Generalized fixed-priority scheduling with limited preemptions

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

Dep. of Mathematics and Computer Science

Group System Architecture and Networking

SAN

TU/e Technische Universiteit Eindhoven University of Technology

ECRTS-2012 (Pisa, Italy)

Where innovation starts

Motivation

- **Drawbacks of arbitrary preemptions (FPPS):**
  - **worst-case memory requirements;**
  - **cost of context switching (e.g. cache).**
- **Non-preemptive scheduling (FPNS):**
  - **resolved by FPNS,**
  - **at the cost of lower schedulability.**
- **Alternative refinements of FPPS (FPTS & FPDS):**
  - **reduced memory costs (compared to FPPS);**
  - **improved schedulability (compared to FPPS);**
  - **orthogonal approaches !**

Technische Universiteit Eindhoven University of Technology

Motivation

- **Goal:**
  - **combine strength of FPTS and FPDS in a single scheme: FPGS,**
  - **with the aim to improve efficiency,**
  - **focus on improvement of the feasibility.**

Technische Universiteit **Eindhoven** University of Technology

Overview

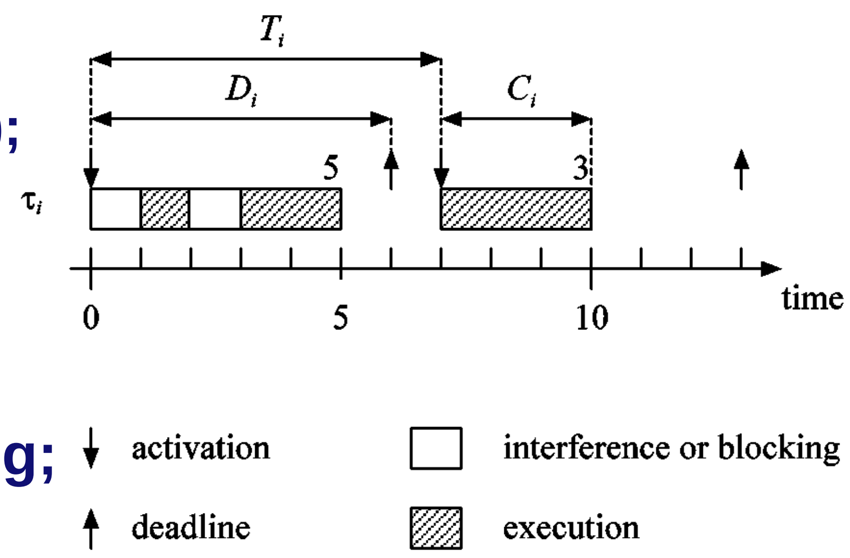
- Motivation
- Introduction fixed-priority scheduling (FPS)*
  - model
  - FPPS and FPNS
- Existing limited-preemptive algorithms*
  - preemption thresholds (FPTS)
  - deferred preemption (FPDS)
- Novel hybrid algorithms
- Conclusion

* Buttazzo, Bertogna, and Yao, IEEE TII, 2012.

Technische Universiteit Eindhoven University of Technology

Introduction 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**



Introduction FPS – FPPS and FNPS

- **FPPS:**
  - **highest priority task with work pending executes;**
  - **a task experiences *interference* from *higher* priority tasks (due to *delays* and *preemptions*).**

- **FPNS:**
  - **tasks run to completion;**
  - **the highest priority task is selected to run next;**
  - **a task experiences:**
    - ***interference* from *higher* priority tasks (only *delays*);**
    - ***blocking* from *lower* priority tasks.**

TU/e Technische Universiteit Eindhoven University of Technology

Introduction FPS – FPPS and FPNS

- **FPNS:**
  - **blocking:** $B_i = \max(0, \max_{l:\pi_i > \pi_l} C_l)$
- ***Blocking tolerance* (β_i) [1]:**
  - **the maximum amount of time that a task (τ_i) can be blocked without missing its deadline (D_i);**
  - **depends on scheduling algorithm.**
- **Neither FPPS dominates FPNS nor vice versa.**

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

Technische Universiteit Eindhoven University of Technology

Existing limited-preemptive algorithms

- **Two orthogonal approaches: FPTS [2, 3]** and FPDS

  - **FPTS:** ***preemption threshold*** **($\theta_i \geq \pi_i$)**
    - **interference: (reduce *preemptions*)**
      - **tasks τ_h with $\pi_h > \theta_i$ can preempt τ_i;**
    - **blocking:** $B_i = \max(0, \max_{l:\theta_l \geq \pi_i > \pi_l} C_l)$

  - **Special cases of FPTS:**
    - **FPPS: $\theta_i = \pi_i$;**
    - **FPNS: $\theta_i = \pi_1$.**

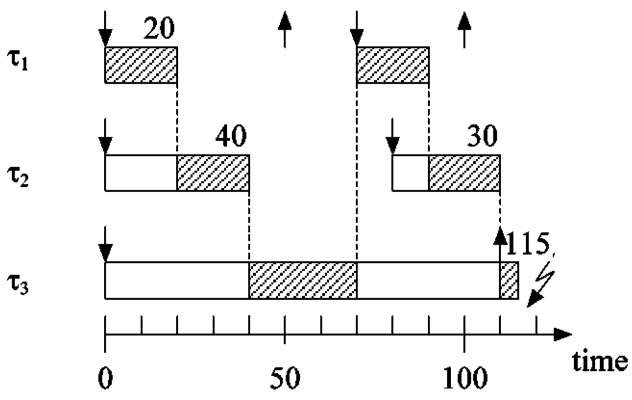
[2] Y. Wang and M. Saksena, RTCSA, 1999.
[3] J. Regehr, RTSS, 2002.

FPTS – preemption thresholds

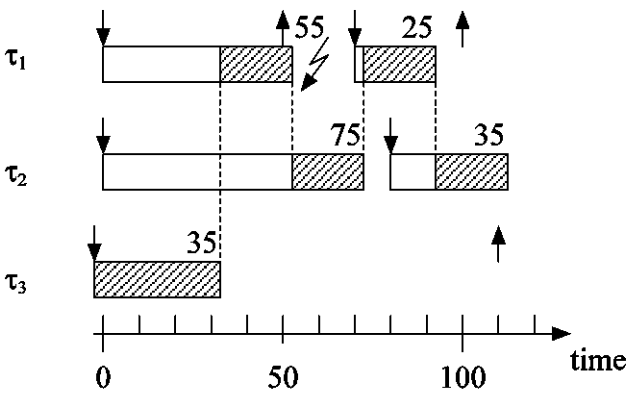
	T_i	D_i	C_i	π_i	WR_i^P	WR_i^N
τ_1	70	50	20	3	20	***55***
τ_2	80	80	20	2	40	75
τ_3	200	100	35	1	***115***	75

Blocking tolerance τ_1:
- $\beta_1 = 30$, $C_2 < \beta_1 < C_3$.

Blocking tolerance τ_2:
- FPPS: $\beta_2^P = 30 < C_3$;
- FPNS: $\beta_2^N = 40 > C_3$.



Not schedulable with FPPS



Not schedulable with FPNS

FPTS – preemption thresholds

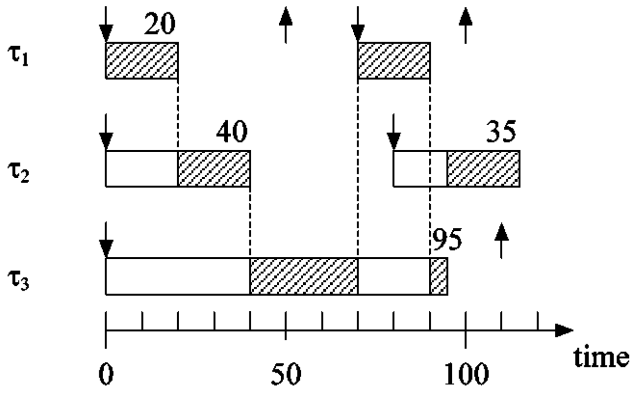
	T_i	D_i	C_i	π_i	WR_i^P	WR_i^N	θ_i	WR_i^T
τ_1	70	50	20	3	20	***55***	3	40
τ_2	80	80	20	2	40	75	3	75
τ_3	200	100	35	1	***115***	75	2	95

Blocking tolerance τ_1:

- $\beta_1 = 30$, $C_2 < \beta_1 < C_3$.

Blocking tolerance τ_2:

- FPPS: $\beta_2^P = 30 < C_3$;
- FPNS: $\beta_2^N = 40 > C_3$.



Schedulable with FPTS

Existing limited-preemptive algorithms

- **Two orthogonal approaches: FPTS and FPDS [4, 5]**

  - **FPDS: *deferred preemption***
    - **a task is a sequence of m_i non-preemptive *sub-tasks*;**
    - **characteristic subtask $\tau_{i,k}$: computation time $C_{i,k}$;**
    - **tasks τ_h with $\pi_h > \pi_i$ can only preempt τ_i at *preemption points* (between subtasks);**
    - **blocking:** $B_i = \max(0, \max_{l:\pi_i > \pi_l} \max_{1 \le k \le m_l} C_{l,k})$

  - **Special case of FPDS:**
    - **FPNS: $m_i = 1$.**

[4] A. Burns, in *Advances in Real-Time Systems*, 1994
[5] R. Bril, J. Lukkien, and W. Verhaegh, ECRTS, 2007.

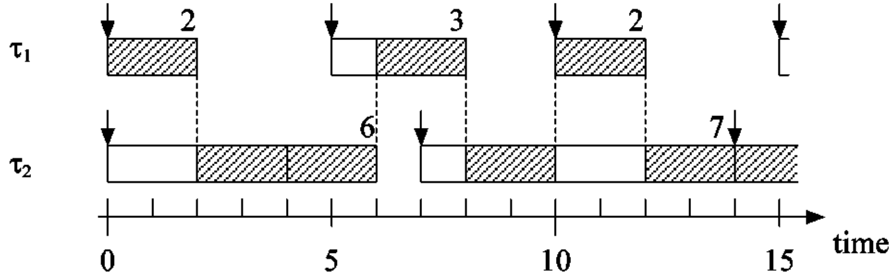
TU/e Technische Universiteit Eindhoven University of Technology

FPDS – deferred preemption

	$T_i = D_i$	C_i	π_i	WR_i^P	WR_i^N	WR_i^D
τ_1	5	2	2	2	***6***	4
τ_2	7	2+2	1	***8***	6	7

Blocking tolerance τ_1:

- $\beta_1 = 3 < C_3$.



Schedulable with FPDS

FPDS – deferred preemption

	$T_i = D_i$	C_i	π_i	WR_i^P	WR_i^N	WR_i^D
τ_1	5	2	2	2	***6***	4
τ_2	7	2+2	1	***8***	6	7

Blocking tolerance τ_1:

- $\beta_1 = 3 < C_3$.

- **FPTS:**
  - **$\theta_2 = \pi_2$: FPPS**
  - **$\theta_2 = \pi_1$: FPNS**
- **Conclusion:**
  - **Not schedulable with FPTS, hence**
  - **FPTS does not dominate FPDS**

Technische Universiteit **Eindhoven** University of Technology

FPDS does not dominate FPTS

- **Previous slide:**
  - **FPTS does not dominate FPDS;**
- **Without example (space & time reasons):**
  - **FPDS does not dominate FPTS**
- **Conclusion:**
  - **neither FPDS dominates FPTS nor vice versa**

Novel hybrid algorithms

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

Generalization graph for FPS algorithms

Novel hybrid algorithms – model

- **FPTS+ [6, 7]**
  - **a task is a sequence of m_i sub-tasks;**
  - **each subtask $\tau_{i,k}$ has a preemption threshold $\theta_{i,k}$;**
  - **a task has no (longer a) preemption threshold;**
  - **blocking:** $B_i = \max(0, \max_{l:\pi_i > \pi_l} \max_{1 \le k \le m_l : \theta_{l,k} \ge \pi_i} C_{l,k})$

- **Special cases for FPPS+:**
  - $m_i = 1$**: FPTS;**
  - $\theta_{i,k} = \pi_i$**: FPPS+;**
  - $\theta_{i,k} = \pi_1$**: FPDS.**

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

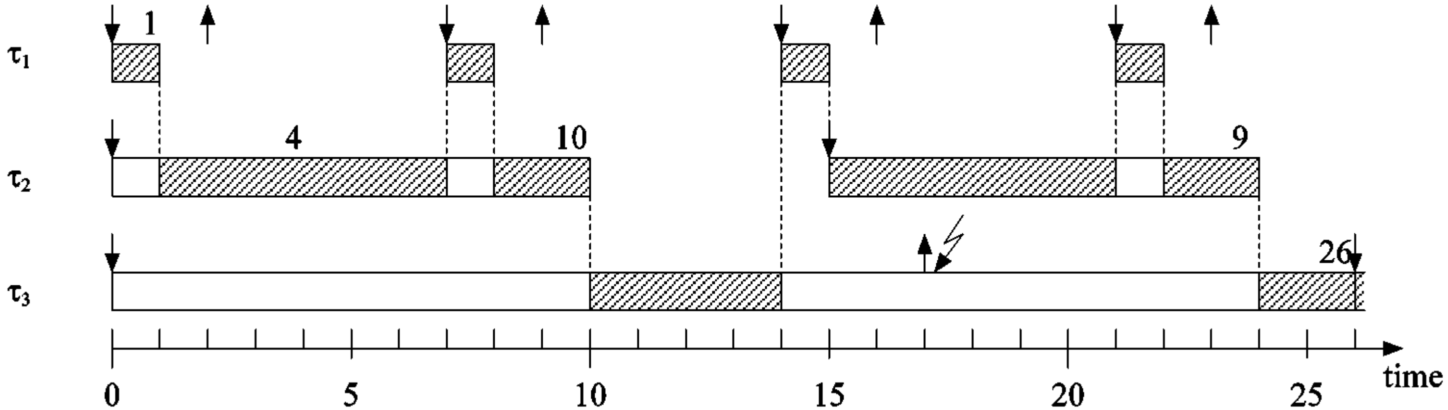
Technische Universiteit Eindhoven University of Technology

FPTS⁺ – preemption thresholds

	T_i	D_i	C_i	π_i	WR_i^P	WR_i^N
τ_1	7	2	1	3	1	***9***
τ_2	15	15	7+1	2	10	15
τ_3	26	17	1+5	1	***26***	16

Blocking tolerance τ_1:

- $\beta_1 = 1$, $C_3 > \beta_1$, $C_2 > \beta_1$.



Blocking tolerance $\beta_1 = 1 \Rightarrow$ **Not** schedulable with FPDS.

FPTS⁺ – preemption thresholds

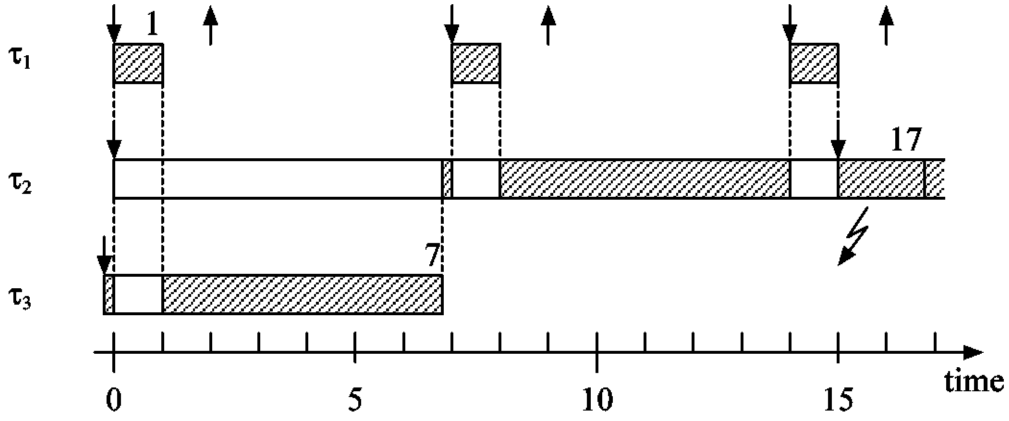
	T_i	D_i	C_i	π_i	WR_i^P	WR_i^N
τ_1	7	2	1	3	1	***9***
τ_2	15	15	7+1	2	10	15
τ_3	26	17	1+5	1	***26***	16

Blocking tolerance τ_1:

- $\beta_1 = 1$, $C_3 > \beta_1$, $C_2 > \beta_1$.

Blocking tolerance τ_2:

- FPPS: $\beta_2^P = 4 < C_3$;



Blocking tolerance $\beta_2 < C_3 \Rightarrow$ **Not** schedulable with FPTS.

FPTS⁺ – preemption thresholds

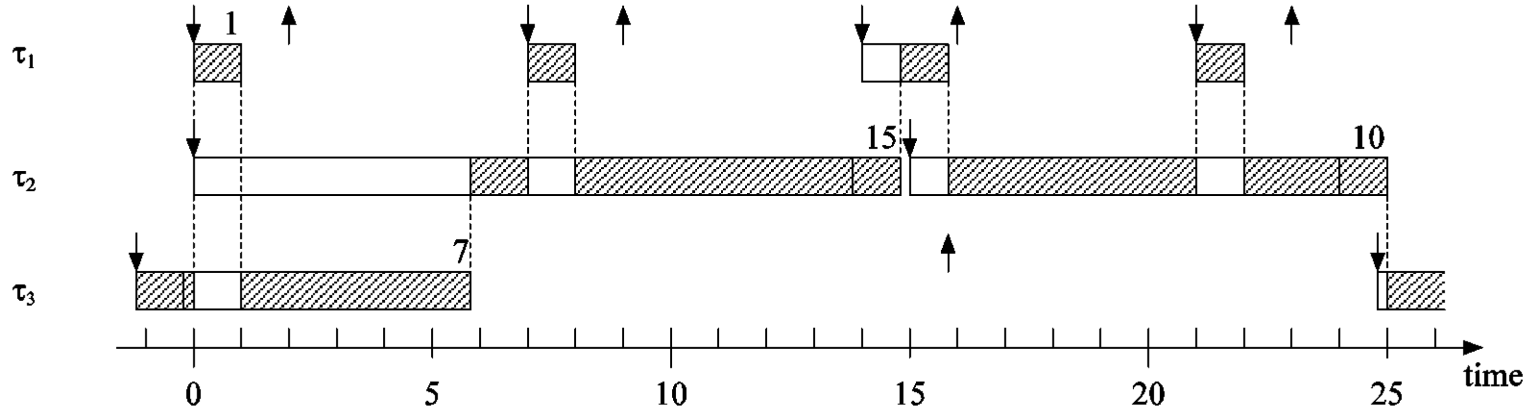
	T_i	D_i	C_i	π_i	WR_i^P	WR_i^N	$\theta_{i,1}$	$\theta_{i,2}$	WR_i^{T+}
τ_1	7	2	1	3	1	***9***	3	-	2
τ_2	15	15	7+1	2	10	15	2	3	15
τ_3	26	17	1+5	1	***26***	16	1	2	17

Blocking tolerance τ_1:

- $\beta_1 = 1$, $C_3 > \beta_1$, $C_2 > \beta_1$.

Blocking tolerance τ_2:

- FPPS: $\beta_2^P = 4 < C_3$;
- FPTS⁺: $\beta_2^{T+} = 5$



FPTS$^+$ – preemption thresholds

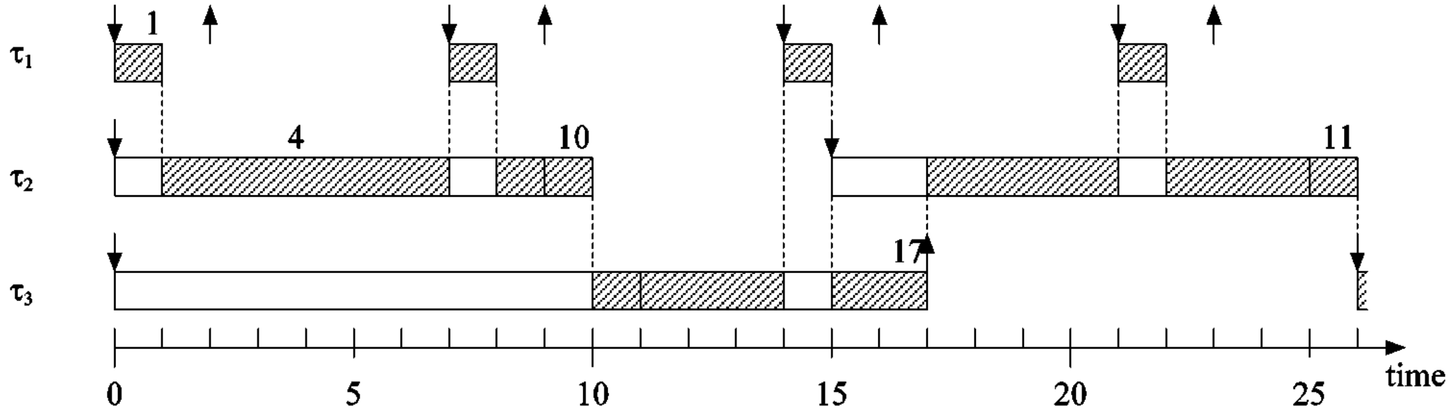
	T_i	D_i	C_i	π_i	WR_i^P	WR_i^N	$\theta_{i,1}$	$\theta_{i,2}$	WR_i^{T+}
τ_1	7	2	1	3	1	***9***	3	-	2
τ_2	15	15	7+1	2	10	15	2	3	15
τ_3	26	17	1+5	1	***26***	16	1	2	17

Blocking tolerance τ_1:

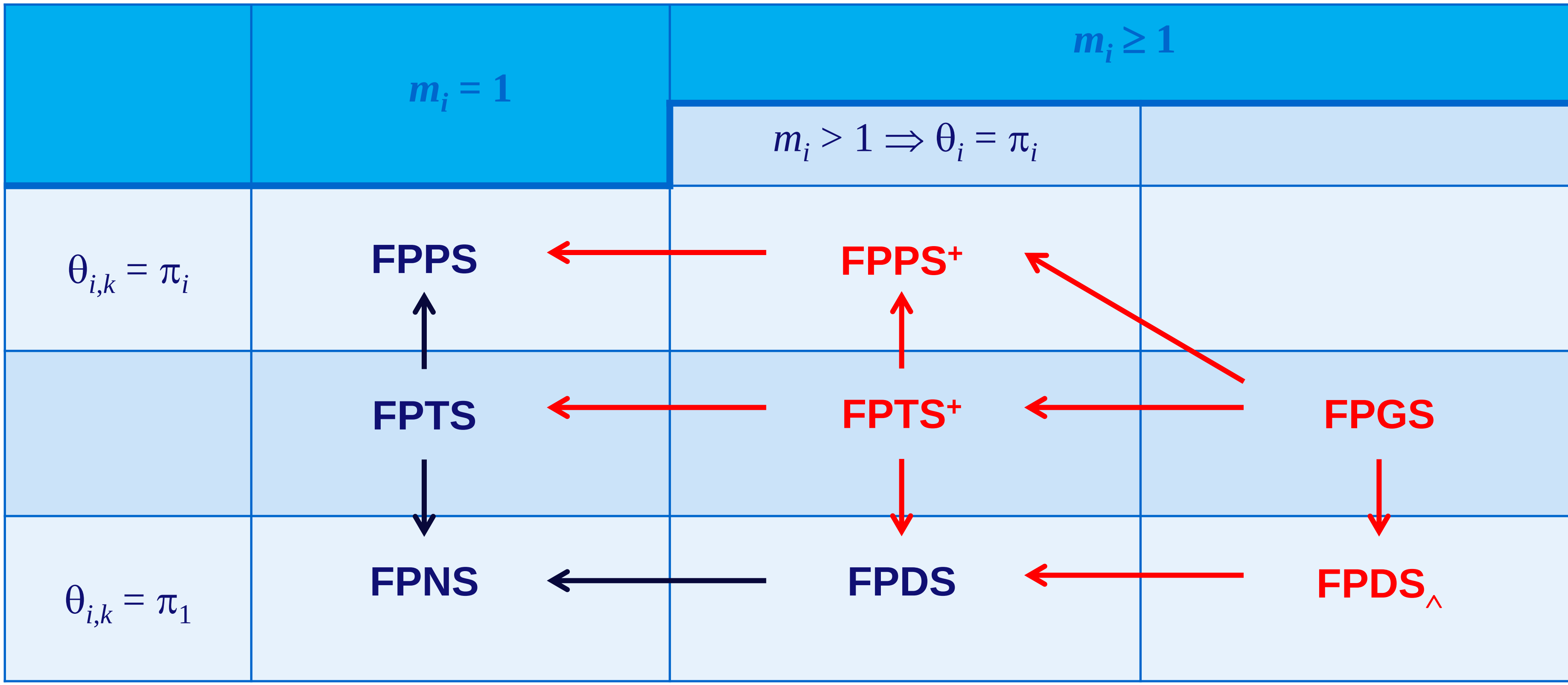
- $\beta_1 = 1$, $C_3 > \beta_1$, $C_2 > \beta_1$.

Blocking tolerance τ_2:

- FPPS: $\beta_2^P = 4 < C_3$;
- FPTS$^+$: $\beta_2^{T+} = 5$



Novel hybrid algorithms



Generalization graph for FPS algorithms

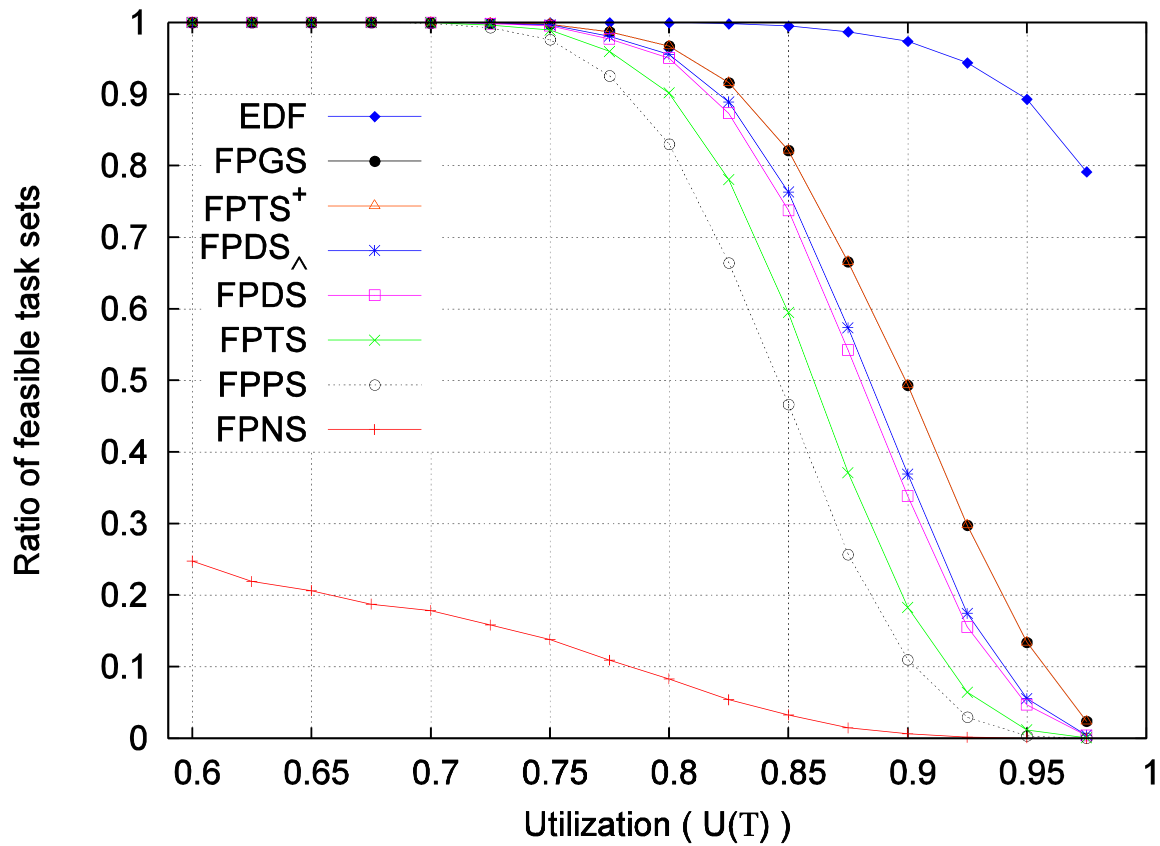
Novel hybrid algorithms – results

1. **Novel scheduling algorithms: FPGS**
   - **subtasks (similar to FPDS);**
   - **preemption thresholds for tasks (FPTS) and subtasks;**
   - **generalizes existing FPS algs.**
2. **Schedulability analysis for FPGS**
   - **specializes to all existing FPS algs;**
3. **Algorithm to maximize schedulability under FPS:**
   - **given: T_i, D_i, C_i, and π_i;**
   - **determine: $C_{i,mi}$, θ_i, $\theta_{i,mi}$ (inspired by [8]).**
4. **Evaluation: FPGS dominates existing FPS algs.**

[8] M. Bertogna, G.C. Buttazzo, G, Yao, RTSS, 2011.

TU/e Technische Universiteit Eindhoven University of Technology

Novel hybrid algorithms – evaluation



10 tasks, 10.000 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);

Conclusion

- **FPGS and existing FPS algorithms:**
  - **FPGS generalizes all others;**
  - **analysis of FPGS specializes to all others;**
  - **FPGS dominates all others.**

- **Future work:**
  - **further improvements of schedulability:**
    - **preemption thresholds for *preemption points*;**
  - **context switching cost;**
  - **...**

TU/e Technische Universiteit Eindhoven University of Technology