{(T)}(t) \leqslant t.$$

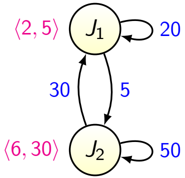
- *Efficient*: Only *one* test, no combinatorial explosion

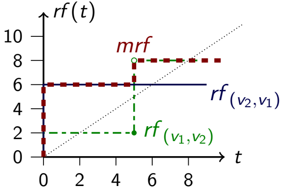
Overapproximation: *mrf*

- 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:

$$\exists t \leqslant d(J) : e(J) + \sum_{T \in \tau} mrf^{(T)}(t) \leqslant t.$$

- *Efficient*: Only *one* test, no combinatorial explosion
- Problem: Imprecise!



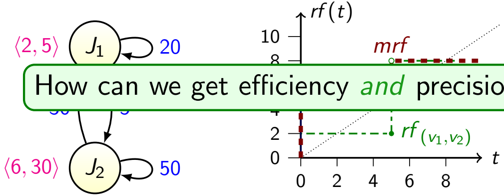


Overapproximation: *mrf*

- 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:

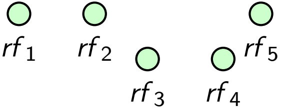
$$\exists t \leqslant d(J) : e(J) + \sum_{T \in \tau} mrf^{(T)}(t) \leqslant t.$$

- *Efficient*: Only *one* test, no combinatorial explosion
- Problem: Imprecise!



How can we get efficiency *and* precision?

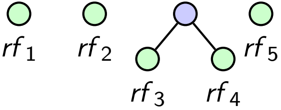
Abstraction Tree



Define an *abstraction tree* per task:

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

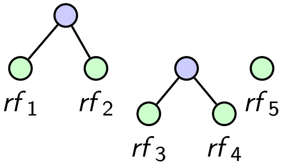
Abstraction Tree



Define an *abstraction tree* per task:

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

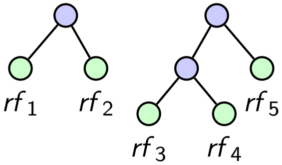
Abstraction Tree



Define an *abstraction tree* per task:

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

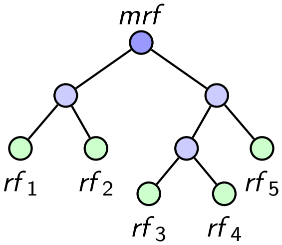
Abstraction Tree



Define an *abstraction tree* per task:

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

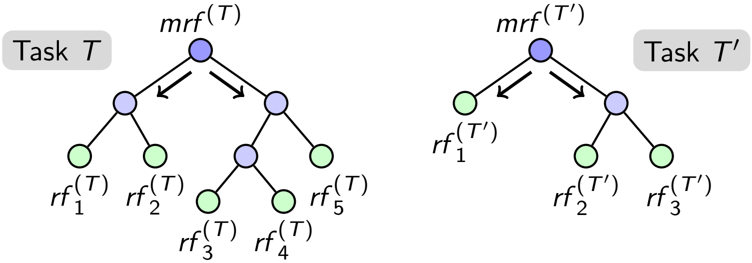
Abstraction Tree



Define an *abstraction tree* per task:

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

Combinatorial Abstraction Refinement

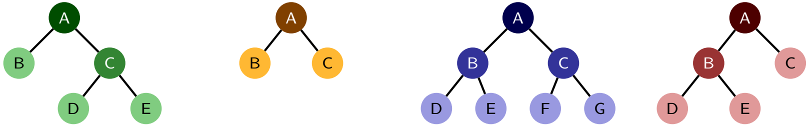


New Algorithm:

- 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

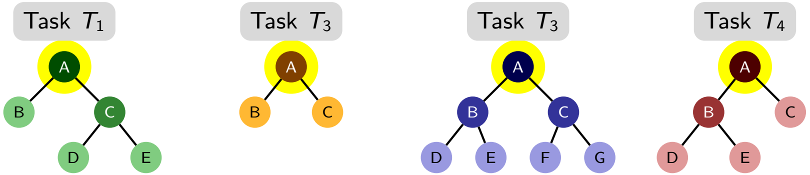
Combinatorial Abstraction Refinement: Example

Task T_1 Task T_3 Task T_3 Task T_4



Testing *rf* tuples:

Combinatorial Abstraction Refinement: Example

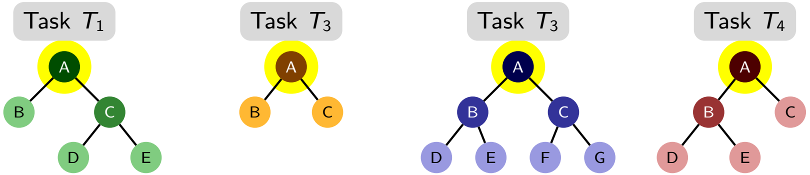


Testing *rf* tuples:

(A, A, A, A) **?**

Test: $\exists t \leqslant d(J) : e(J) + \sum_{T \in \tau} rf^{(T)}(t) \leqslant t$

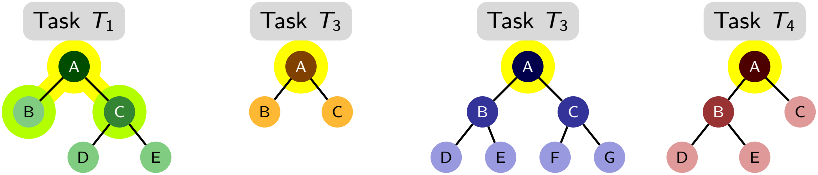
Combinatorial Abstraction Refinement: Example



Testing *rf* tuples:

UNSCHED

Combinatorial Abstraction Refinement: Example



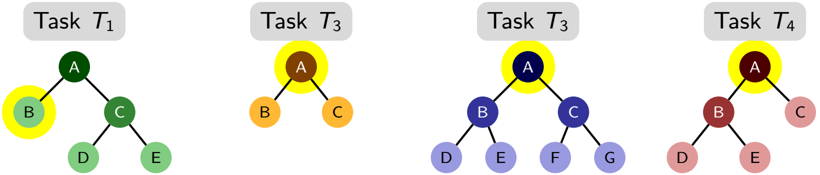
Testing *rf* tuples:

(A, A, A, A) **UNSCHED**

(B, A, A, A)

(C, A, A, A)

Combinatorial Abstraction Refinement: Example



Testing *rf* tuples:

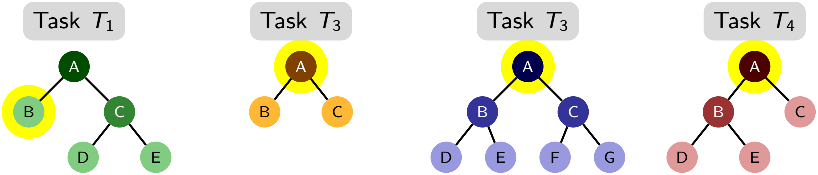
(A, A, A, A) **UNSCHED**

(B, A, A, A) **?**

(C, A, A, A)

Test: $\exists t \leqslant d(J) : e(J) + \sum_{T \in \tau} rf^{(T)}(t) \leqslant t$

Combinatorial Abstraction Refinement: Example



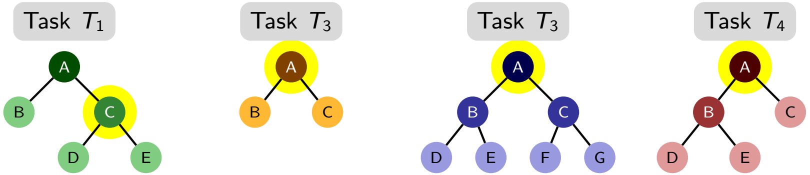
Testing *rf* tuples:

(A, A, A, A) **UNSCHED**

(B, A, A, A) **SCHED**

(C, A, A, A)

Combinatorial Abstraction Refinement: Example



Testing *rf* tuples:

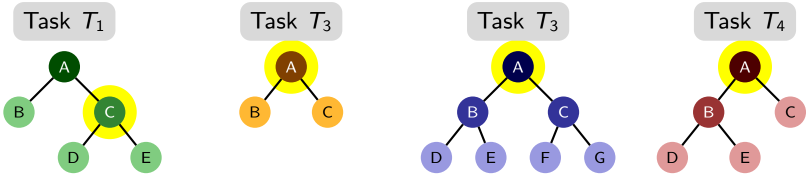
(A, A, A, A) **UNSCHED**

(B, A, A, A) **SCHED**

(C, A, A, A) **?**

Test: $\exists t \leqslant d(J) : e(J) + \sum_{T \in \tau} rf^{(T)}(t) \leqslant t$

Combinatorial Abstraction Refinement: Example



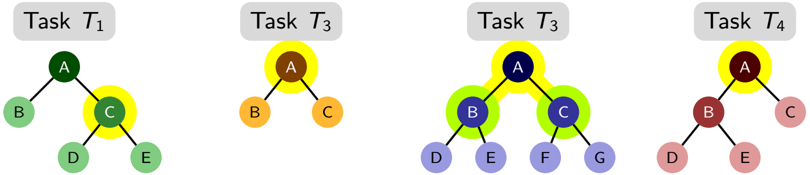
Testing *rf* tuples:

(A, A, A, A) **UNSCHED**

(B, A, A, A) **SCHED**

(C, A, A, A) **UNSCHED**

Combinatorial Abstraction Refinement: Example



Testing *rf* tuples:

(A, A, A, A) **UNSCHED**

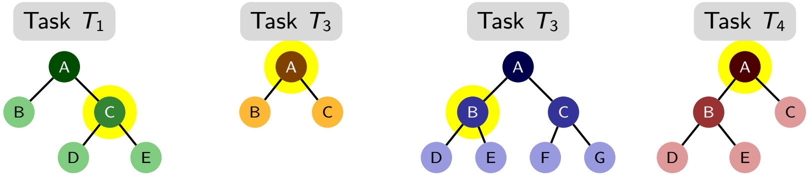
(B, A, A, A) **SCHED**

(C, A, A, A) **UNSCHED**

(C, A, B, A)

(C, A, C, A)

Combinatorial Abstraction Refinement: Example



Testing *rf* tuples:

(A, A, A, A) **UNSCHED**

(B, A, A, A) **SCHED**

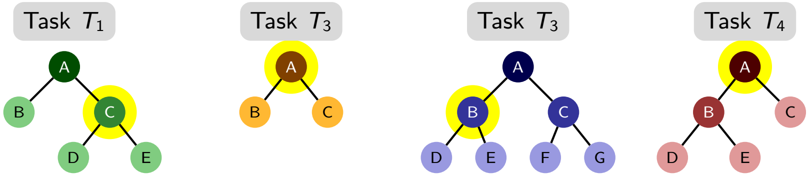
(C, A, A, A) **UNSCHED**

(C, A, B, A) **?**

(C, A, C, A)

Test: $\exists t \leqslant d(J) : e(J) + \sum_{T \in \tau} rf^{(T)}(t) \leqslant t$

Combinatorial Abstraction Refinement: Example



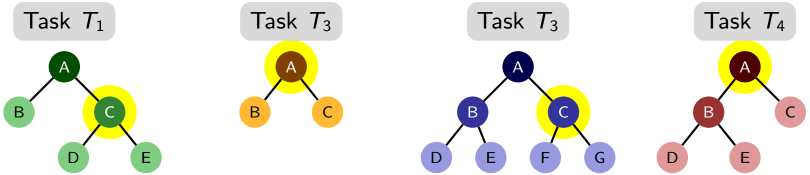
Testing *rf* tuples:

(A, A, A, A) **UNSCHED**

(B, A, A, A) **SCHED**
(C, A, A, A) **UNSCHED**

(C, A, B, A) **SCHED**
(C, A, C, A)

Combinatorial Abstraction Refinement: Example



Testing *rf* tuples:

(A, A, A, A) **UNSCHED**

(B, A, A, A) **SCHED**

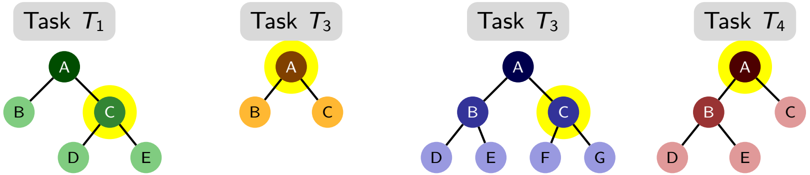
(C, A, A, A) **UNSCHED**

(C, A, B, A) **SCHED**

(C, A, C, A) **?**

Test: $\exists t \leqslant d(J) : e(J) + \sum_{T \in \tau} rf^{(T)}(t) \leqslant t$

Combinatorial Abstraction Refinement: Example



Testing *rf* tuples:

(A, A, A, A) **UNSCHED**

(B, A, A, A) **SCHED**

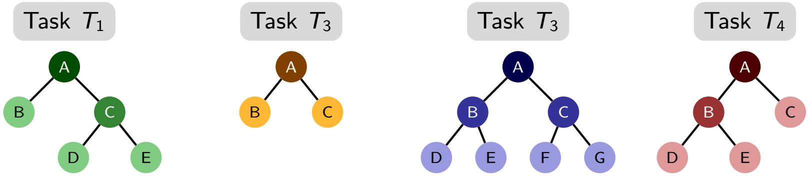
(C, A, A, A) **UNSCHED**

(C, A, B, A) **SCHED**

(C, A, C, A) **SCHED**

Result: *Schedulable!*

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) **SCHED**

Result: *Schedulable!*

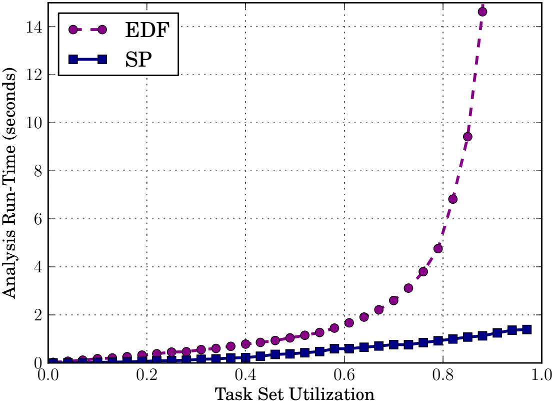
Total combinations: $3 \cdot 2 \cdot 4 \cdot 3 = 72$; Tested: 5 (!)

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

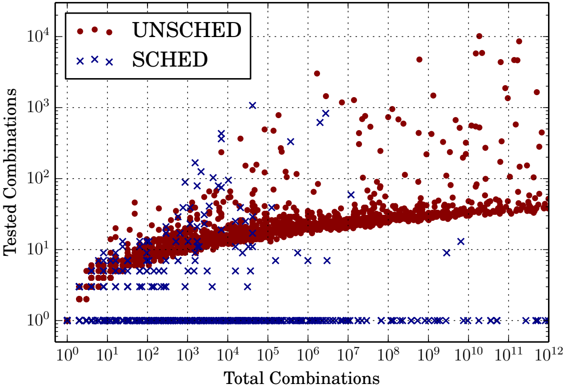
Evaluation: Runtime vs. Utilization



Comparing runtimes of

- EDF-test using dbf (pseudo-polynomial)
- SP-test based on *Combinatorial Abstraction Refinement*

Evaluation: Tested vs. Total Combinations

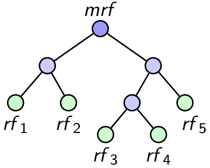


10^5 samples of single-job tests.

- 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!