Improved feasibility of fixed-priority scheduling with deferred preemption using preemption thresholds for preemption points

Reinder J. Bril, Martijn M.H.P. van den Heuvel, and Johan J. Lukkien

Dep. of Mathematics and Computer Science
Group System Architecture and Networking

SAN

TU/e Technische Universiteit Eindhoven University of Technology

Where innovation starts

RTNS-2013 (Sophia Antipolis, France)

Background and motivation

- **Facts about fixed-priority scheduling (FPS):**
  - **described in standards, e.g. OSEK (Automotive);**
  - **supported by most COTS RTOS;**
  - **de facto standard in industry.**

- **Advantages of limited preemptive FPS:**
  - ***Reduced* memory requirements;**
  - ***Reduced* cost of arbitrary preemptions;**
  - ***Improved* schedulability of task sets.**

Background and motivation

- **Non-preemptive jobs (FPNS):**
  - ***Pro*: minimal memory requirements;**
  - ***Pro*: no preemption costs;**
  - ***Cons*: reduced schedulability.**

- **Two main alternatives:**

  1. Sequence of non-preemptive *sub-jobs* (FPDS):
  - ***Preemption points***

  ***2. Preemption thresholds* for *tasks* (FPTS).**

- **Schedulability: neither dominates the other.**

Technische Universiteit Eindhoven University of Technology

Background and motivation

- **FPDS:**
  - **highest schedulability ratio;**
  - **generates less preemptions;**
  - **most predictable for estimating preemption costs**
    - **because the number of preemptions and their positions are fixed and known from the code.**

- **Question: can we improve on FPDS?**

[8] Buttazzo, Bertogna, and Yao, IEEE TII, Feb. 2013.

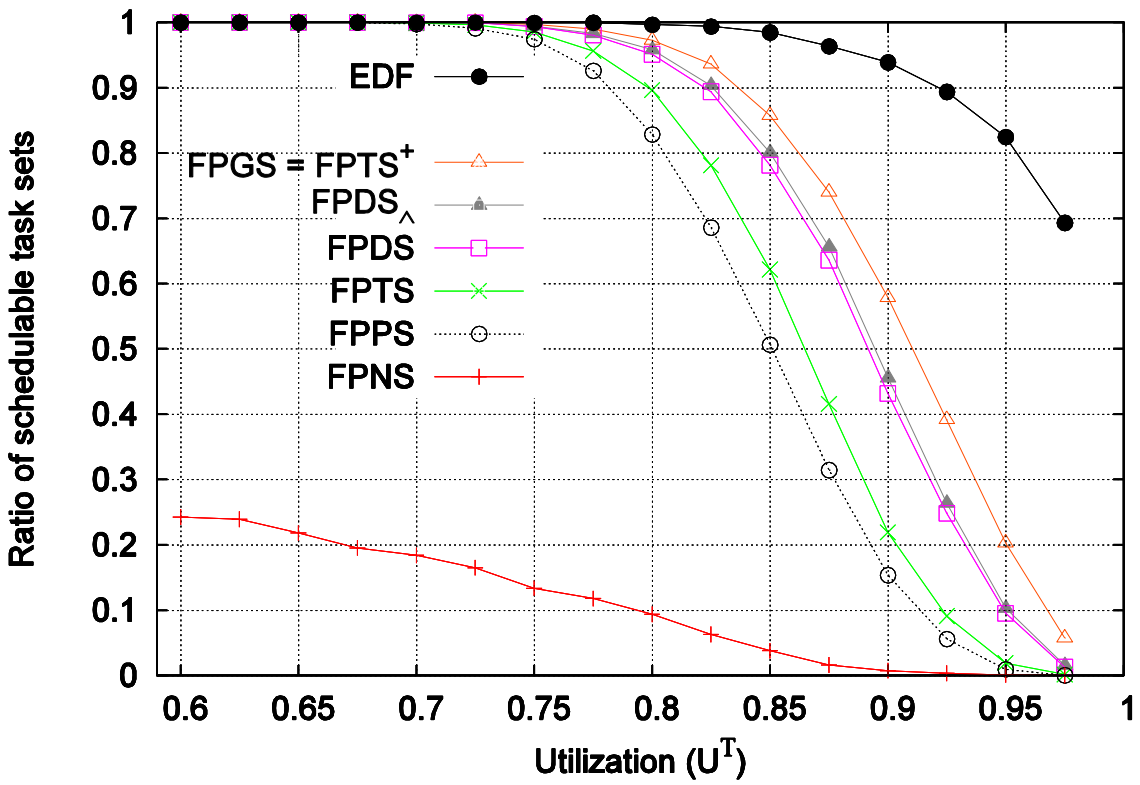
Background and motivation

- **Generalization of FPTS and FPDS (FPGS):**
  - **Preemption thresholds for *tasks* and per *sub-job***
    - ***Pro*: improved schedulability ratio;**
    - ***Cons*: arbitrary preemptions.**

- **Specialization of FPGS ($FPDS_{\wedge}$):**
  - **Preemption thresholds for *tasks* only**
    - ***Pro*: limits preemptions to preemption points;**
    - ***Cons*: limited schedulability ratio improvement.**

[6] R.J. Bril, M.M.H.P. v.d. Heuvel, U. Keskin, and J.J. Lukkien, ECRTS, 2012.

Background and motivation



8 tasks, 5.000 task sets,
$T_i \in$ [100, 10.000] (uniform), U_i by UUnifast ($\Rightarrow C_i$),
$D_i \in$ [0.5(T_i + C_i), T_i] (uniform);

Background and motivation

- **Goal:**
  - **Limited preemptive FPS scheme**
    - **schedulability ratio comparable to FPGS;**
    - **preemptions limited to preemption points.**

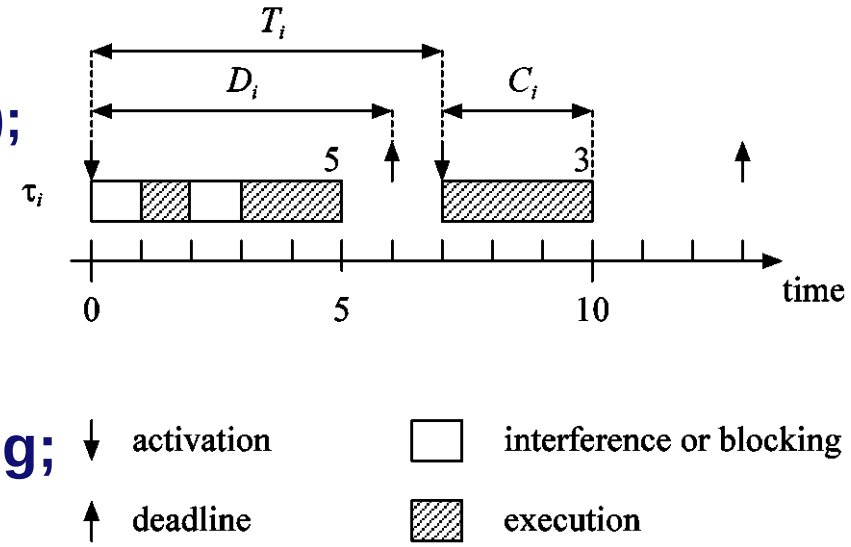
- **Approach:**
  - **Preemption thresholds per *preemption point*.**

Overview

- Background and motivation
- Real-time scheduling model (FPS)
  - Basic model
  - Enhanced model
  - Generalization graph
- An example
  - A schedulable configuration
  - Determining the configuration
- Contributions
- Conclusion

Basic FPS – model

- **Events: implicit**
- **Tasks (τ):**
  - **independent, no self-suspension**
  - **characteristics (R^+):**
    - **minimal inter-arrival time (T);**
    - **computation time (C);**
    - **deadline (D);**
- **Scheduling algorithm:**
  - **fixed-priority (π) & non-idling;**
  - **[non-] preemptive**
- **Platform: single CPU**



Enhanced FPS – model

- **Existing enhancements:**
  - **Job of τ_i is a sequence of m_i sub-jobs.**
  - **Preemption threshold**
    - **per task: θ_i, where $\pi_i \leq \theta_i \leq \pi_1$;**
    - **per sub-job: $\theta_{i,k}$, where $\pi_i \leq \theta_{i,k} \leq \pi_1$.**

- **Novel enhancement:**
  - **Preemption threshold**
    - **per *preemption point*: $\theta^{\bullet}_{i,k}$, where $\pi_i \leq \theta^{\bullet}_{i,k} \leq \pi_1$**
    - **superseding θ_i.**

Generalization graph for FPS algorithms

Job as a sequence of m_i sub-jobs.

	$m_i = 1$	$m_i \geq 1$		
		$m_i > 1 \Rightarrow \theta^{\bullet}_{i,a} = \pi_i$		
$\theta_{i,k} = \pi_i$	**FPPS** ←	**FPPS⁺**		
$\theta_{i,k} = \pi_1$	**FPNS** ←	**FPDS**		

[7] A. Burns, in *Advances in Real-Time Systems*, 1994

[5] R. Bril, J. Lukkien, and W. Verhaegh, ECRTS, 2007.

Technische Universiteit Eindhoven University of Technology

Generalization graph for FPS algorithms

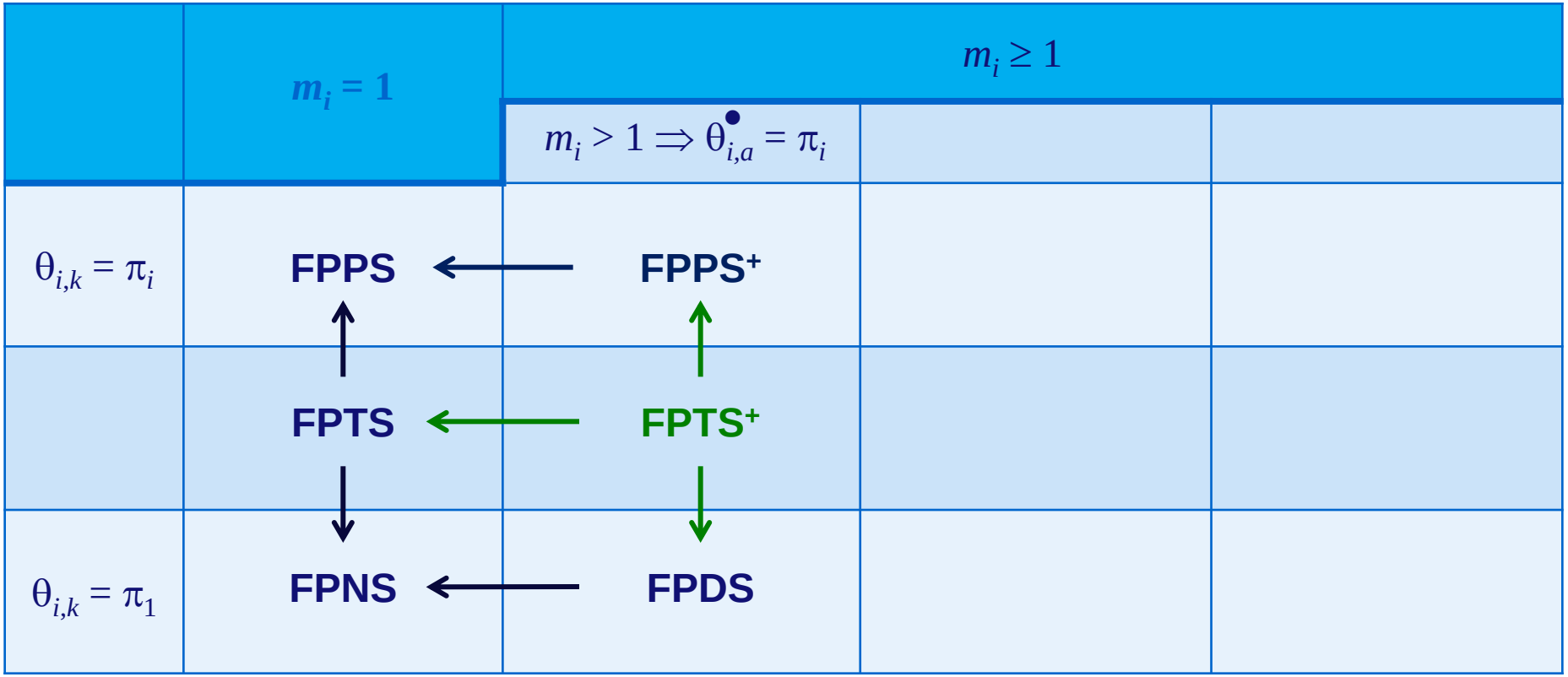
Preemption threshold per task.

	$m_i = 1$	$m_i \geq 1$		
		$m_i > 1 \Rightarrow \theta_{i,a} = \pi_i$		
$\theta_{i,k} = \pi_i$	**FPPS**	**FPPS⁺**		
	FPTS			
$\theta_{i,k} = \pi_1$	**FPNS**	**FPDS**		

[20] Y. Wang and M. Saksena, RTCSA, 1999.
[17] J. Regehr, RTSS, 2002.

Generalization graph for FPS algorithms

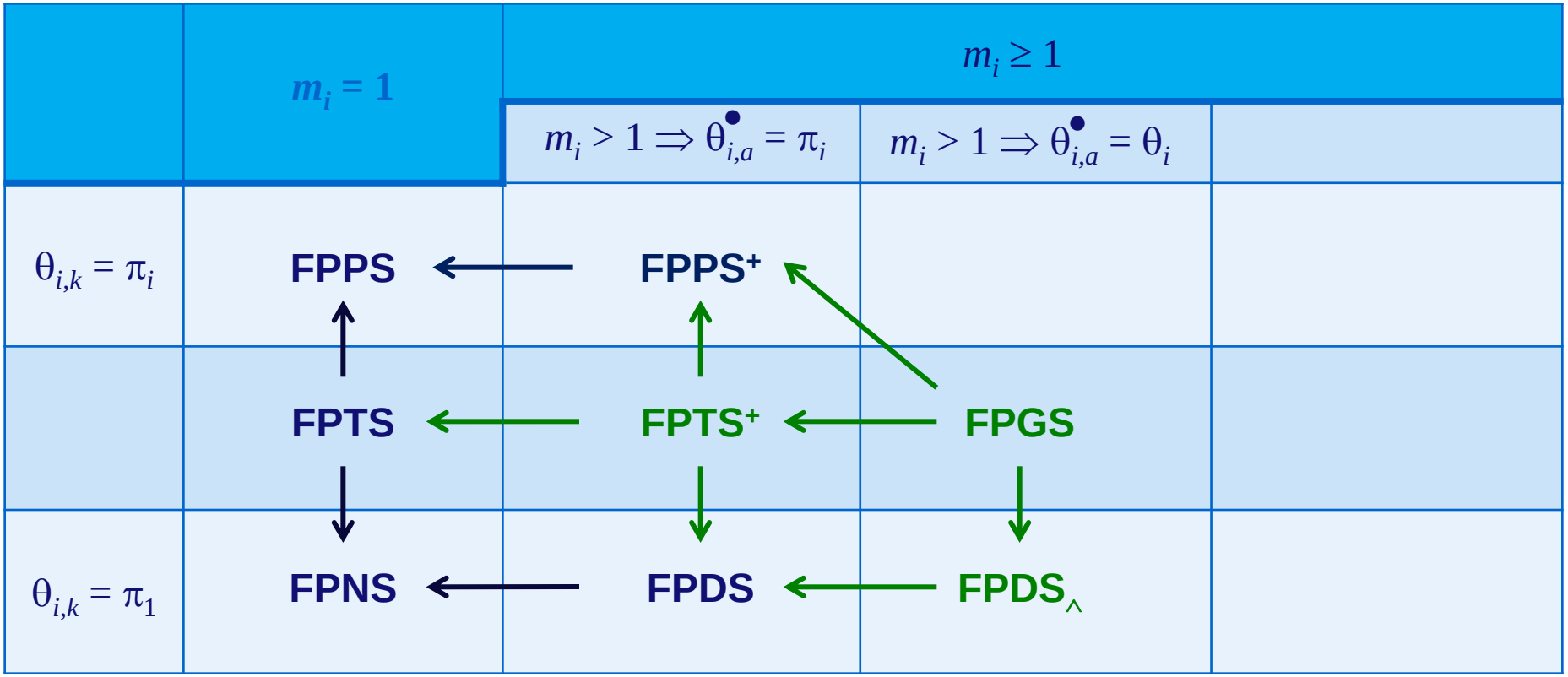
Preemption threshold per sub-job (only).



[12] U. Keskin, R.J. Bril, and J.J. Lukkien, ETFA, 2010.
[21] G. Yao and G. Buttazzo, EMSOFT, 2010.

Generalization graph for FPS algorithms

Preemption threshold per task and per sub-job.

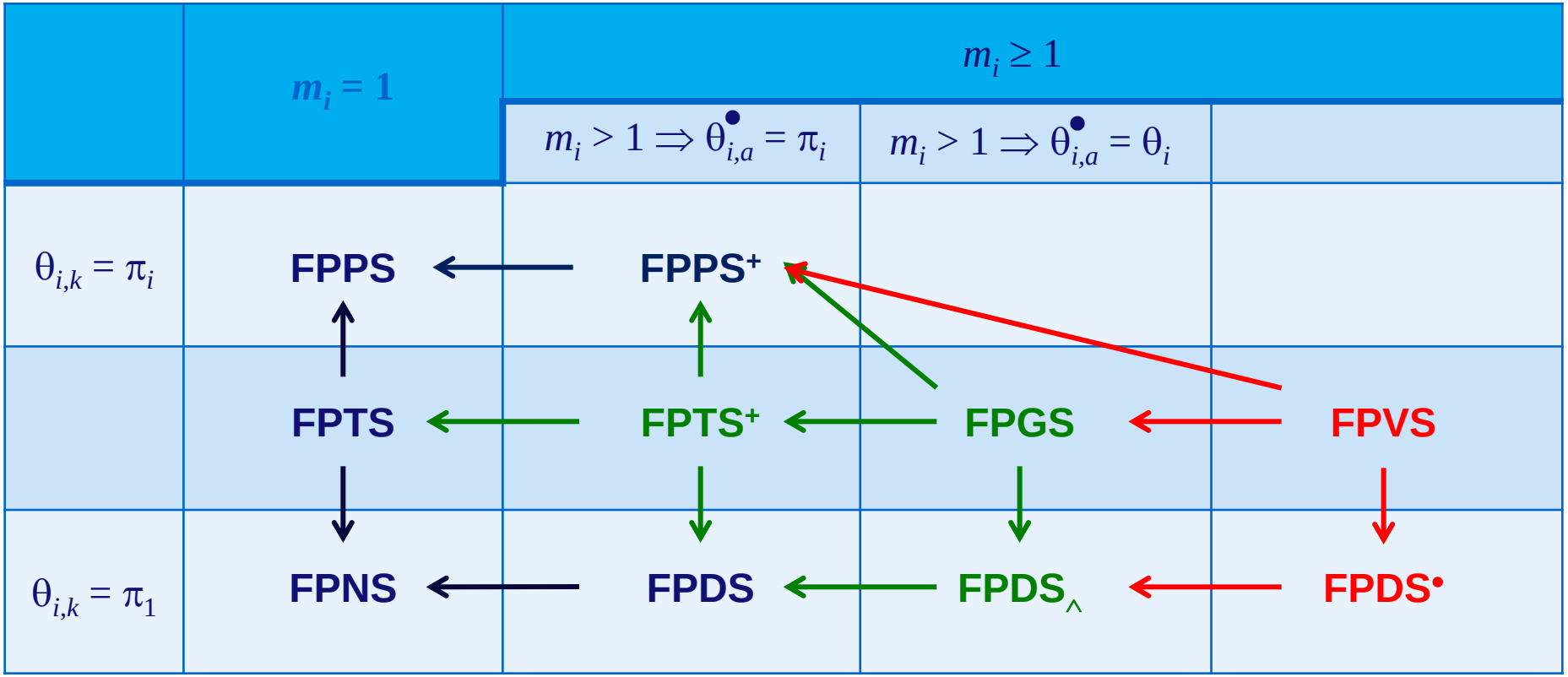


[6] R.J. Bril, M.M.P.H. v.d. Heuvel, U. Keskin, and J.J. Lukkien, ECRTS, 2012.

Technische Universiteit Eindhoven University of Technology

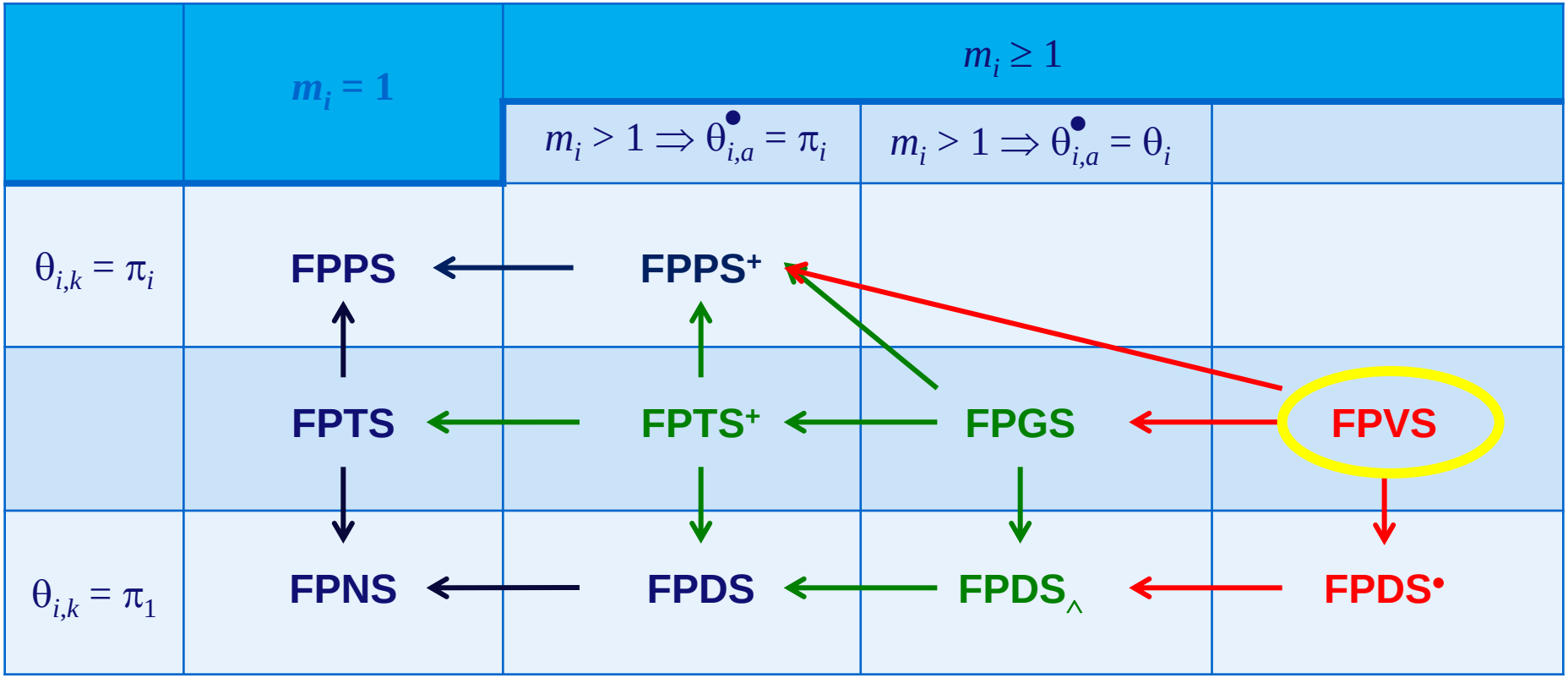
Generalization graph for FPS algorithms

Preemption threshold per sub-job and per preemption point.



[0] This presentation/paper.

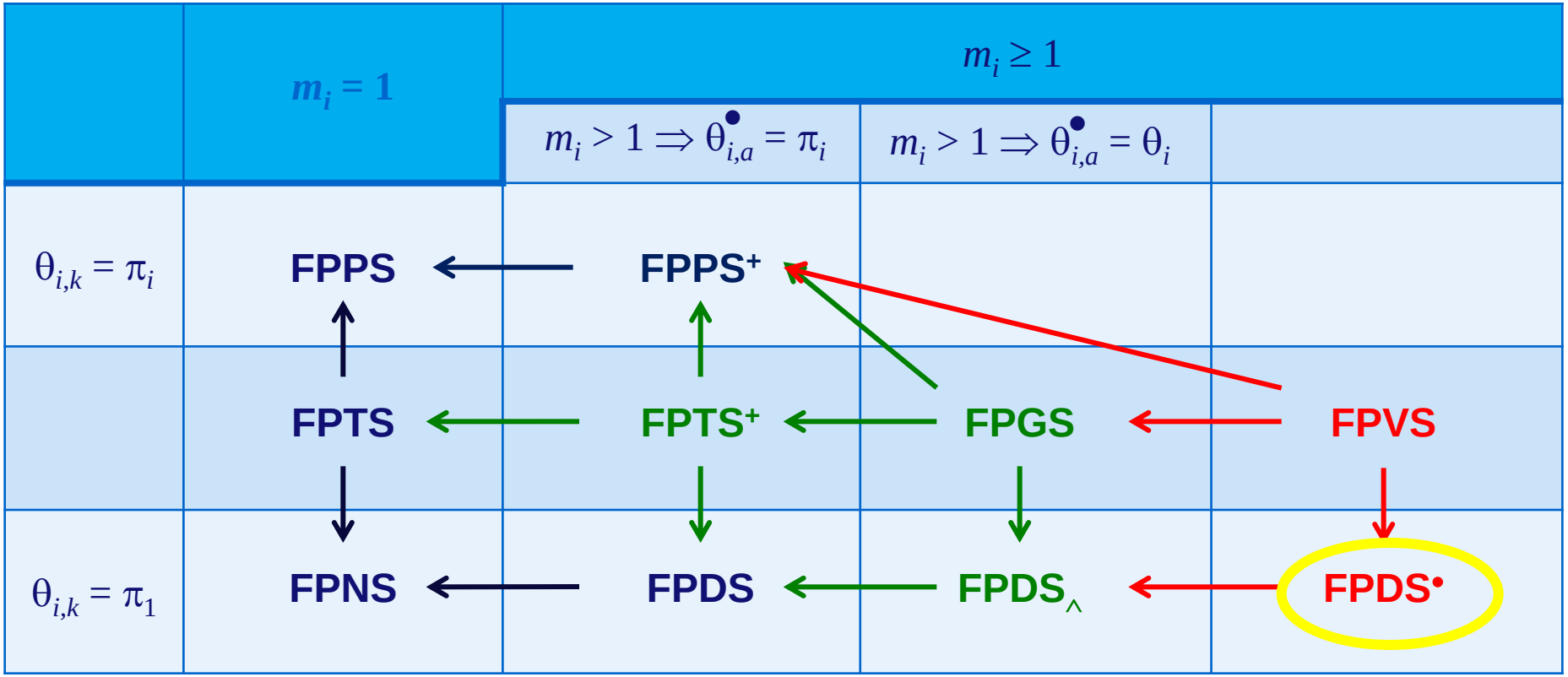
Generalization graph for FPS algorithms



FPVS: *Pro*: highest schedulability ratio;
Cons: potential of arbitrary preemptions;
Cons: potential of higher memory requirements.

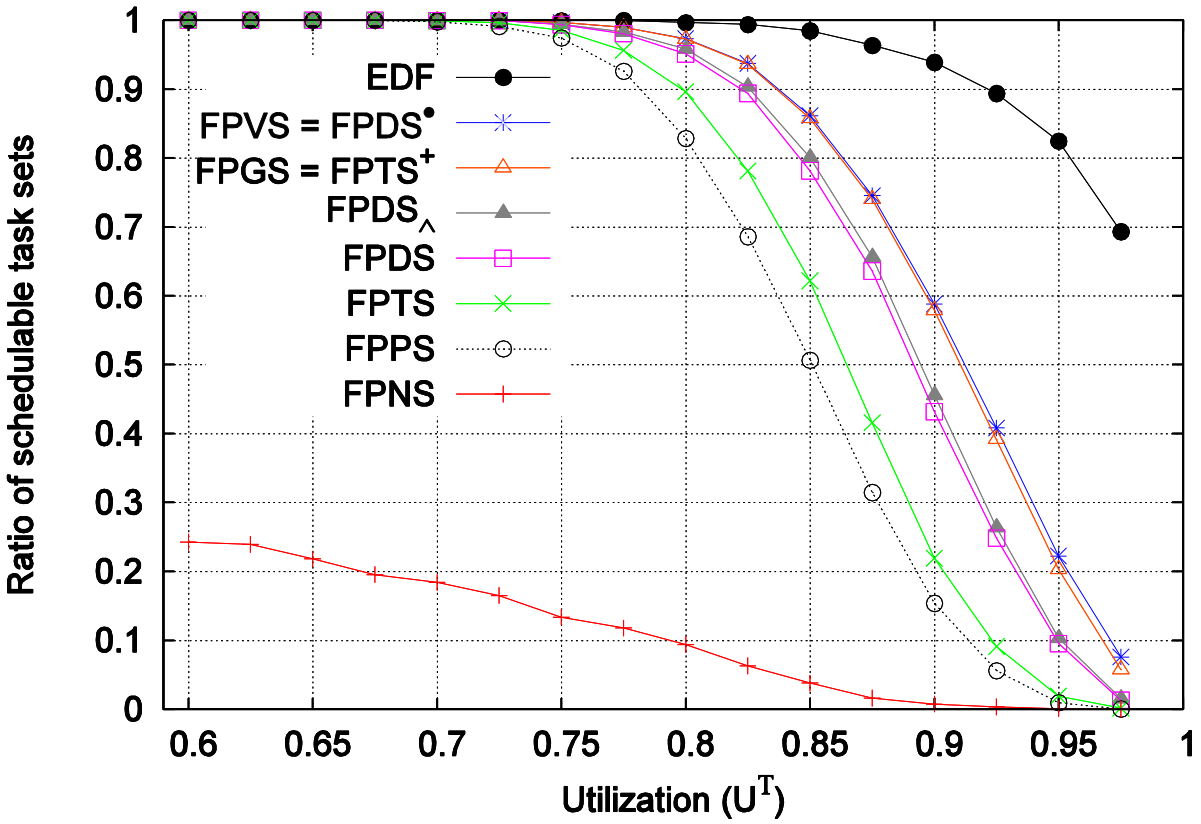
Technische Universiteit Eindhoven University of Technology

Generalization graph for FPS algorithms



FPDS•: *Pro*: high schedulability ratio;
Pro: predictable preemptions costs;
Pro: predictable memory requirements.

Evaluation



8 tasks, 5.000 task sets,
$T_i \in$ [100, 10.000] (uniform), U_i by UUnifast ($\Rightarrow C_i$),
$D_i \in$ [0.5(T_i + C_i), T_i] (uniform);

An example task set $\mathcal{T}$

Task	T_i	D_i	C_i	π_i	$C_{i,a}$	$\theta_{i,a}$	$\theta^{\bullet}_{i,a}$
τ_1	7	7	2	3	2	3	-
τ_2	23	23	8	2	3, 5	3, 3	2
τ_3	29	25	10	1	1, 4, 5	3, 3, 3	1, 2

- **Task set $\mathcal{T}$ is**
  - ***not schedulable* with any existing limited preemptive FPS scheme, not even under any other priority assignment.**
  - ***schedulable* with both FPVS and FPDS$^{\bullet}$.**

- **Conclusion:**
  - **FPVS strictly dominates all existing schemes.**

An example task set $\mathcal{T}$ – configuration

Task	T_i	D_i	C_i	π_i	$C_{i,a}$	$\theta_{i,a}$	$\dot{\theta}_{i,a}$
τ_1	7	7	2	3	2	3	-
τ_2	23	23	8	2			
τ_3	29	25	10	1			

- **Determining the configuration of task τ_1.**

An example task set $\mathcal{T}$ – configuration

Task	T_i	D_i	C_i	π_i	$C_{i,a}$	$\theta_{i,a}$	$\dot{\theta}_{i,a}$	β_i
τ_1	7	7	2	3	2	3	-	**5**
τ_2	23	23	8	2	3, **5**			
τ_3	29	25	10	1				

- **Determining the configuration of task τ_2.**

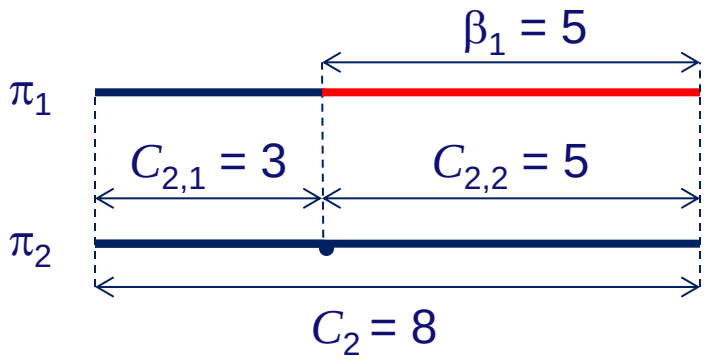
- ***Blocking tolerance*** **(β_i) [1]:**
  - **the maximum amount of time that a task (τ_i) can be blocked without missing its deadline (D_i).**

[1] V.B. Lortz and K.G. Shin, IEEE TSE, 1995.

An example task set $\mathcal{T}$ – towards analysis

Task	T_i	D_i	C_i	π_i	$C_{i,a}$	$\theta_{i,a}$	$\dot{\theta}_{i,a}$	β_i
τ_1	7	7	2	3	2	3	-	**5**
τ_2	23	23	8	2	3, **5**	3, 3	2	
τ_3	29	25	10	1				

- **Determining the configuration of task τ_2.**

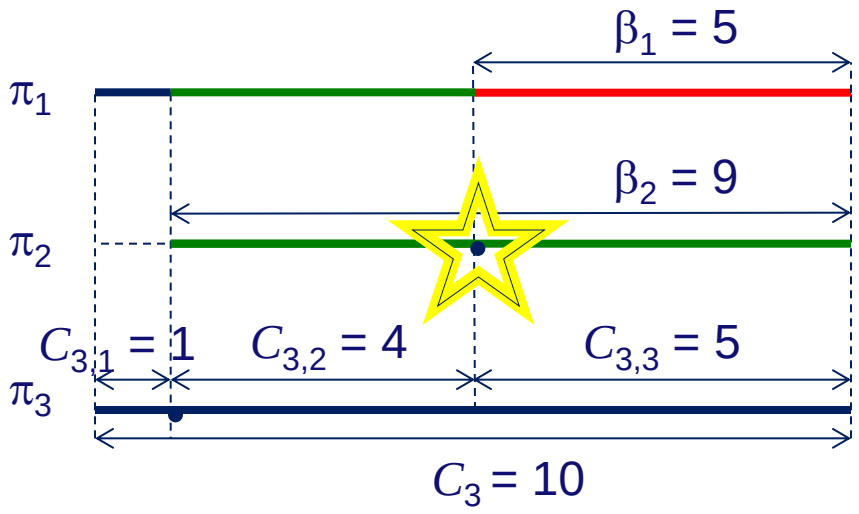


FPTS⁺ configuration
&
FPDS configuration

An example task set $\mathcal{T}$ – towards analysis

Task	T_i	D_i	C_i	π_i	$C_{i,a}$	$\theta_{i,a}$	$\theta^{\bullet}_{i,a}$	β_i
τ_1	7	7	2	3	2	3	-	**5**
τ_2	23	23	8	2	3, **5**	3, 3	2	**9**
τ_3	29	25	10	1	1, **4**, **5**	3, 3, 3	1, 2	1

- **Determining the configuration of task τ_3.**



FPVS configuration
&
FPDS• configuration

Contributions

1. **Novel scheduling algorithms: FPVS**
   - **Sub-jobs (similar to FPDS);**
   - **preemption thresholds**
     - **for sub-jobs and preemption points;**
   - **generalizes existing FPS algs.**
2. **Schedulability analysis for FPVS (similar to [11])**
   - **specializes to all existing FPS algs;**
3. **Algorithm to maximize schedulability under FPS:**
   - **given: T_i, D_i, C_i, and π_i;**
   - **determine: $C_{i,a}$, $\theta_{i,a}$, $\theta^{\bullet}_{i,a}$ (inspired by [8]).**
4. **Evaluation: FPVS dominates existing FPS algs.**

[11] M. González Harbour et al., RTSS, 1991.
[2] M. Bertogna, G.C. Buttazzo, G. Yao, RTSS, 2011.

Conclusion (significance of the result)

- **FPVS and FPDS•**
  - **have highest schedulability ratio;**
  - **can be emulated, e.g. with OSEK-standard (Automotive).**
- **FPDS• meets our goal:**
  - **Schedulability comparable to FPGS;**
  - **Preemptions limited to preemption points.**
- **If `schedulable`($\mathcal{T}$, FPDS) then FPDS• may have**
  - **reduced memory requirements;**
  - **tighter preemption cost estimation;**
  - **reduced power consumption.**

Technische Universiteit Eindhoven University of Technology