ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Engineering and
Networks Laboratory TIK

Energy-Efficient Static Priority and Speed Assignment for Real-Time Tasks with Non-Deterministic Release Times

Simon Perathoner [1], Jian-Jia Chen [2], Lothar Thiele [1]

[1] ETH Zurich (Switzerland) [2] KIT Karlsruhe (Germany)

RTCSA, August 24th, 2010, Macau SAR, China

Outline

- Motivation
- Problem Definition
- Analysis Framework
- Proposed Algorithms
- Experimental Evaluation
- Conclusion

Embedded Systems

Computer systems physically embedded into larger device

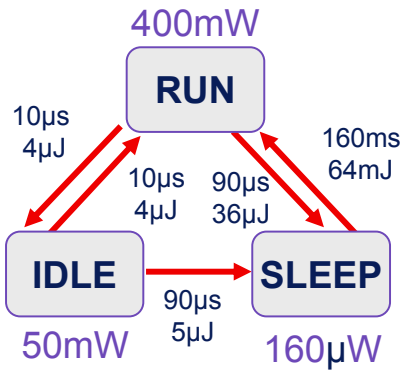
Requirements

- **Performance**
- **Energy**
- **Fault tolerance**
- **Size**
- **Weight**
- **Cost**

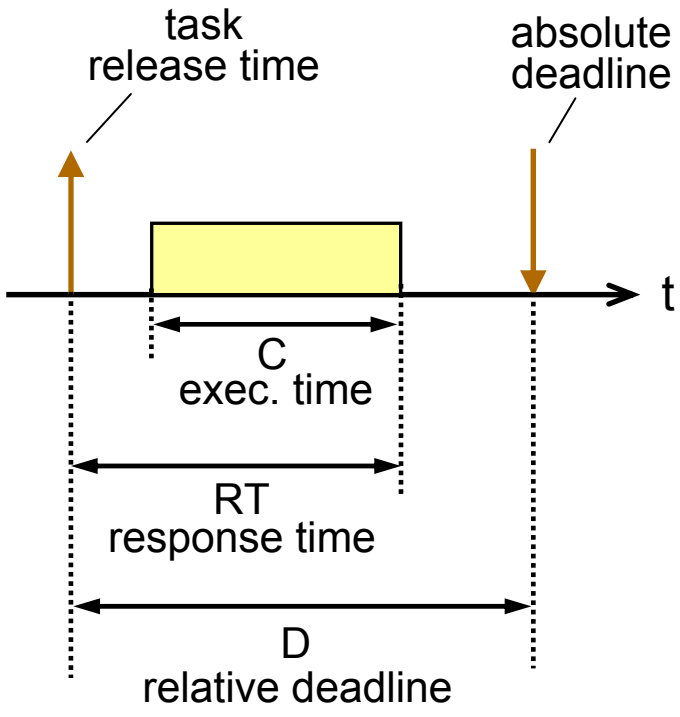
Methods for CPU Power Management

- **Dynamic Voltage Scaling (DVS)**
  - For CMOS circuits $E \propto V_{dd}^2$ (ignoring leakage)
  - Save energy by reducing supply voltage (clock frequency)

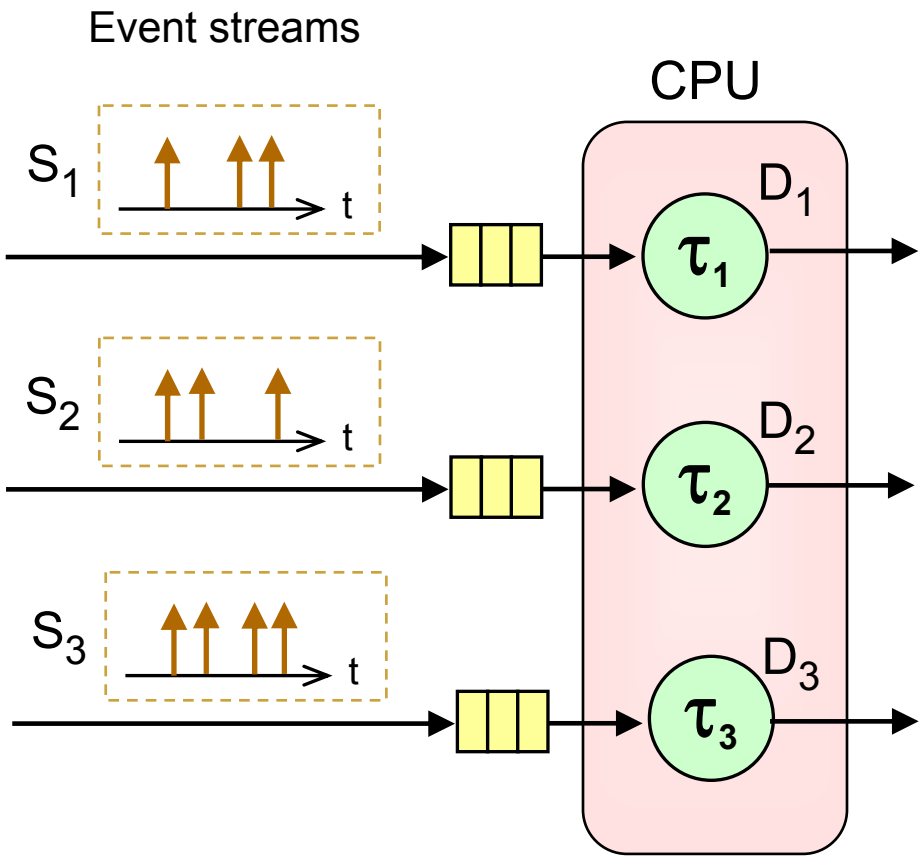
- **Dynamic Power Management (DPM)**
  - Switch between different power states
  - Helps to reduce power dissipation due to leakage



Real-Time System



$$WCRT \leq D$$



Goal: Minimize energy consumption while guaranteeing deadlines

Related Work

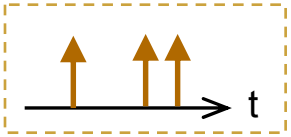
- **Offline Approaches**
  - Take scheduling decisions statically (at design time) based on expected worst-case workload
  - E.g. [Yao et al. 1995], [Quan et al. 2007], [Yun et al. 2003]

- **Online Approaches**
  - Take scheduling decisions dynamically (at run time) by adapting to actual workload
  - E.g. [Yao et al. 1995], [Aydin et al. 2001], [Mochocki et al. 2005]

Motivation

Limitations of existing approaches:

1) **Very restrictive assumptions on task release times**



Examples:
- Periodic / sporadic event streams
- Full a priori knowledge of release times

But in real systems task release times are less deterministic !

- Tasks triggered by physical events
- Tasks triggered by other tasks in complex distributed architectures

2) **Task priorities are not subject to optimization**

Outline

- Motivation
- Problem Definition
- Analysis Framework
- Proposed Algorithms
- Experimental Evaluation
- Conclusion

Problem Definition

Task set $\Gamma = \{\tau_1, \cdots, \tau_N\}$

Each task τ_i **is characterized by:**

- $\alpha_i(\Delta)$ **Arrival curve**
- C_i **Execution time at** s_{max}
- D_i **Deadline**

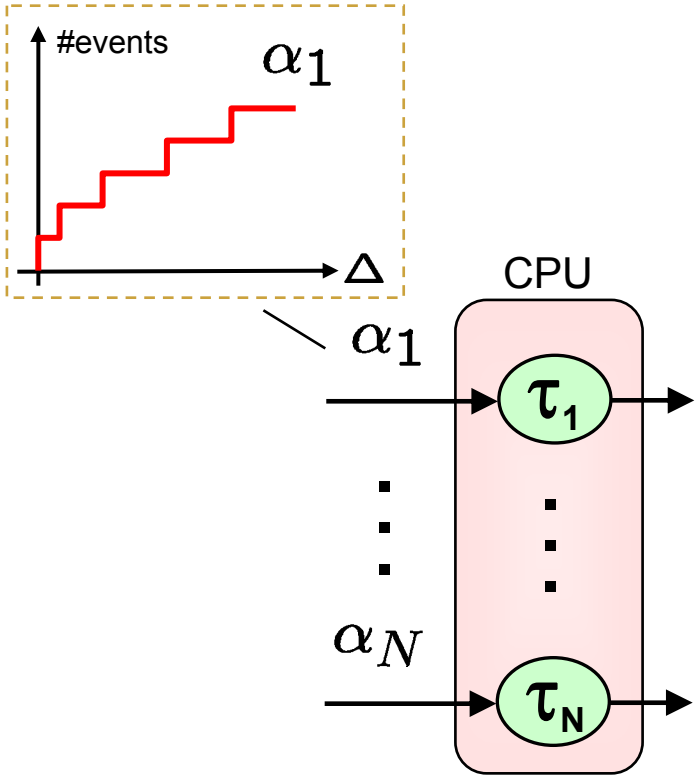
CPU with execution speeds

$\mathcal{S} = [s_{min}, s_{max}]$ **or**

$\mathcal{S} = \{s_1, s_2, \cdots, s_K\}$

Priority assignment $\Pi: \Gamma \rightarrow \{1, \cdots, N\}$

Speed assignment $\Sigma: \Gamma \rightarrow \mathcal{S}$



Find <u>static</u> Π, Σ **such that**

- $WCRT_{\tau_i} \leq D_i \quad \forall i \in \{1, \cdots, N\}$
- $E(\Delta)$ **is minimized**

Example

$\Gamma = \{\tau_1, \tau_2, \tau_3\}$ **Task release: Periodic with non-deterministic but bounded jitter**

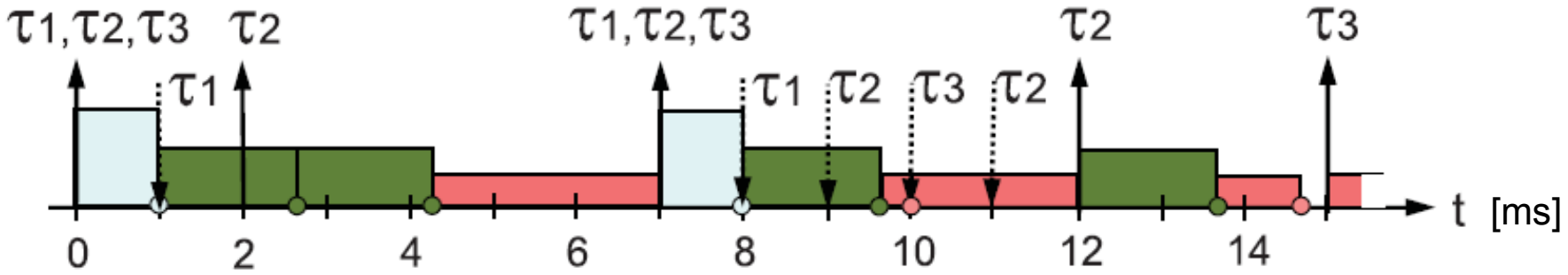
$p_1 = 10 \quad j_1 = 3 \quad C_1 = 1 \quad D_1 = 1$ [ms]
$p_2 = 5 \quad j_2 = 3 \quad C_2 = 1 \quad D_2 = 9$
$p_3 = 8 \quad j_3 = 1 \quad C_3 = 1 \quad D_3 = 10$

$$P(s) = \hbar(0.08 + 1.52s^3)\ Watt$$

(a)

$$\Pi(\tau_1) = 1 \qquad \Pi(\tau_2) = 2 \qquad \Pi(\tau_3) = 3$$
$$\Sigma(\tau_1) = 1 \qquad \Sigma(\tau_2) = 0.6 \qquad \Sigma(\tau_3) = 0.\bar{3}$$

$$E_a(10s) = 3475mJ$$



Example

$\Gamma = \{\tau_1, \tau_2, \tau_3\}$ **Task release: Periodic with non-deterministic but bounded jitter**

$p_1 = 10 \quad j_1 = 3 \quad C_1 = 1 \quad D_1 = 1$ [ms]
$p_2 = 5 \quad j_2 = 3 \quad C_2 = 1 \quad D_2 = 9$
$p_3 = 8 \quad j_3 = 1 \quad C_3 = 1 \quad D_3 = 10$

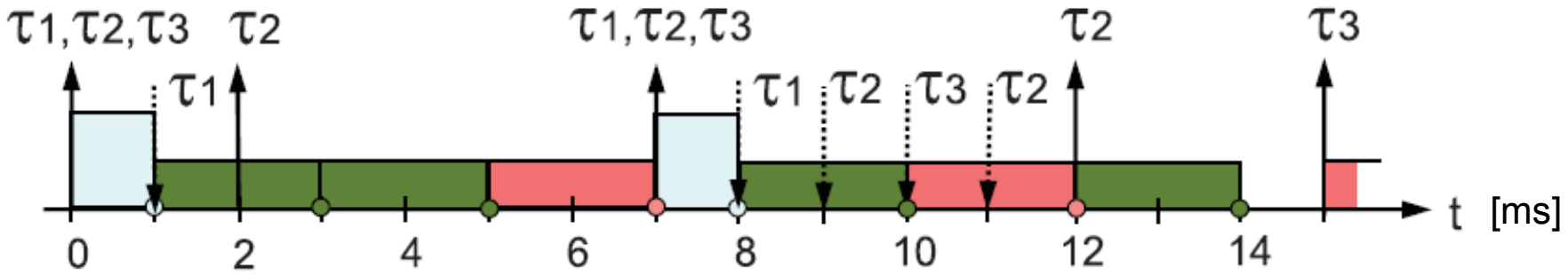
$$P(s) = \hbar(0.08 + 1.52s^3)\ Watt$$

(b)

$\Pi(\tau_1) = 1 \qquad \Pi(\tau_2) = 2 \qquad \Pi(\tau_3) = 3$

$\Sigma(\tau_1) = 1 \qquad \Sigma(\tau_2) = 0.5 \qquad \Sigma(\tau_3) = 0.5$

$E_b(10s) = 3357mJ$



Example

$\Gamma = \{\tau_1, \tau_2, \tau_3\}$ **Task release: Periodic with non-deterministic but bounded jitter**

$p_1 = 10 \quad j_1 = 3 \quad C_1 = 1 \quad D_1 = 1$ [ms]
$p_2 = 5 \quad j_2 = 3 \quad C_2 = 1 \quad D_2 = 9$
$p_3 = 8 \quad j_3 = 1 \quad C_3 = 1 \quad D_3 = 10$

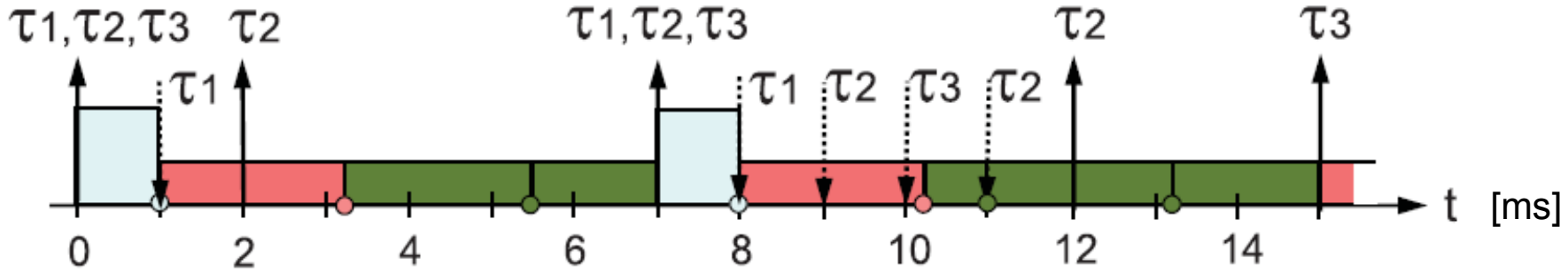
$$P(s) = \hbar(0.08 + 1.52s^3)\ Watt$$

(c)

$\Pi(\tau_1) = 1 \quad \Pi(\tau_2) = 3 \quad \Pi(\tau_3) = 2$

$\Sigma(\tau_1) = 1 \quad \Sigma(\tau_2) = 0.445 \quad \Sigma(\tau_3) = 0.445$

$E_c(10s) = 3163mJ$

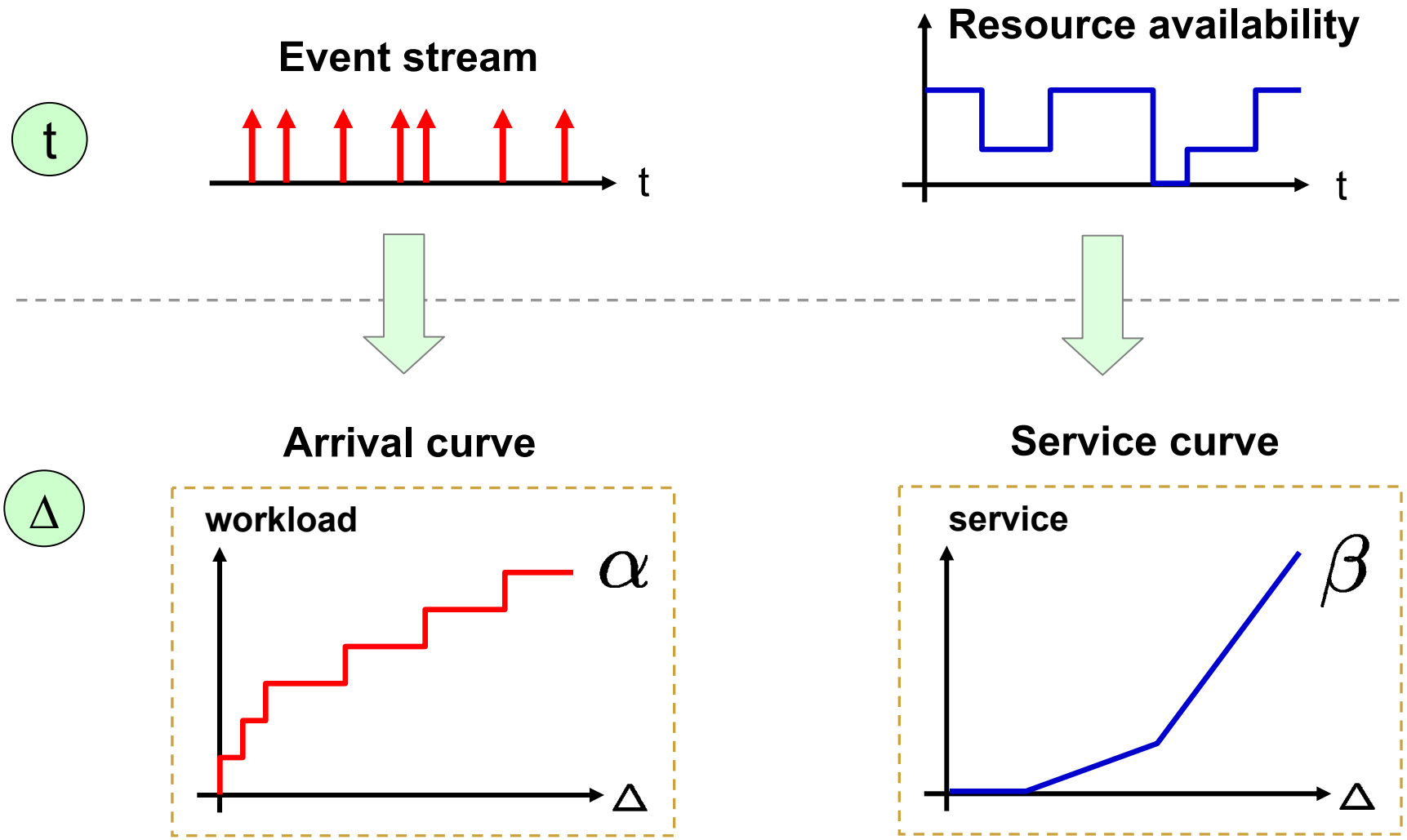


Outline

- Motivation
- Problem Definition
- Analysis Framework
- Proposed Algorithms
- Experimental Evaluation
- Conclusion

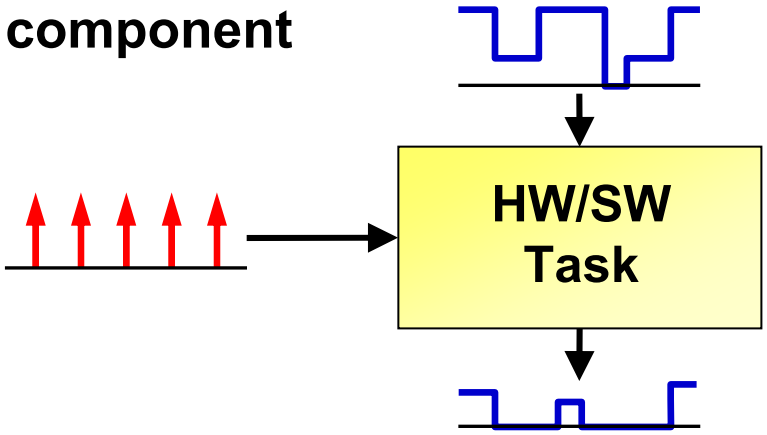
Real-Time Calculus (RTC)

[Thiele et al. 2000]



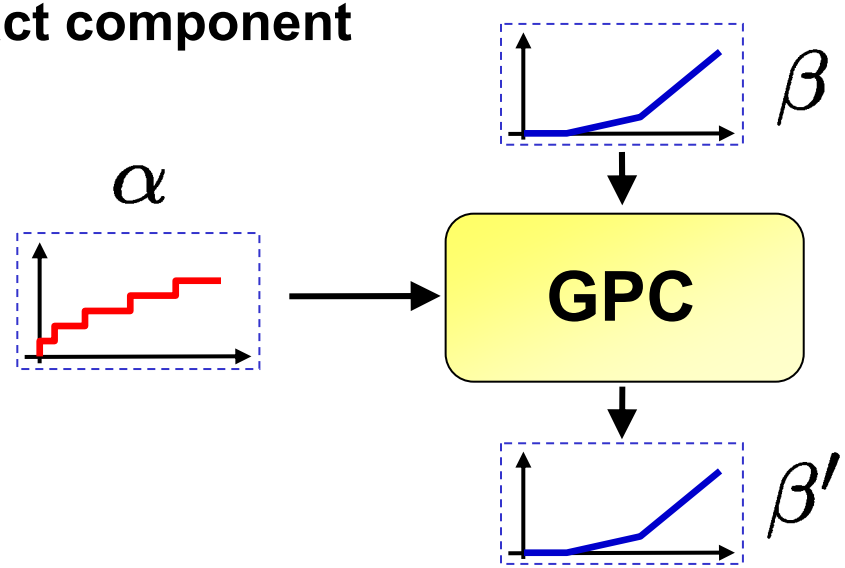
RTC: Greedy Processing Component

Concrete component



t

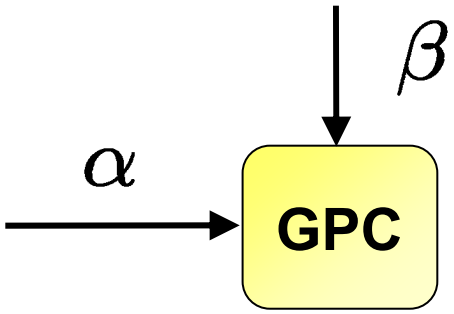
Abstract component

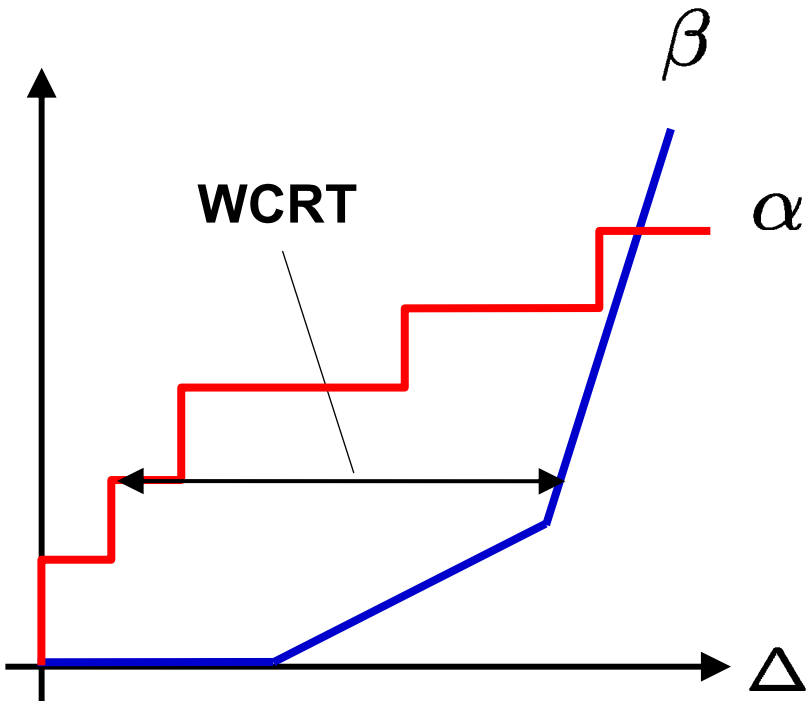


Δ

$$\beta'(\Delta) = f_{RT}(\alpha, \beta)$$

RTC: Scheduling Analysis



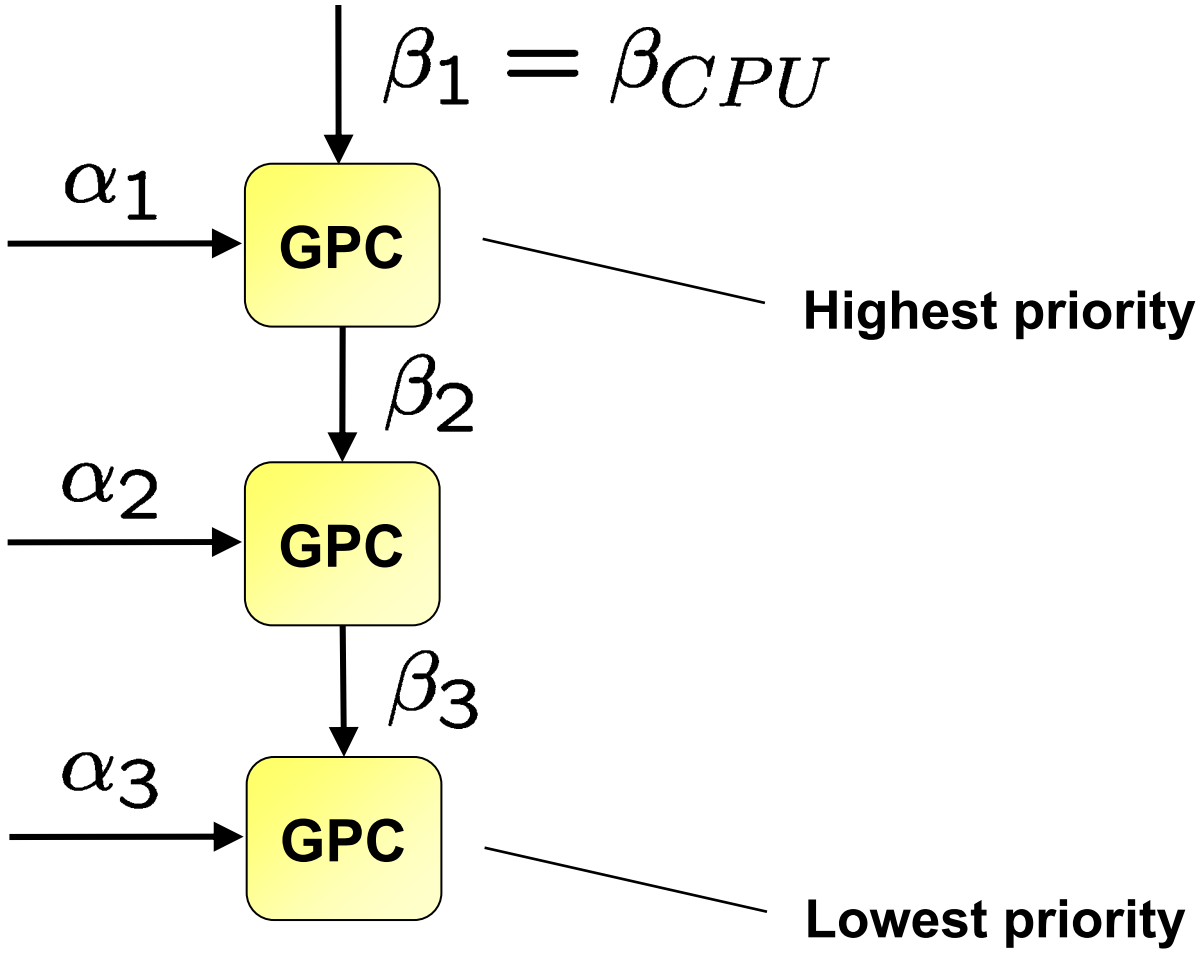


$$WCRT = f_{Del}(\alpha, \beta)$$

Task schedulable if

$$WCRT \leq D$$

RTC: Preemptive Fixed Priority Scheduling



Outline

- Motivation
- Problem Definition
- Analysis Framework
- Proposed Algorithms
- Experimental Evaluation
- Conclusion

Contributions

- Algorithm for static assignment of priorities
- Algorithm for static assignment of speeds
- Algorithm for combined optimization of priorities and speeds

Priority Assignment

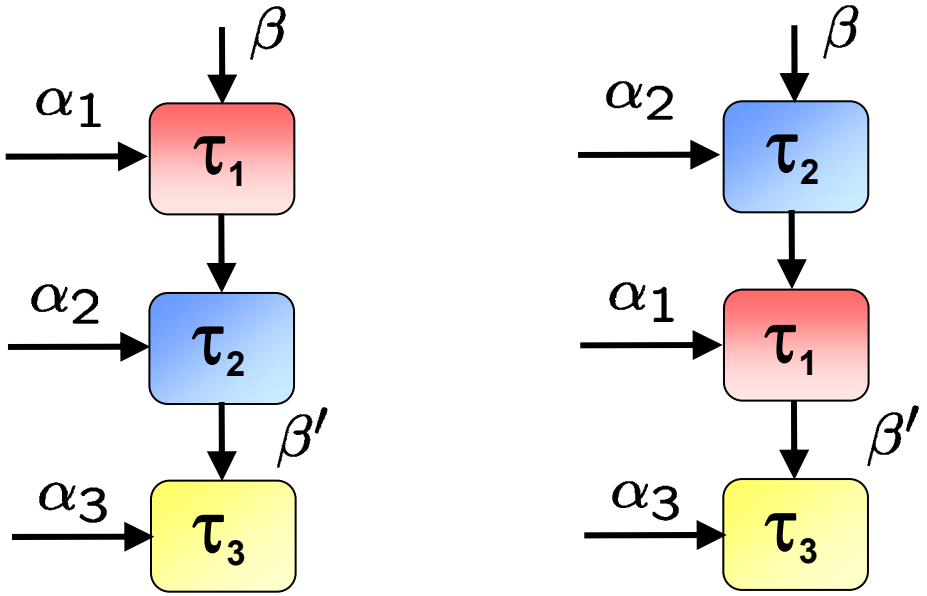
1) Find Π for given Σ

- Energy consumption is fixed
- Π affects schedulability

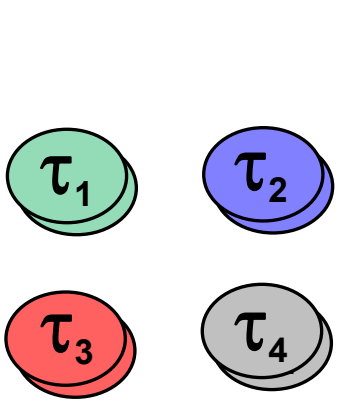
$\rightarrow$ Find Π under which Γ is schedulable, if it exists

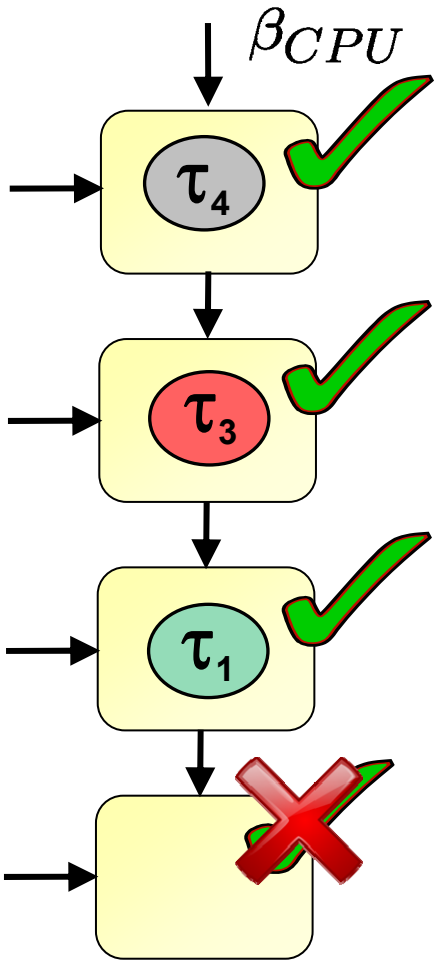
Observation:

Order of high-priority tasks does not affect low-priority task



Priority Assignment Algorithm





Optimal assignment strategy

Time complexity

$$O(N^3)$$

Brute-force algorithm

$$O(N!)$$

Minimum Global Speed

2) Find efficient (Π, Σ) under the restriction of equal speeds for all tasks

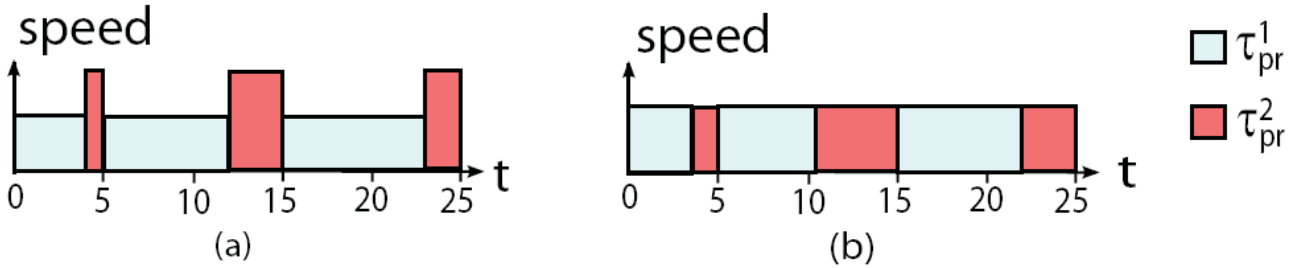
Binary search procedure

- Choose a global speed s_{glob}
- If there is no schedulable priority order increase s_{glob}
- Otherwise decrease s_{glob}
- Repeat until minimum speed is approximated with precision ϵ

Speed Assignment

3) Find Σ for given Π

- A global speed is usually not energy-efficient
- One should reduce individual speeds to obtain a *tight* assignment
- But how to choose the individual speeds?



→ **Intuitions:**

- Never run a low-priority task with a higher speed than a high-priority task
- Use as close speeds as possible

Speed Assignment

3) Find Σ for given Π

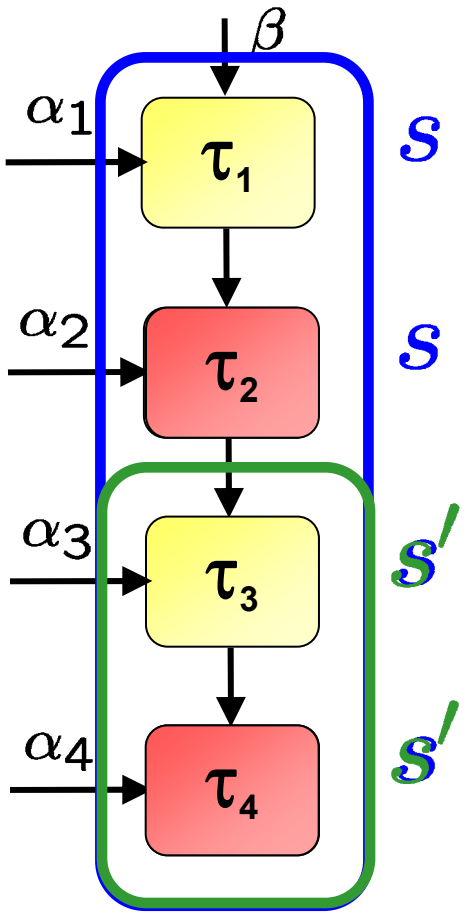
Proposal: Use speed assignments with the following properties:

$$s_{pr}^1 \geq s_{pr}^2 \geq \cdots \geq s_{pr}^N$$

$s_{pr}^{i-1} > s_{pr}^i$ **only if WCRT** $\tau_{pr}^{i-1} = D_{pr}^{i-1}$

Bottleneck Algorithm

3) Find Σ for given Π



- Assign minimum global speed
- Find bottleneck
- Fix speeds until bottleneck
- Repeat procedure for lower tasks
- Stop when last task becomes bottleneck

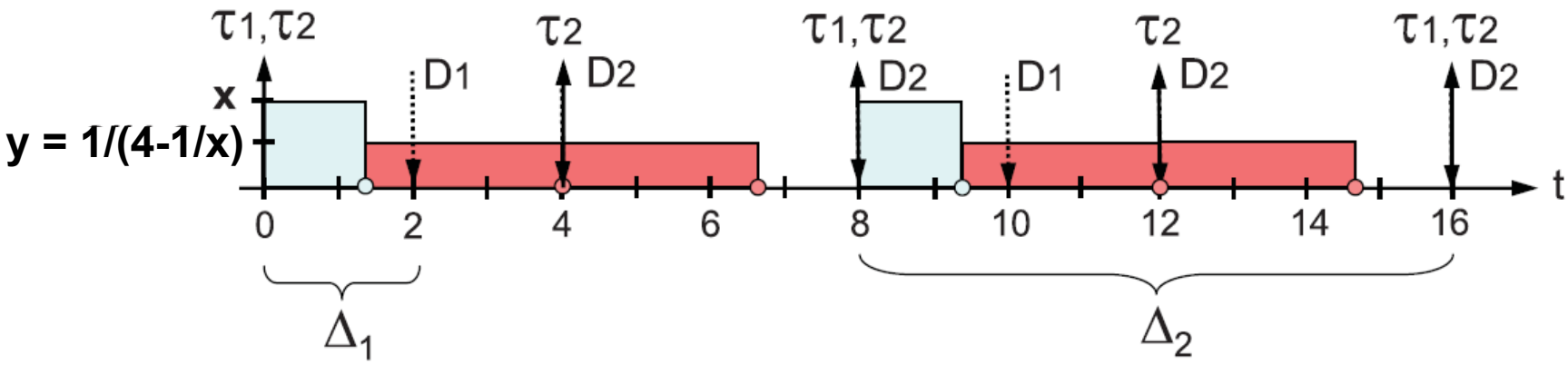
Time complexity:

$$O(N^2 \log \frac{1}{\epsilon})$$

Potential Non-Optimality

Example:

Speed assignment according to proposed strategy: x = 0.5, y = 0.5



x = 0.5
y = 0.5

Energy-optimal speeds

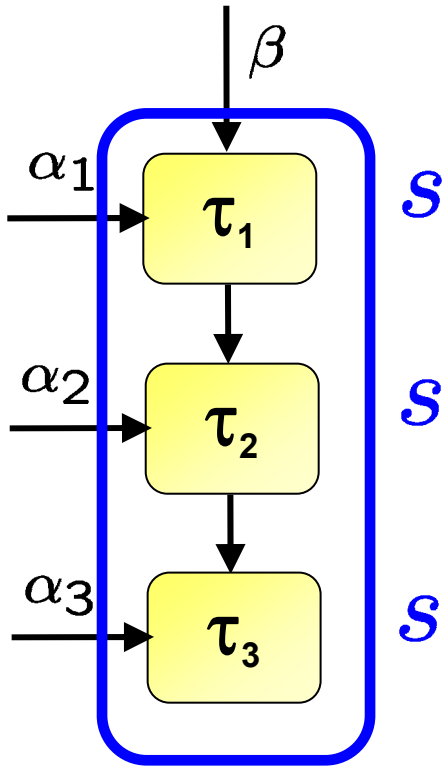
x = 0.565
y = 0.449

→ **As close speeds as possible are not always beneficial**

Combined Priority and Speed Assignment

4) Find Π and Σ (general case)

Combination of both algorithms:



At each iteration of the bottleneck algorithm do priority reordering

Time complexity:

$$O(N^4 \log \tfrac{1}{\epsilon})$$

Adaptations for Discrete Speeds

In the discrete case there is no notion of ‚bottleneck' task

(speed changes by arbitrary small ϵ are not possible)

$\rightarrow$ Repeat speed assignment at each priority level of the chain

		Brute-force algorithms
Speed assignment:	$O(N^2 \log M)$	$O(M^N)$
Combined assignment:	$O(N^4 \log M)$	$O(N! \cdot M^N)$

Outline

- Motivation
- Problem Definition
- Analysis Framework
- Proposed Algorithms
- Experimental Evaluation
- Conclusion

Experimental setup

9 test cases

10 tasks per test case

Periodic event streams with jitter

Intel XScale 1GHz

$$\mathcal{S}_{cont} = [0, 1]$$

$$\mathcal{S}_{discr} = \{0.15, 0.4, 0.6, 0.8, 1\}$$

$$P(s) = \hbar(0.08 + 1.52s^3)\ Watt$$

Parameters for test case No. 6 [ms]:

	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_8	τ_9	τ_{10}
p	14	28	15	30	28	7	25	25	30	22
j	20	35	3	19	35	5	30	20	0	15
d	2	2	0	0	4	0	5	0	0	0
C	1	1	1	1	1	1	2	1	1	2
D	21	1	53	406	276	41	6	342	4	181

$$\epsilon = 10^{-4}$$

RTC Toolbox for Matlab
www.mpa.ethz.ch

Results

Comparison of 3 different assignment policies for Π and Σ :

a) **Priorities for minimum global speed**

b) **Fixed priorities from a), Bottleneck algorithm for speeds**

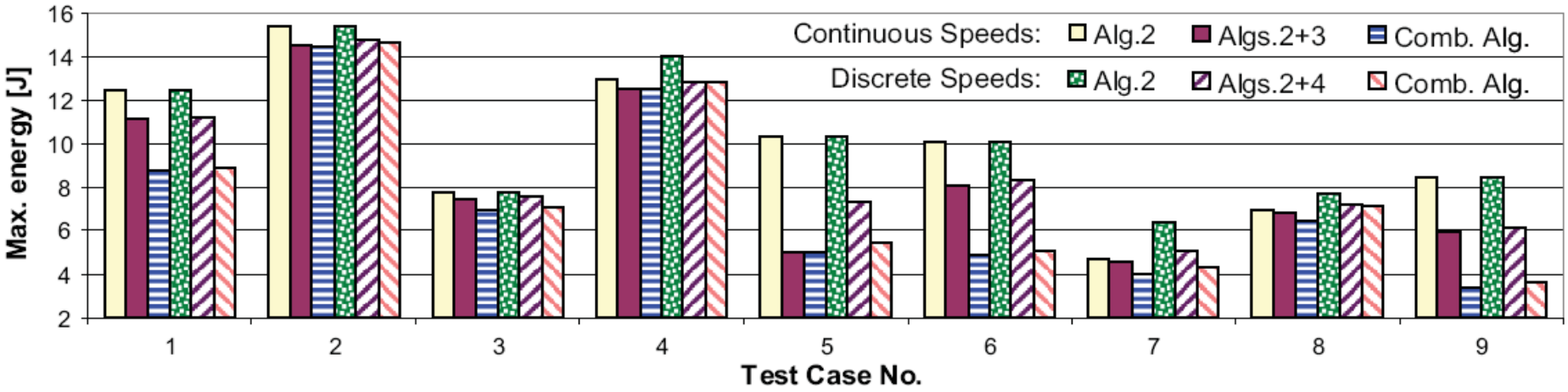
c) **Combined optimization of priorities and speeds**

Results for test case No. 6:

Policy \Priority	1	2	3	4	5	6	7	8	9	10
(a) Alg. 2	τ_2 1.000	τ_9 1.000	τ_7 1.000	τ_{10} 1.000	τ_8 1.000	τ_6 1.000	τ_1 1.000	τ_5 1.000	τ_4 1.000	τ_3 1.000
(b) Alg. 2 + 3	τ_2 1.000	τ_9 .9445	τ_7 .9445	τ_{10} .9445	τ_8 .9445	τ_6 .9445	τ_1 .9445	τ_5 .5776	τ_4 .5776	τ_3 .5776
(c) Combined	τ_2 1.000	τ_9 .7500	τ_7 .7500	τ_8 .5914	τ_1 .5914	τ_6 .5914	τ_3 .5914	τ_5 .5914	τ_4 .5914	τ_{10} .5914

Results

Simulated worst-case energy consumption for an execution of 10^4 ms:



Outline

- Motivation
- Problem Definition
- Analysis Framework
- Proposed Algorithms
- Experimental Evaluation
- Conclusion

Conclusion

- New methods for the design of energy-efficient real-time systems
- Static assignment of priorities and speeds to tasks with arbitrary non-deterministic release-times described by arrival curves
- Algorithms are based on two observations:
  1) Rearranging high-priority tasks does not affect low-priority tasks
  2) Use of priority-monotonic speeds is beneficial
- Combined optimization of priorities and speeds gives best results

Thank you!

Simon Perathoner

perathoner@tik.ee.ethz.ch