# An Empirical Investigation of Eager and Lazy Preemption Approaches in Global Limited Preemptive Scheduling

*Abhilash Thekkilakattil, Kaiqian Zhu, Yonggao Nie, Radu Dobrin and Sasikumar Punnekkat*

MÄLARDALEN UNIVERSITY SWEDEN

Vetenskapsrådet

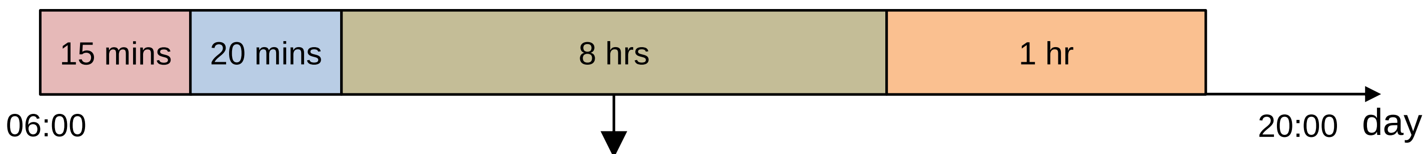
# We do Scheduling Everyday!

15 mins

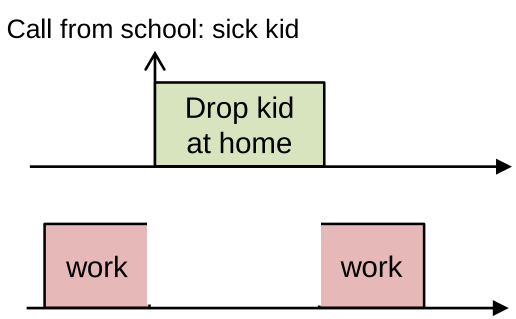
20 mins

8 hrs

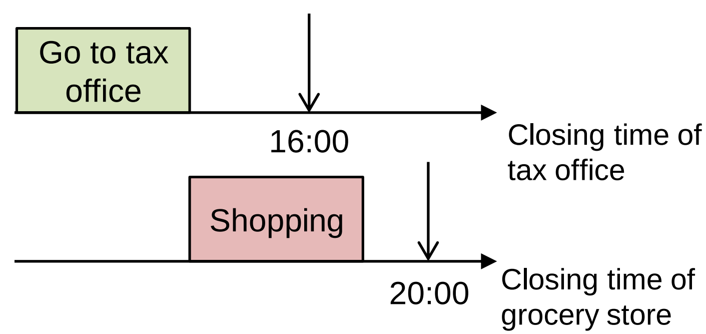
1 hr




## Priority based scheduling




## Deadline based scheduling

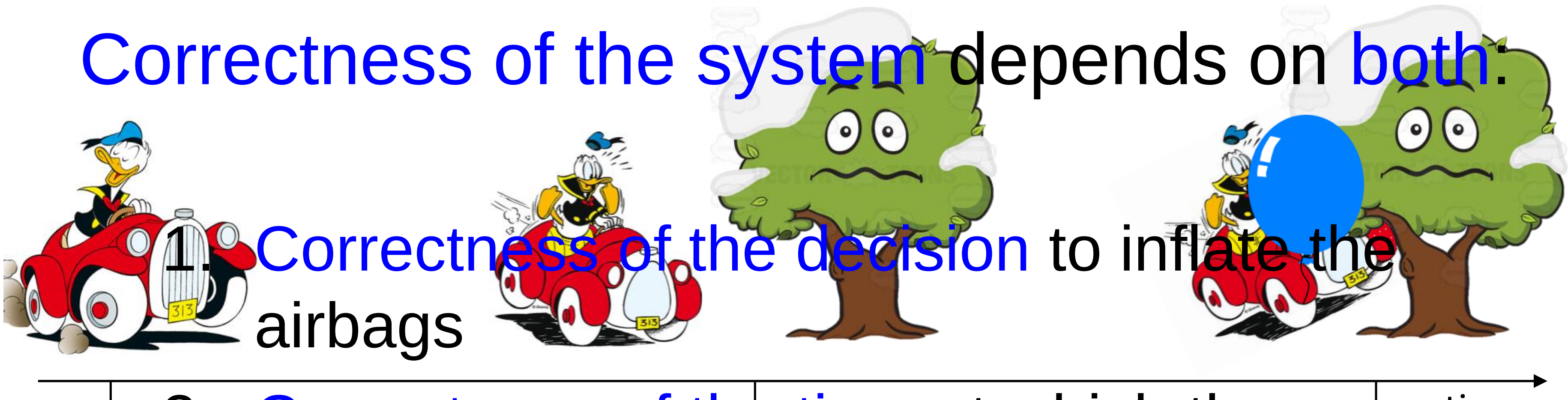





web images

# Real-time Systems

Correctness of the system depends on both:



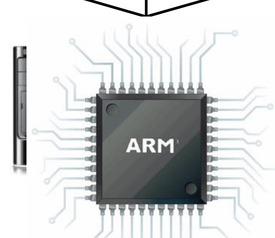

1. Correctness of the decision to inflate the airbags
2. Correctness of the time at which the decision is implemented

Computer programs called “tasks”

Detect the crash

Evaluate its intensity

Inflate the airbags




MÄLARDALEN UNIVERSITY SWEDEN

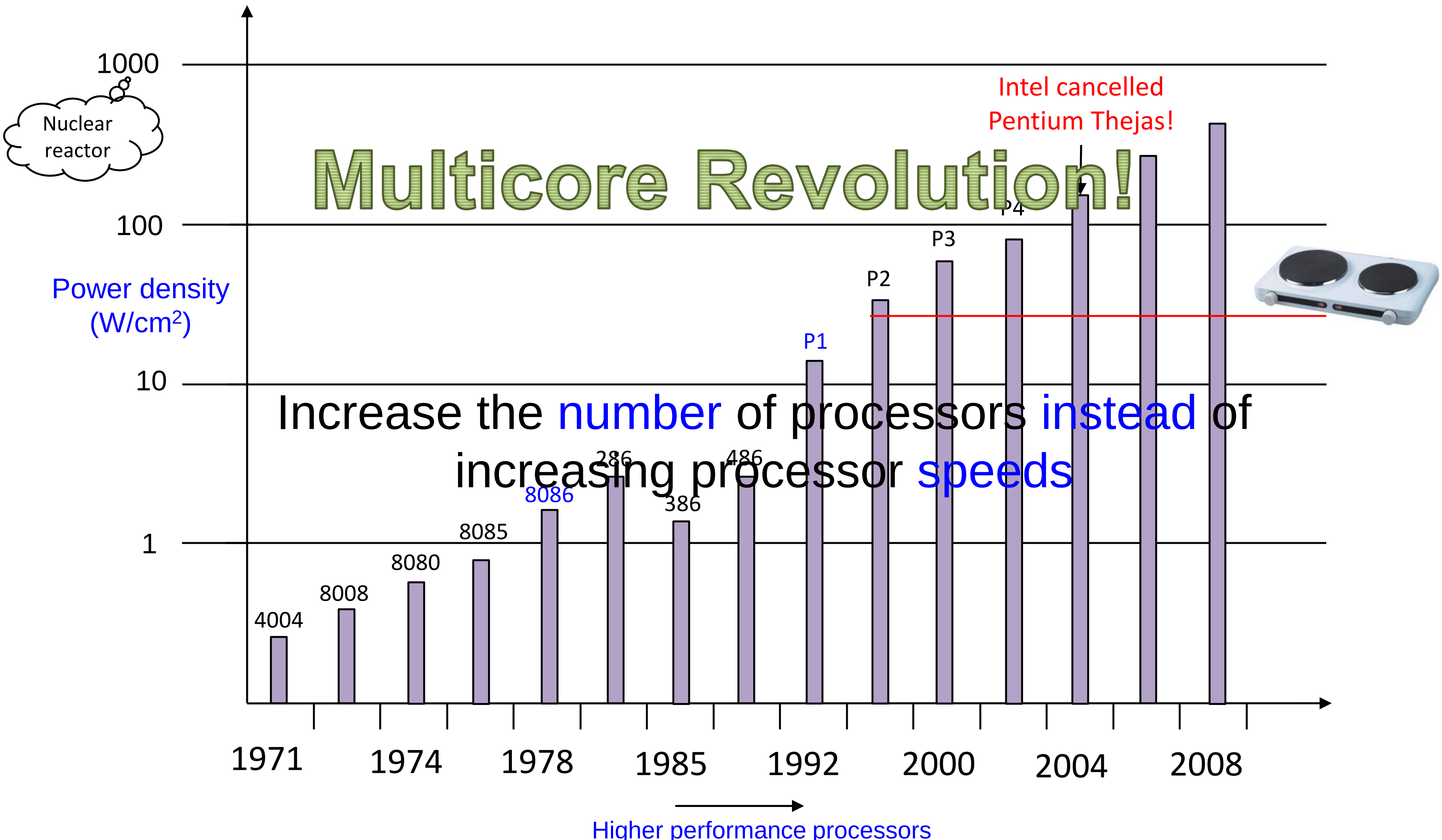
web images

# Real-time Scheduling

- Fewer computers than the number of tasks!
- Need to schedule the tasks such that deadlines are not missed
- For example tasks are typically executed highest priority first or earliest deadline first
- Schedulability test determines if a taskset meets deadlines under a given scheduling algorithm

MÄLARDALEN UNIVERSITY SWEDEN

# Faster Computers vs. Power Demand



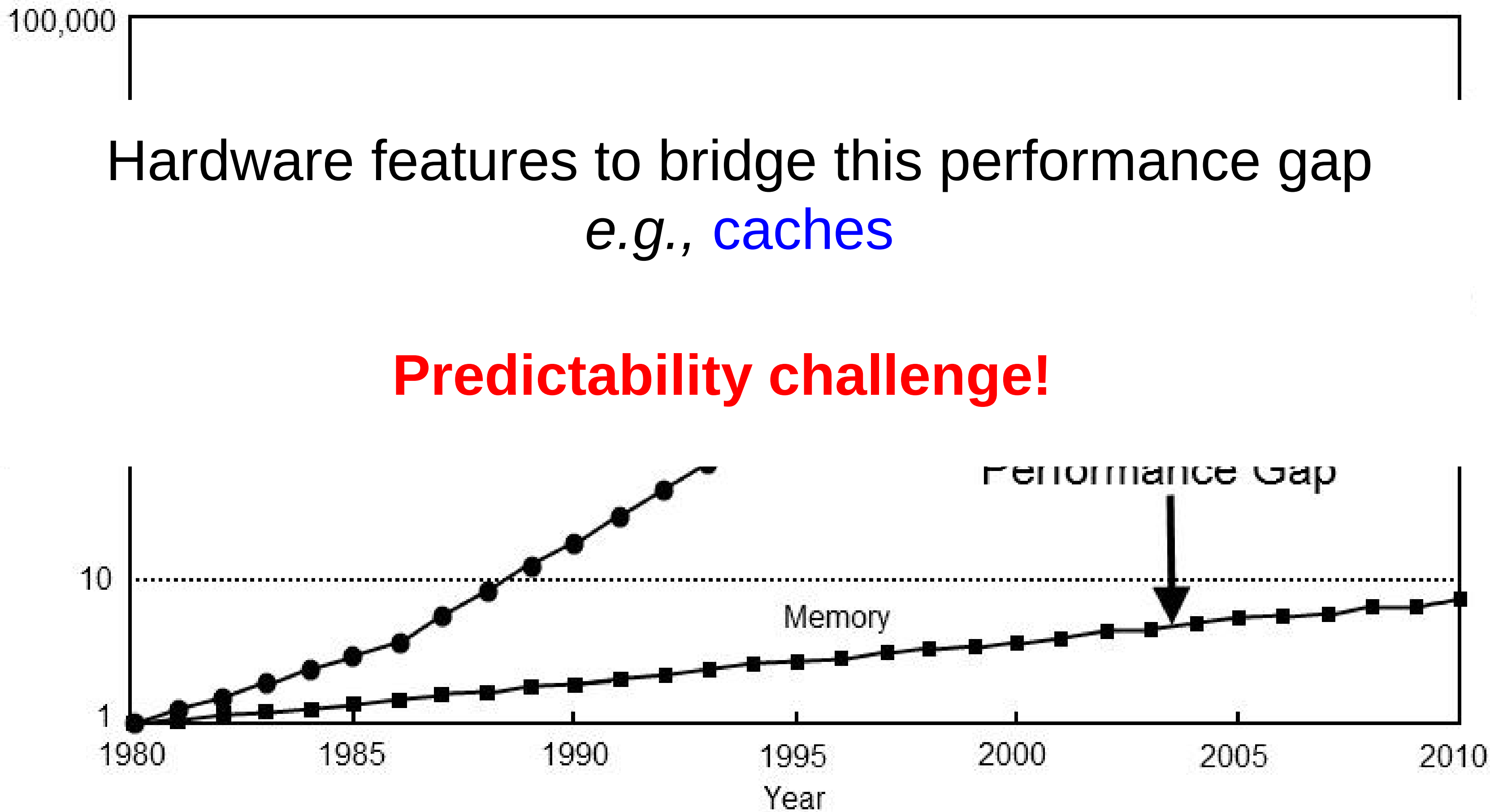

**Multicore Revolution!**

Increase the number of processors instead of increasing processor speeds

(adapted from Marko Bertogna's PhD thesis)

web images

# Processor to Memory Performance Gap



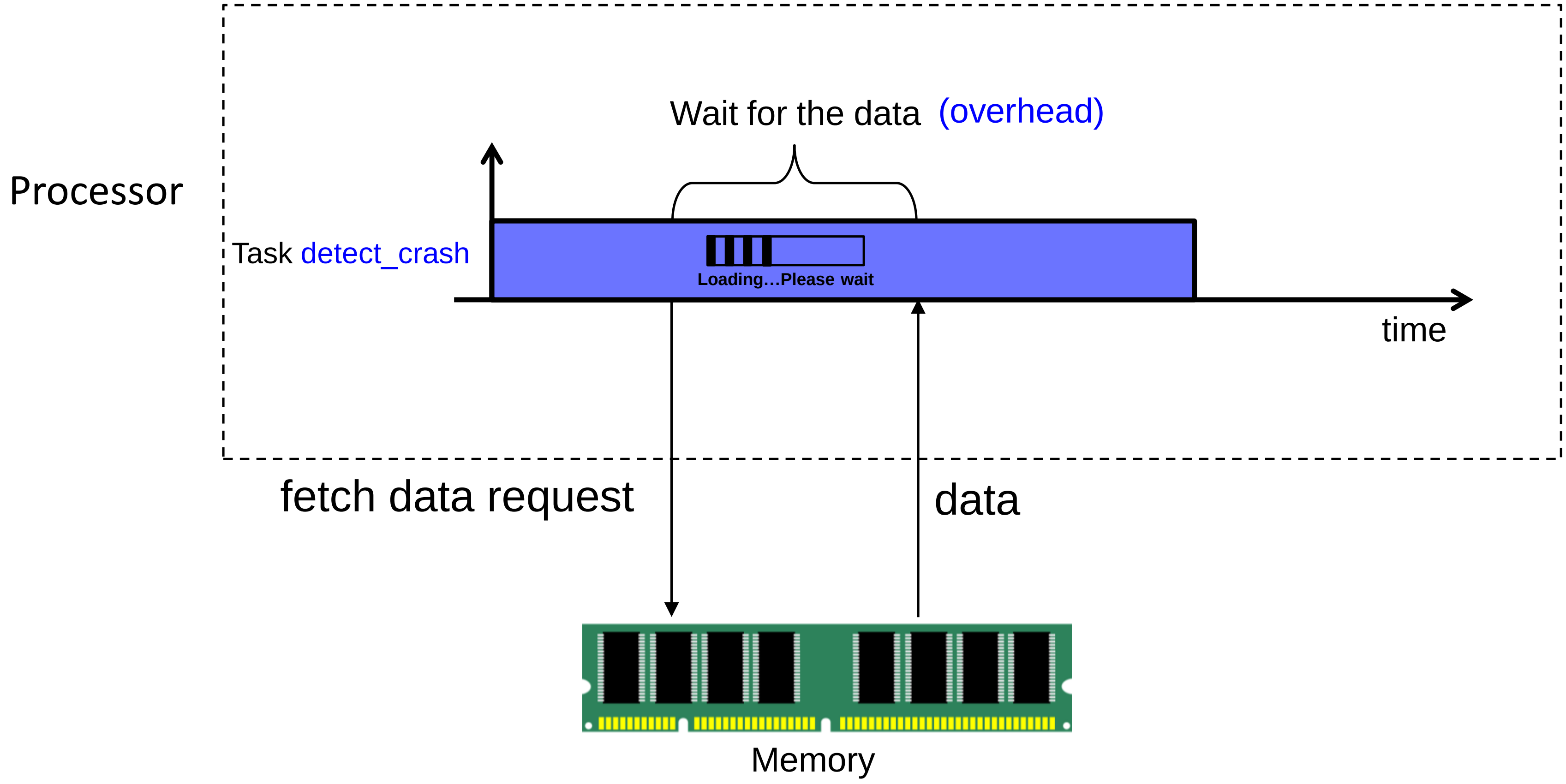

Hardware features to bridge this performance gap

*e.g.*, caches

**Predictability challenge!**

*Computer Architecture: A Quantitative Approach* by John L. Hennessy, David A. Patterson, Andrea C. Arpaci-Dusseau

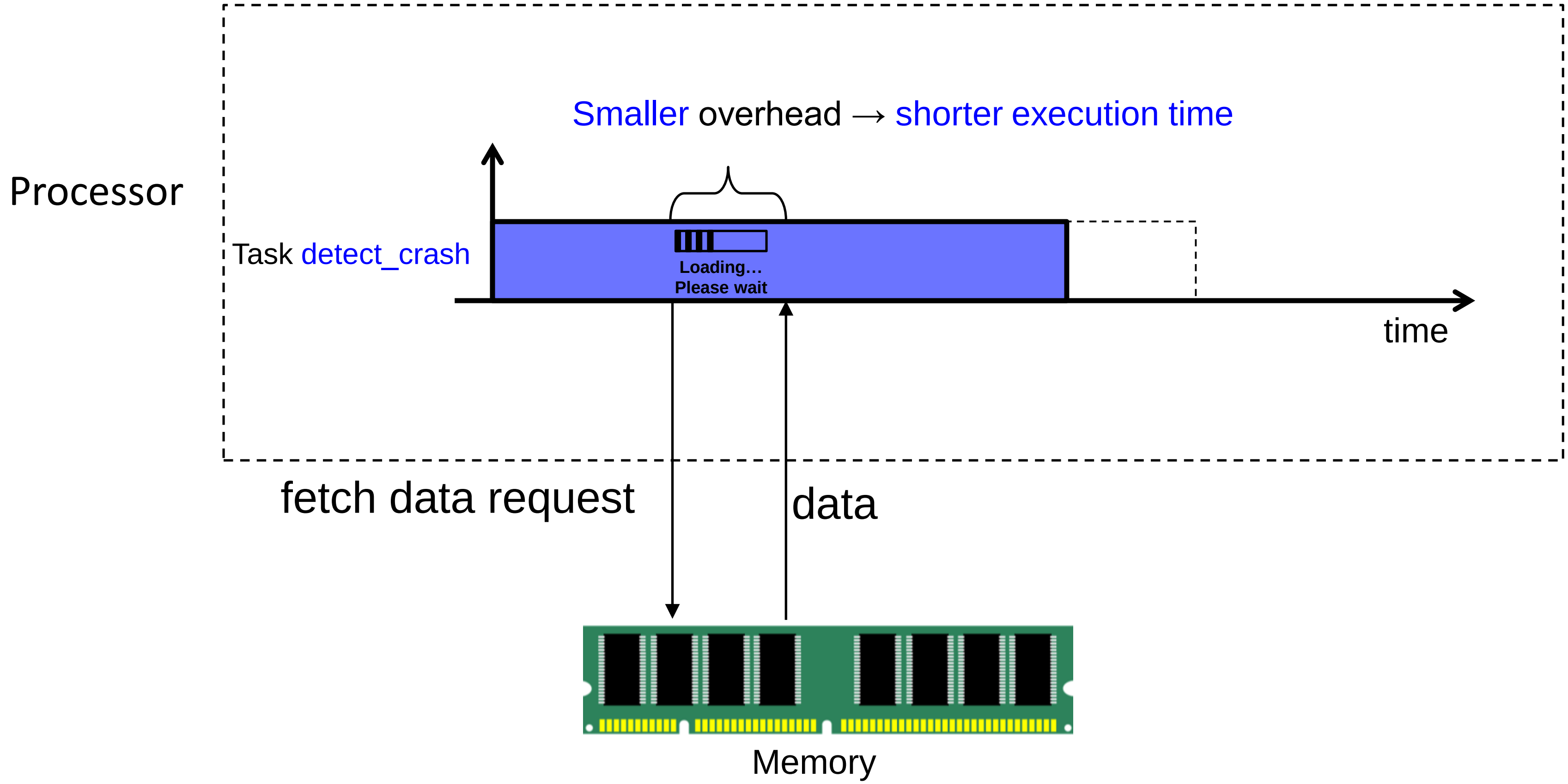
# Processor to Memory Performance Gap





web images

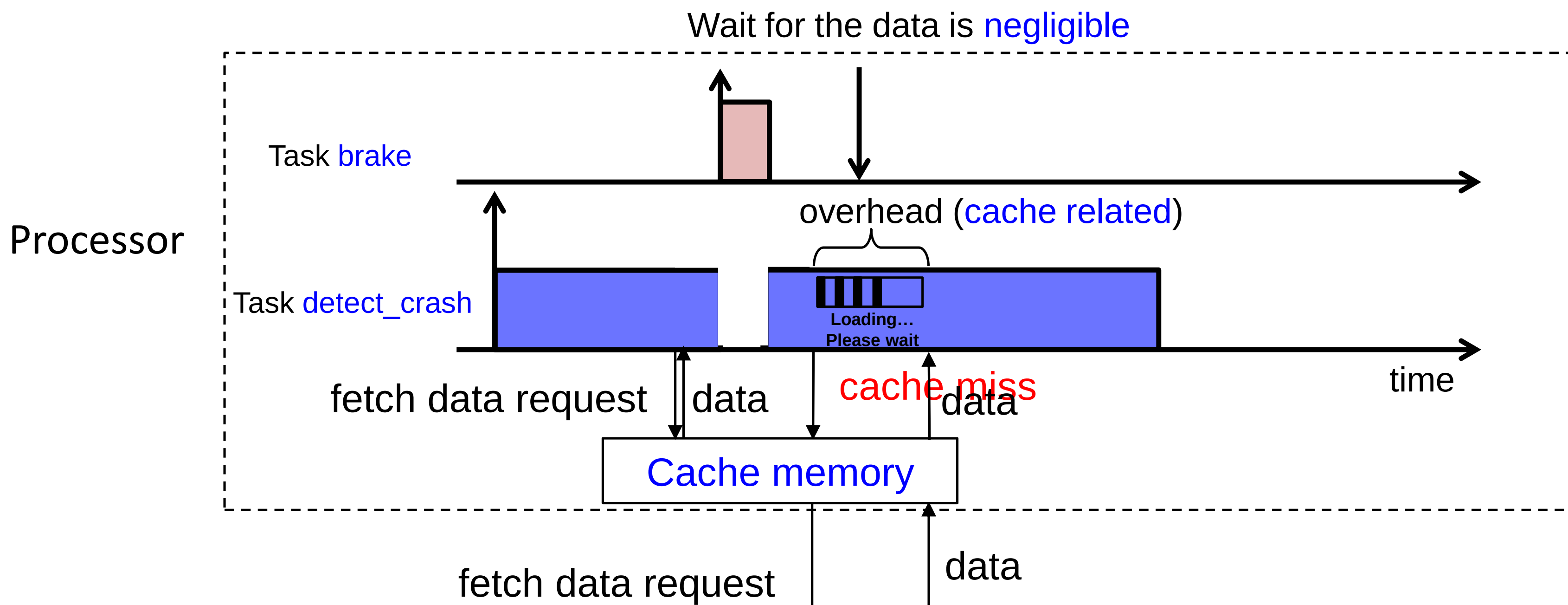
# Processor to Memory Performance Gap

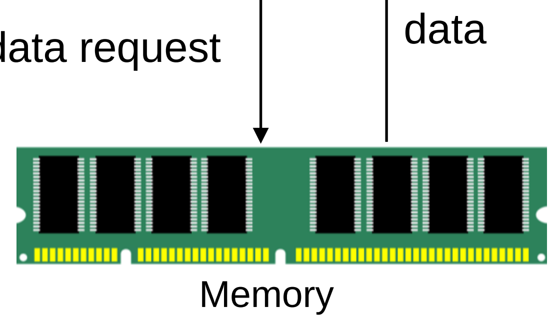




web images

# Bridging the Processor to Memory Performance Gap




**Alternative 1**: build predictable hardware *e.g.,* PRETS

**Alternative 2**: efficient resource allocation & overhead accounting

MÄLARDALEN UNIVERSITY SWEDEN



web images

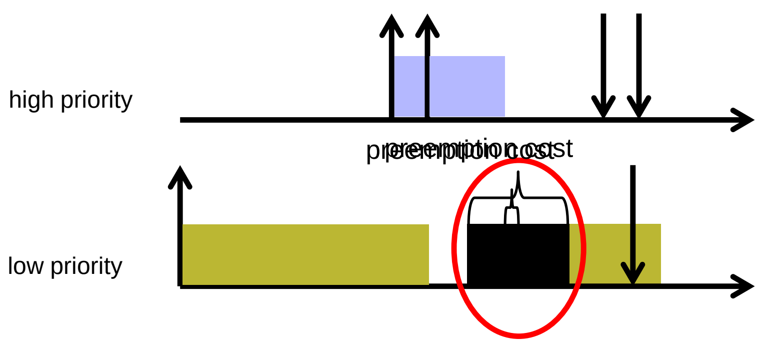
# Motivation

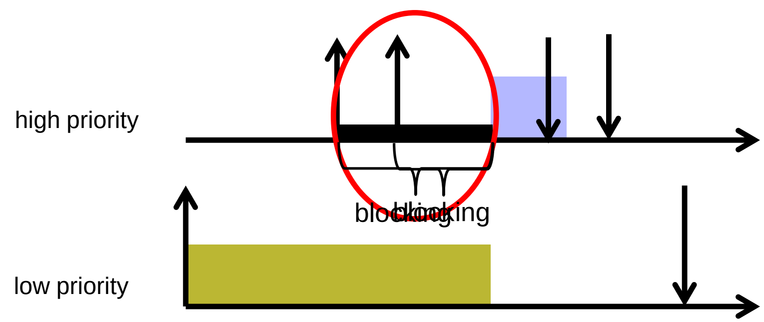
## Preemptive Scheduling

- ☺ Zero blocking: high schedulable utilization
- ☹ High runtime overhead: preemption costs
- ☹ Difficult to perform timing analysis
- ☹ Difficult to demonstrate predictability and safety: multicores

## Non-Preemptive Scheduling

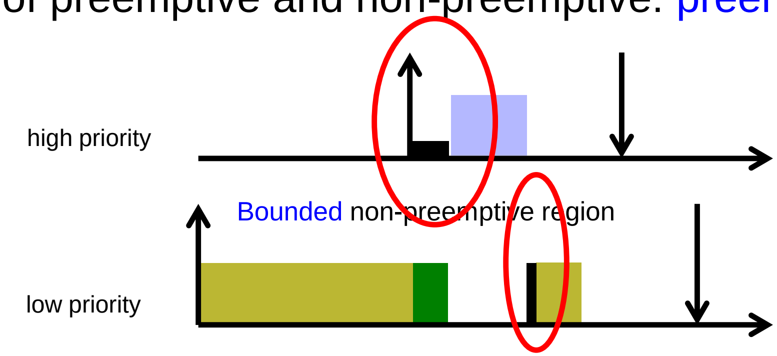
- ☺ Low runtime overhead: zero preemption costs
- ☺ Relatively easier to perform timing analysis
- ☺ Preferred by many safety standards
- ☹ Increased blocking: low schedulable utilization




## Solution: Limited-Preemptive Scheduling

- ☺ Best of preemptive and non-preemptive: preempt only when necessary




MÄLARDALEN UNIVERSITY SWEDEN

# Limited Preemptive Scheduling Models

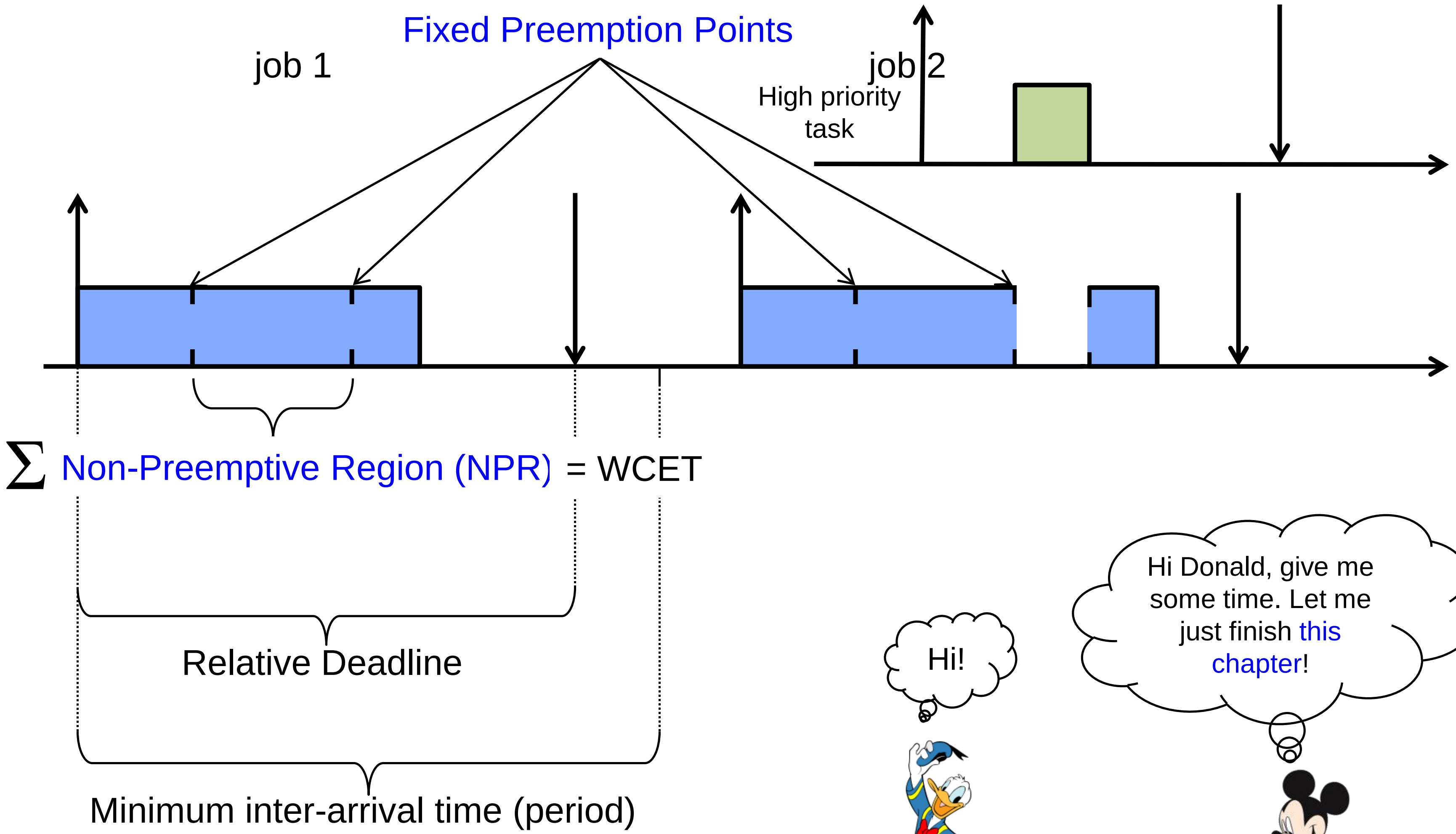
- Preemption Threshold Scheduling (ThreadX, Wang and Saksena, 1999)
- Fixed Preemption Point Scheduling (Burns, 1994, Bril *et al.,* 2009, Yao *et al.,* 2011)

Our focus

- Floating Non-Preemptive Region Scheduling (Baruah, 2005)
- …

MÄLARDALEN UNIVERSITY SWEDEN

# Fixed Preemption Points Scheduling





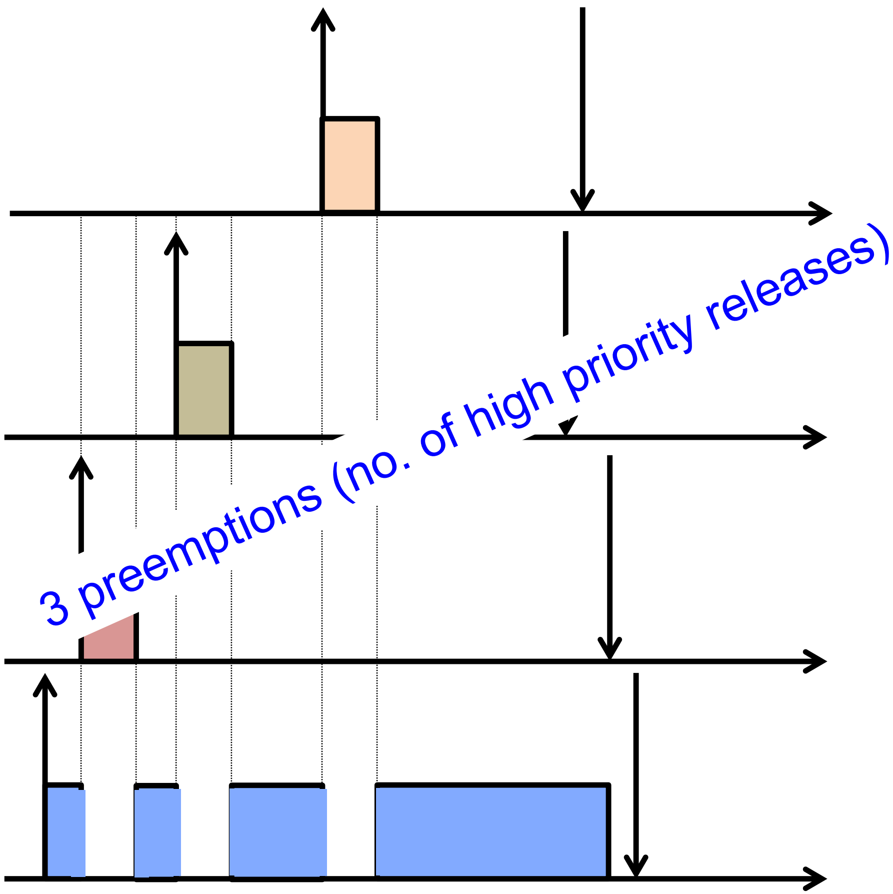
MÄLARDALEN UNIVERSITY SWEDEN



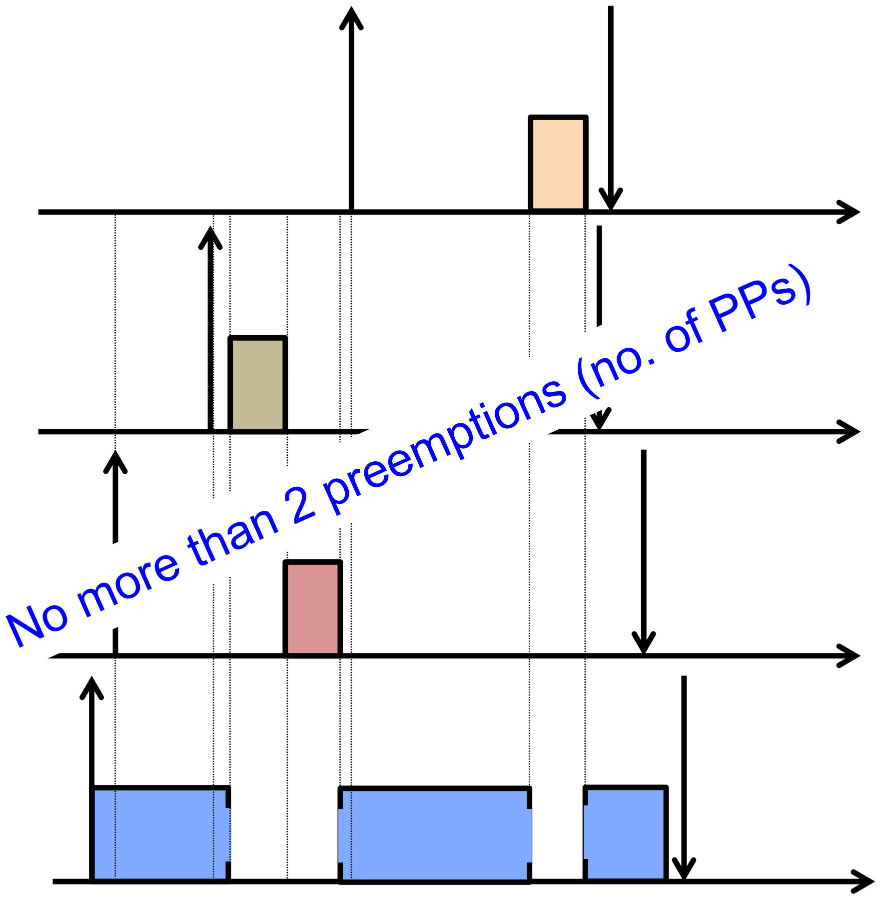
web images

# Reduction in Preemptions

Fully Preemptive Scheduling




Fixed Preemption Points Scheduling




MÄLARDALEN UNIVERSITY SWEDEN

# Limited Preemptive Scheduling Landscape

| | Fixed priority based scheduling | Deadline based scheduling |
|---|---|---|
| **Uniprocessor** | Limited Preemptive FPS | Limited Preemptive EDF |
| **Multiprocessor** | Global Limited Preemptive FPS | Global Limited Preemptive EDF |

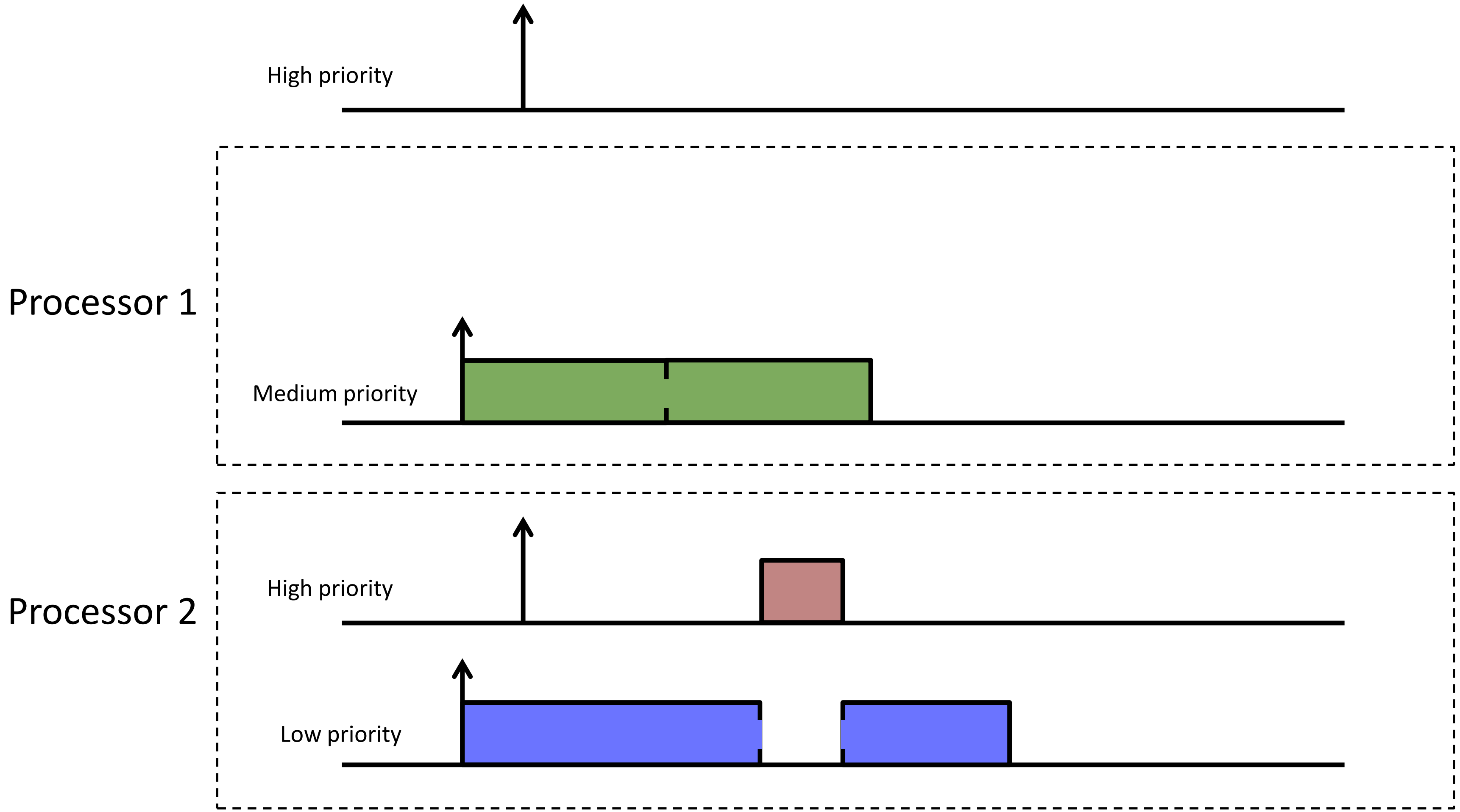
Handling preemptions in global LP scheduling:

1. Eager Preemption Approach
2. Lazy Preemption Approach

MÄLARDALEN UNIVERSITY SWEDEN

# Global Limited Preemptive Scheduling

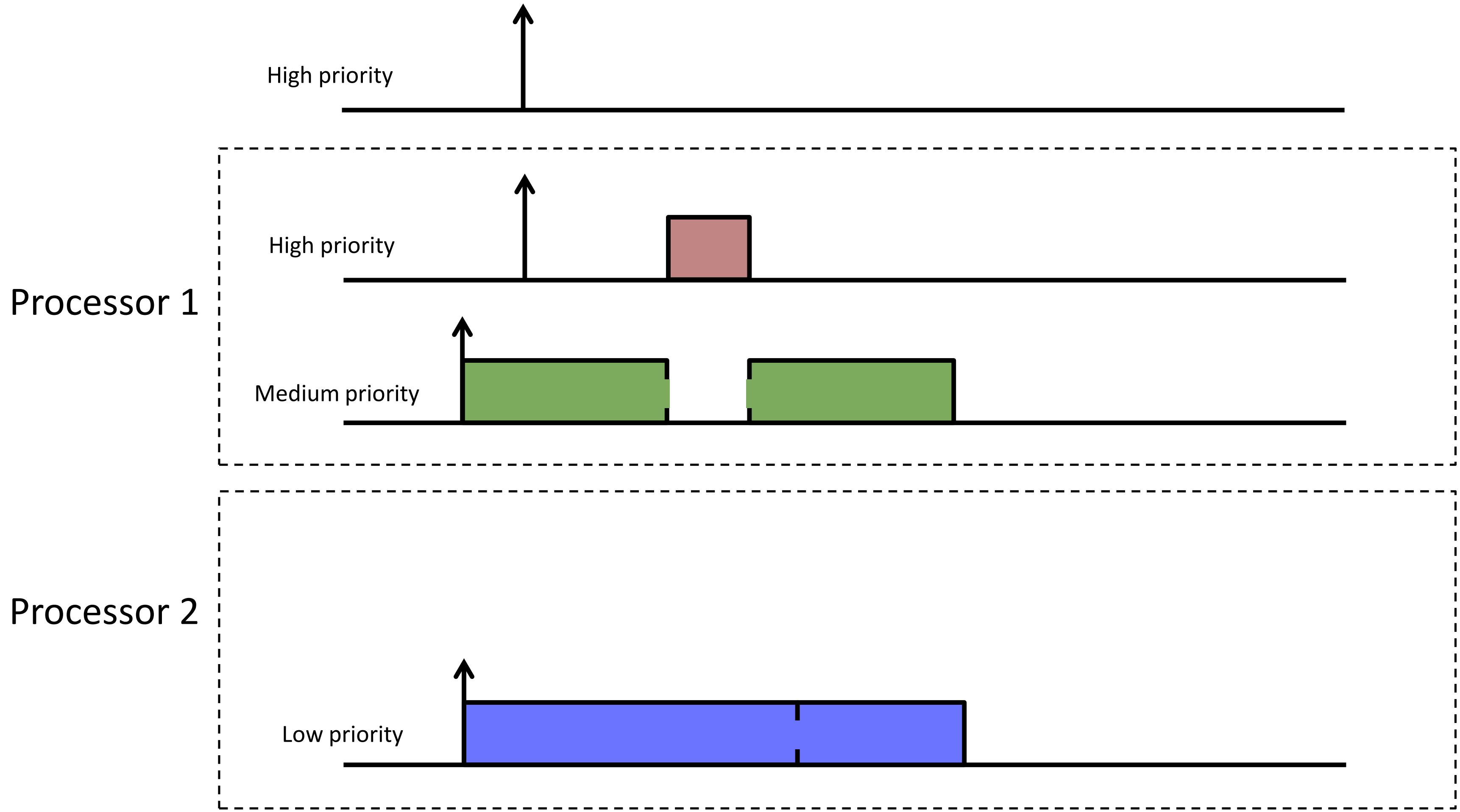
Lazy Preemption Approach: wait to preempt the lowest priority task




MÄLARDALEN UNIVERSITY
SWEDEN

# Global Limited Preemptive Scheduling

Eager Preemption Approach: preempt the lower priority task that first finishes its NPR (not necessarily the lowest)




MÄLARDALEN UNIVERSITY SWEDEN

# Advances in the Limited Preemptive Scheduling Landscape

| | | |
|---|---|---|
| **Uniprocessor** | Limited preemptive FPS<br>Burns'94, Bril *et al.*, RTSJ'09, Yao *et al.*, RTSJ'11 | Limited preemptive EDF<br>Baruah, ECRTS'05 |
| **Multiprocessor** | Global Limited Preemptive FPS<br><br>Marinho *et al.*, RTSS'13, Davis *et al.*, TECS'15, Thekkilakattil *et al.*, RTNS'15, Marinho, PhD Thesis'15, Serrano *et al.*, DATE'16 | Global Limited Preemptive EDF<br><br>Block et al., RTCSA'07<br>Thekkilakattil *et al.*, ECRTS'14,<br>Chattopadhyay *et al.*, RTNS'14,<br>Marinho, PhD Thesis'15 |

Focus on schedulability!

# Preemptive Behavior of FPS and EDF

… under the preemptive and limited preemptive paradigms
for eager and lazy approaches

| | |
|---|---|
| **Uniprocessors** | Buttazzo, RTSJ'05<br>(RM vs EDF: Judgement Day) |
| **Multiprocessors** | No significant work! |

How does the number of preemptions vary with the scheduling algorithm and approach to preemption on multiprocessors?

MÄLARDALEN UNIVERSITY
SWEDEN

# Experimental Setup

- Weighted metric to count preemptions:

$$W(p) = \frac{\sum_{\forall\Gamma} U(\Gamma)P(\Gamma, p)}{\sum_{\forall\Gamma} U(\Gamma)}$$

- Enables investigation of number of preemptions *w.r.t* a second parameter in addition to utilization

- Tasksets generated using Uunifast-Discard
- Periods in the range *5-500*
- Utilizations in the range *1* to *m/2* (in one case upto m)
- Schedule simulated for 10000 time units

MÄLARDALEN UNIVERSITY SWEDEN

# Weighted Metric to Count Preemptions

$$W(p) = \frac{\sum_{\forall \Gamma} U(\Gamma) P(\Gamma, p)}{\sum_{\forall \Gamma} U(\Gamma)}$$

Number of preemptions generated by taskset $\Gamma$ *w.r.t* parameter p

MÄLARDALEN UNIVERSITY SWEDEN

# Weighted Metric to Count Preemptions

Utilization of taskset $\Gamma$

$$W(p) = \frac{\sum_{\forall\Gamma} U(\Gamma) P(\Gamma, p)}{\sum_{\forall\Gamma} U(\Gamma)}$$

MÄLARDALEN UNIVERSITY SWEDEN

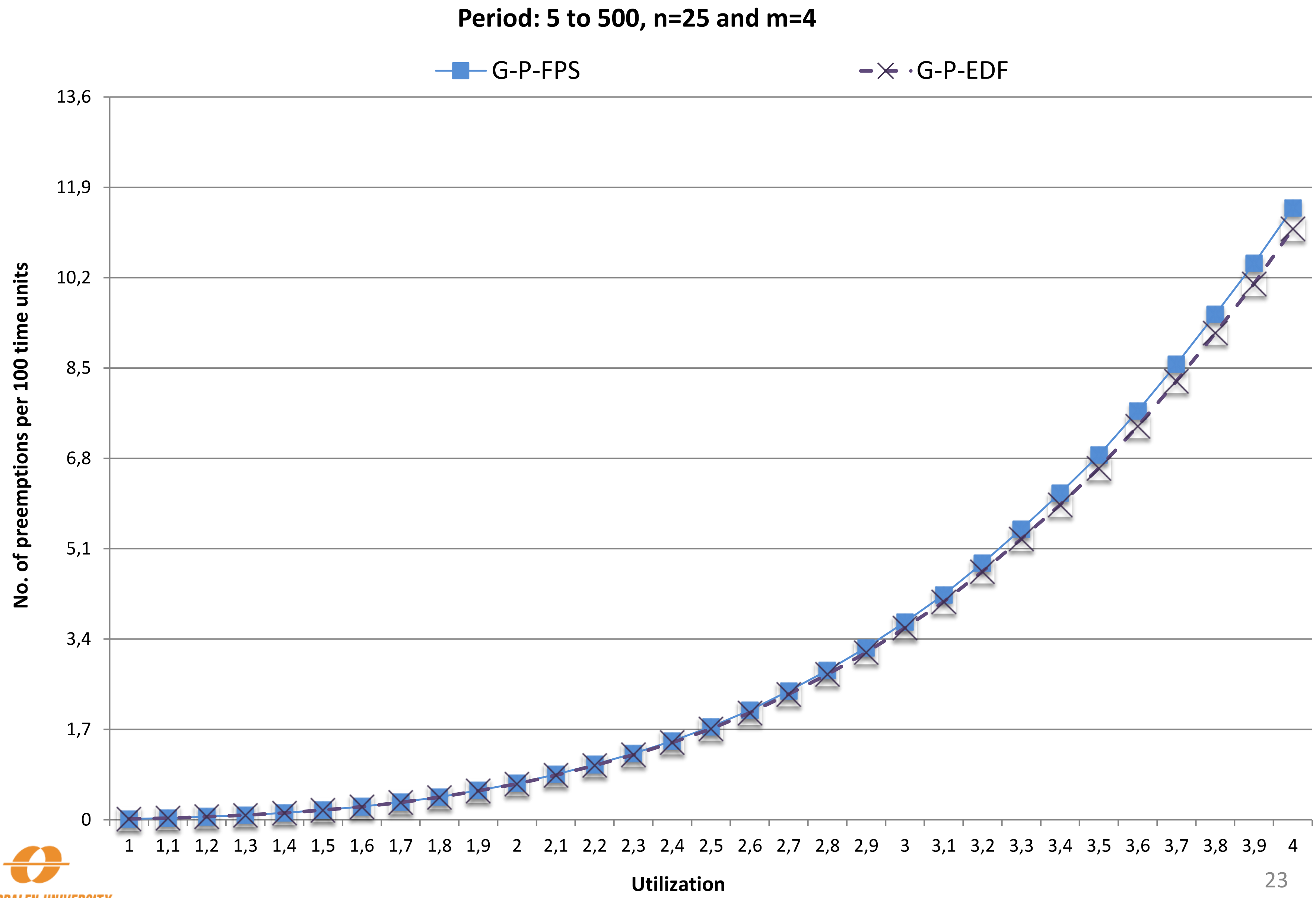
# Experiments

We investigated how *weighted preemptions* vary with:

1. Varying utilizations
2. Varying number of tasks
3. Varying number of processors
4. Varying NPR lengths

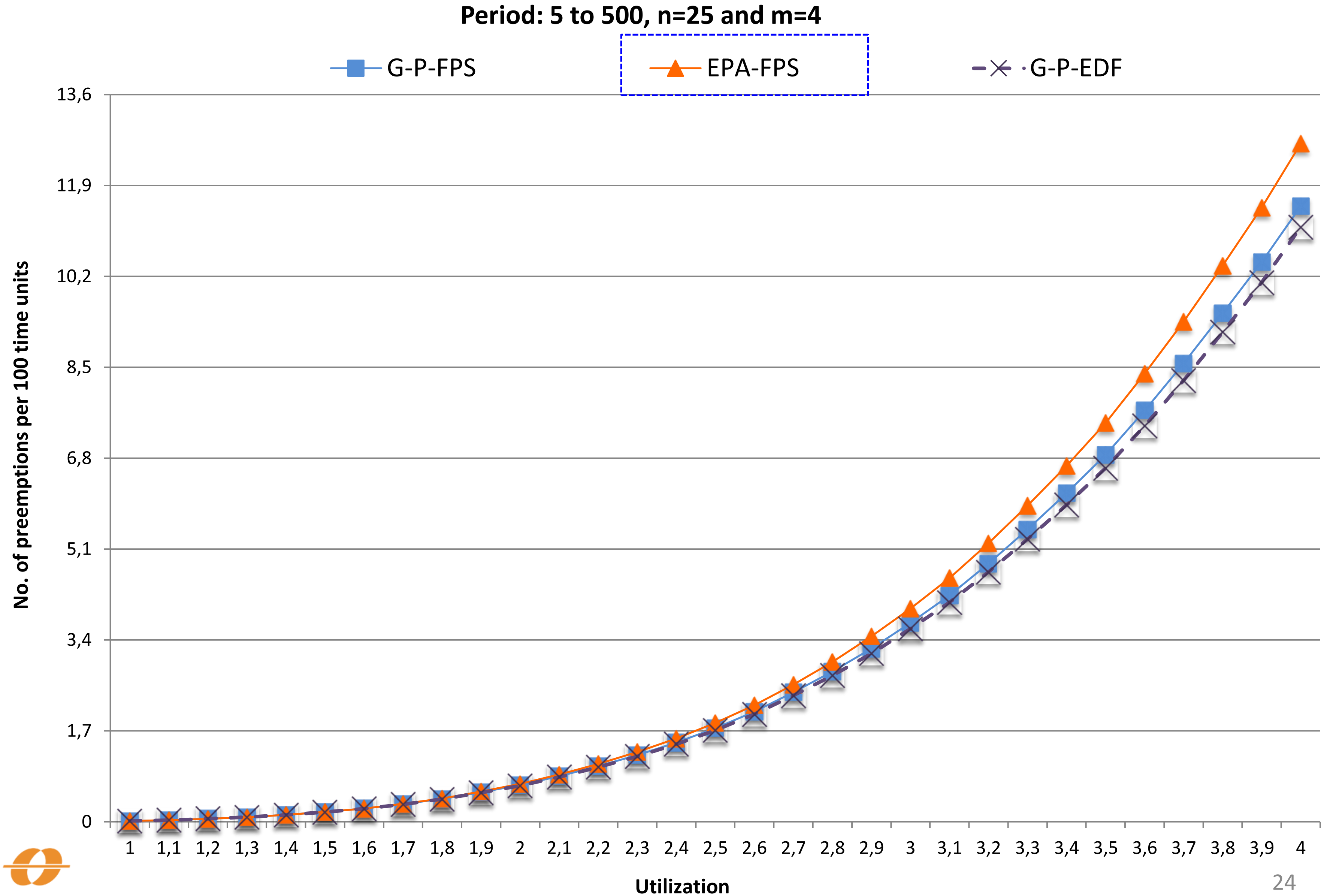
MÄLARDALEN UNIVERSITY SWEDEN

# Varying Utilization



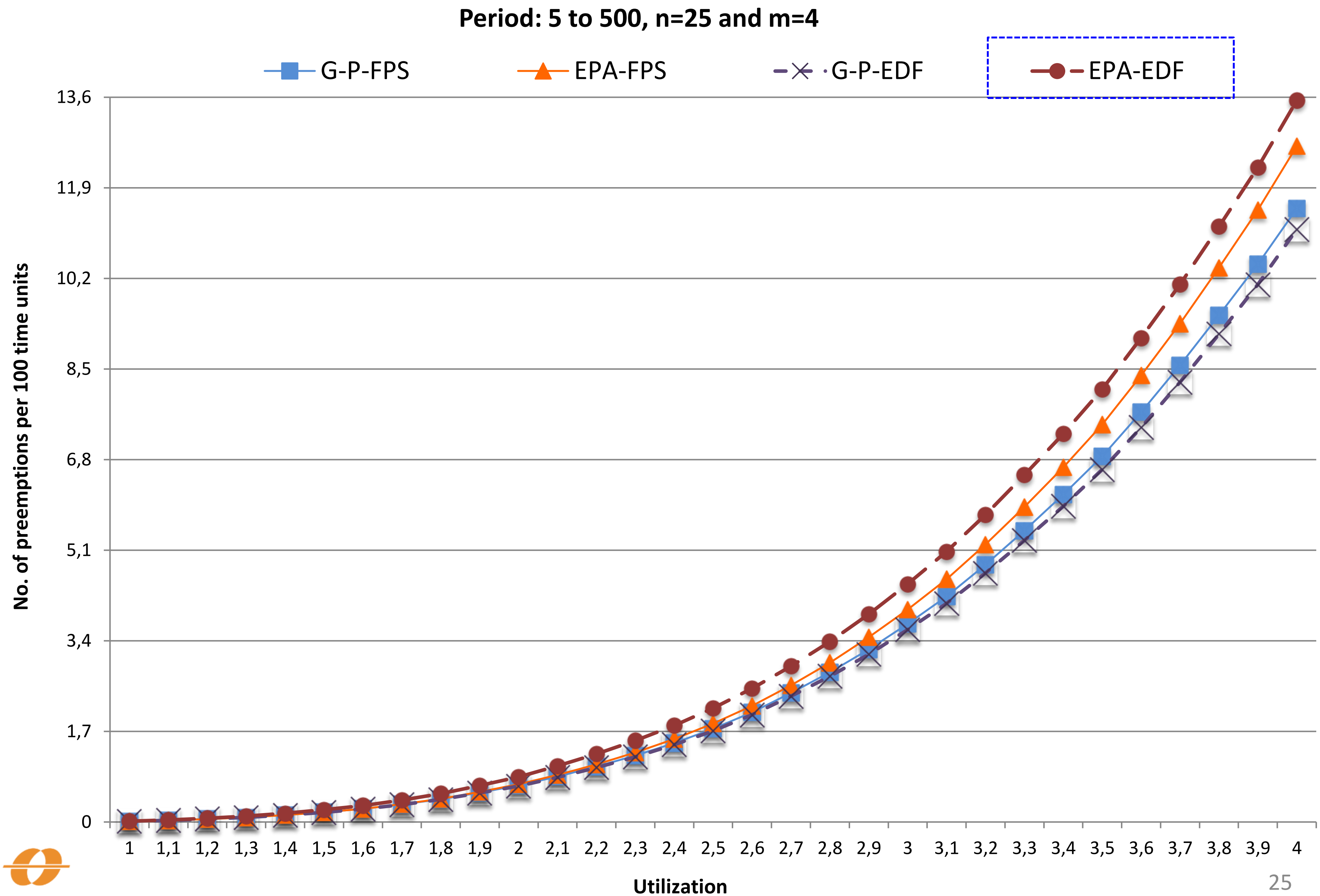

MÄLARDALEN UNIVERSITY SWEDEN

# Varying Utilization




MÄLARDALEN UNIVERSITY SWEDEN

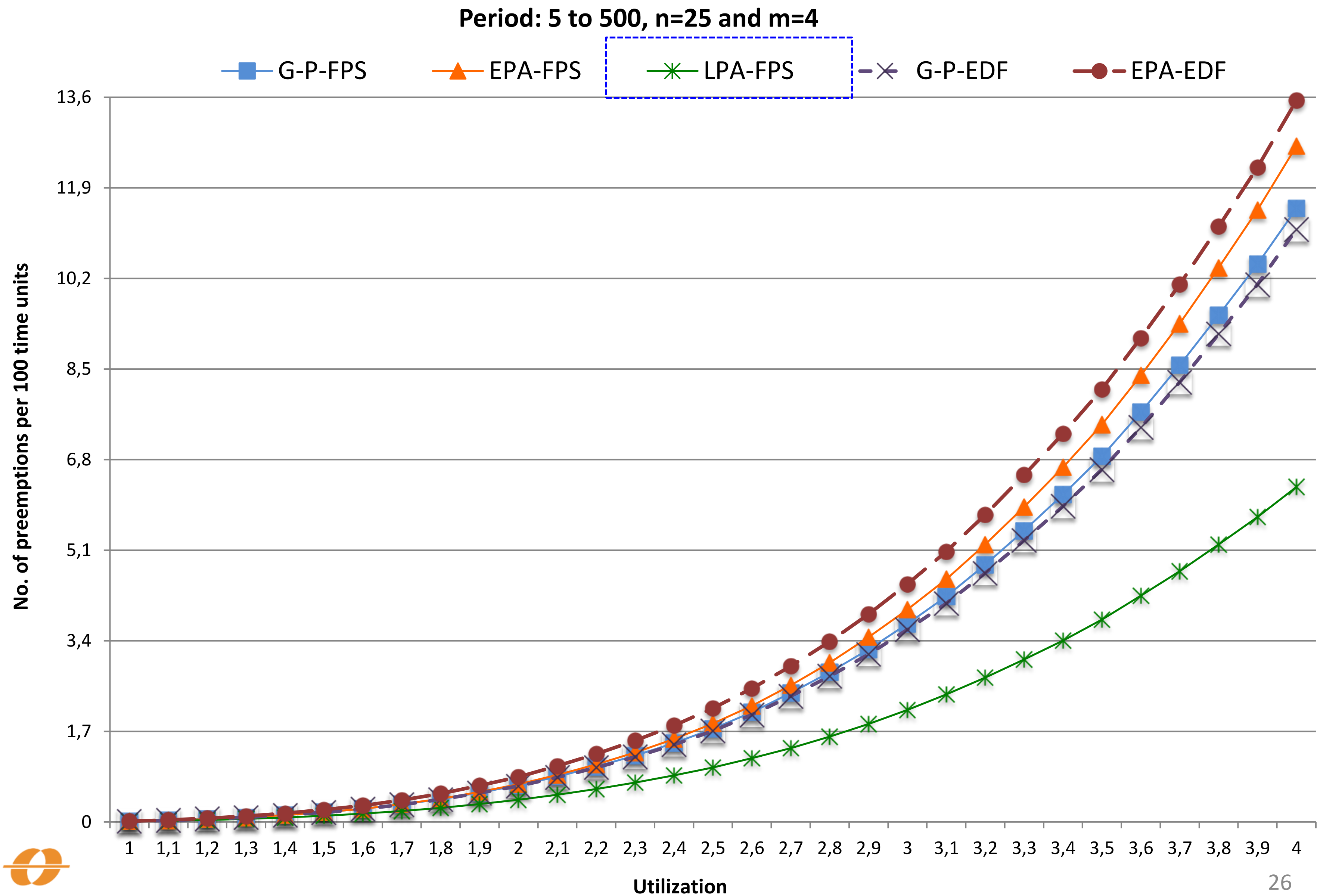
# Varying Utilization





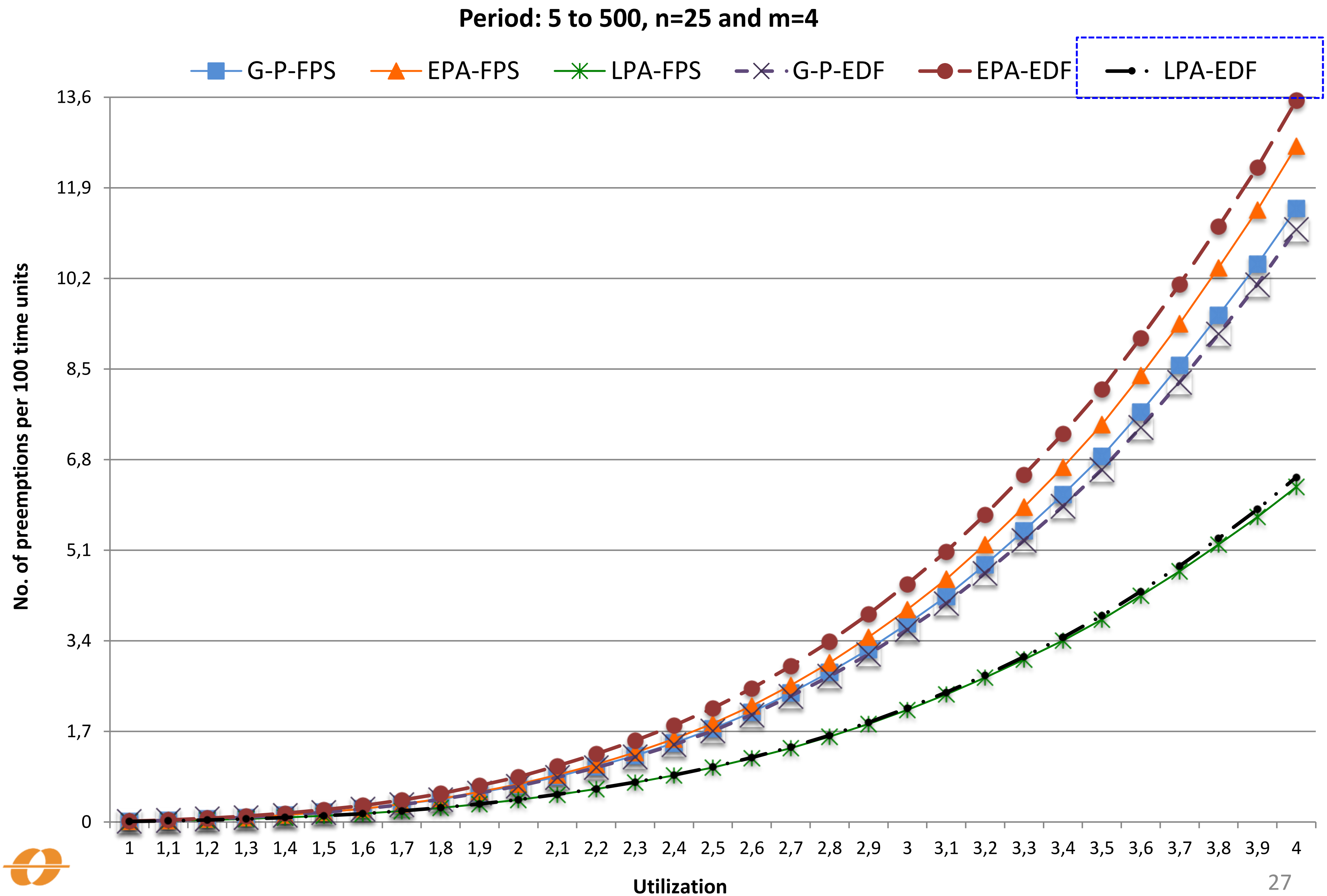

MÄLARDALEN UNIVERSITY SWEDEN

# Varying Utilization




MÄLARDALEN UNIVERSITY SWEDEN

# Varying Utilization




MÄLARDALEN UNIVERSITY SWEDEN

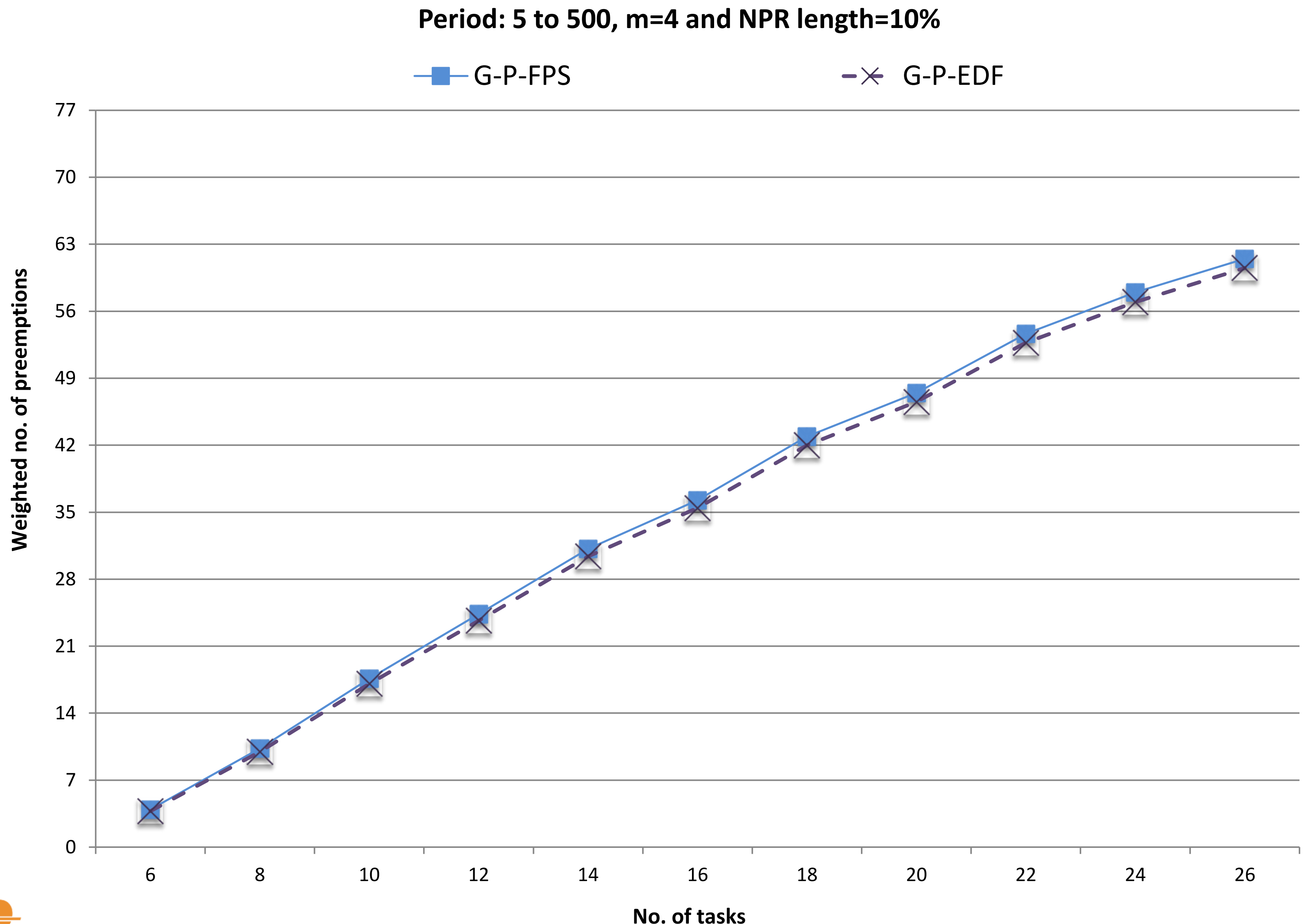
# Experiments

We investigated how *weighted preemptions* vary with:

1. Varying utilizations
2. Varying number of tasks
3. Varying number of processors
4. Varying NPR lengths

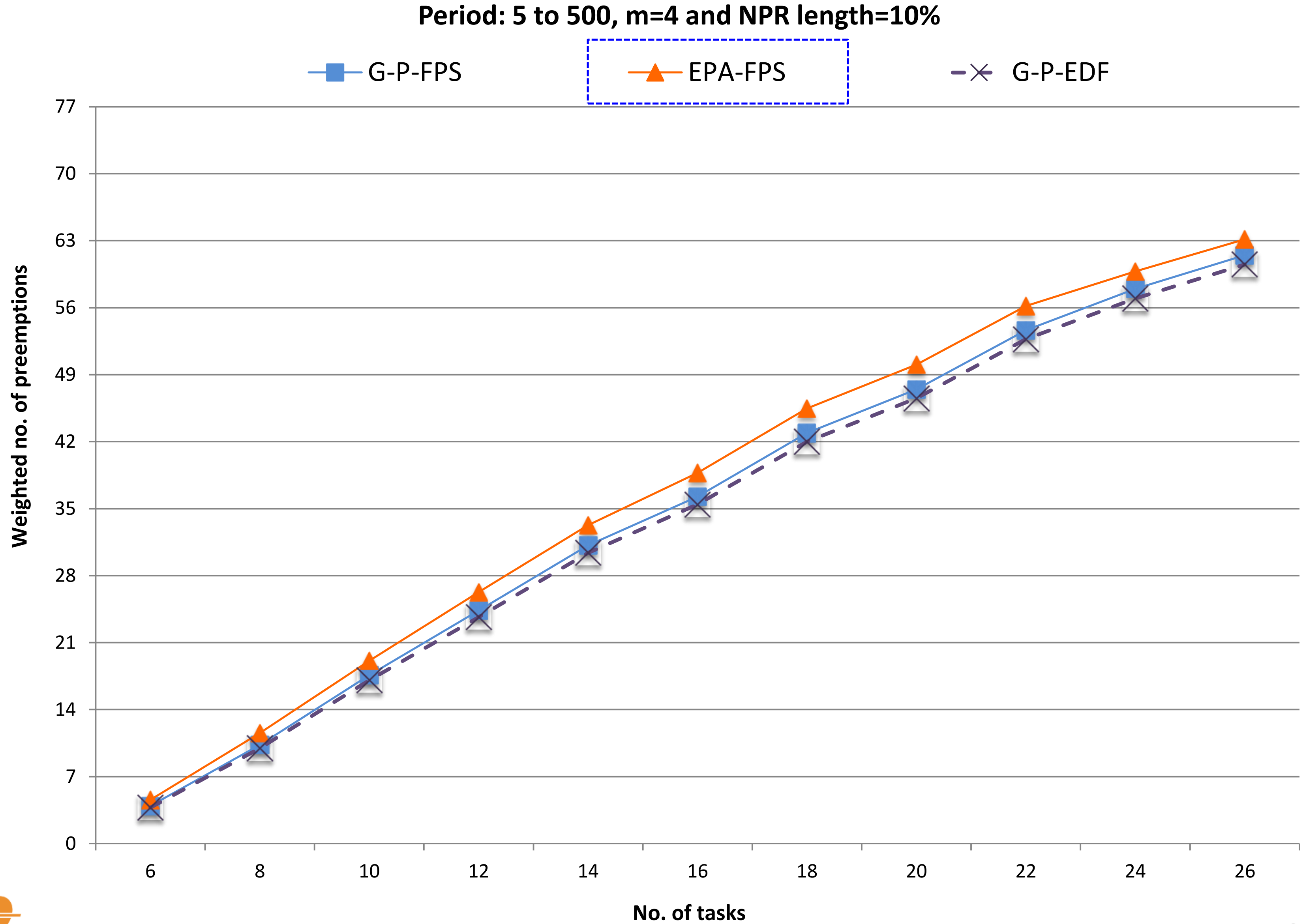
MÄLARDALEN UNIVERSITY SWEDEN

# Varying Number of Tasks



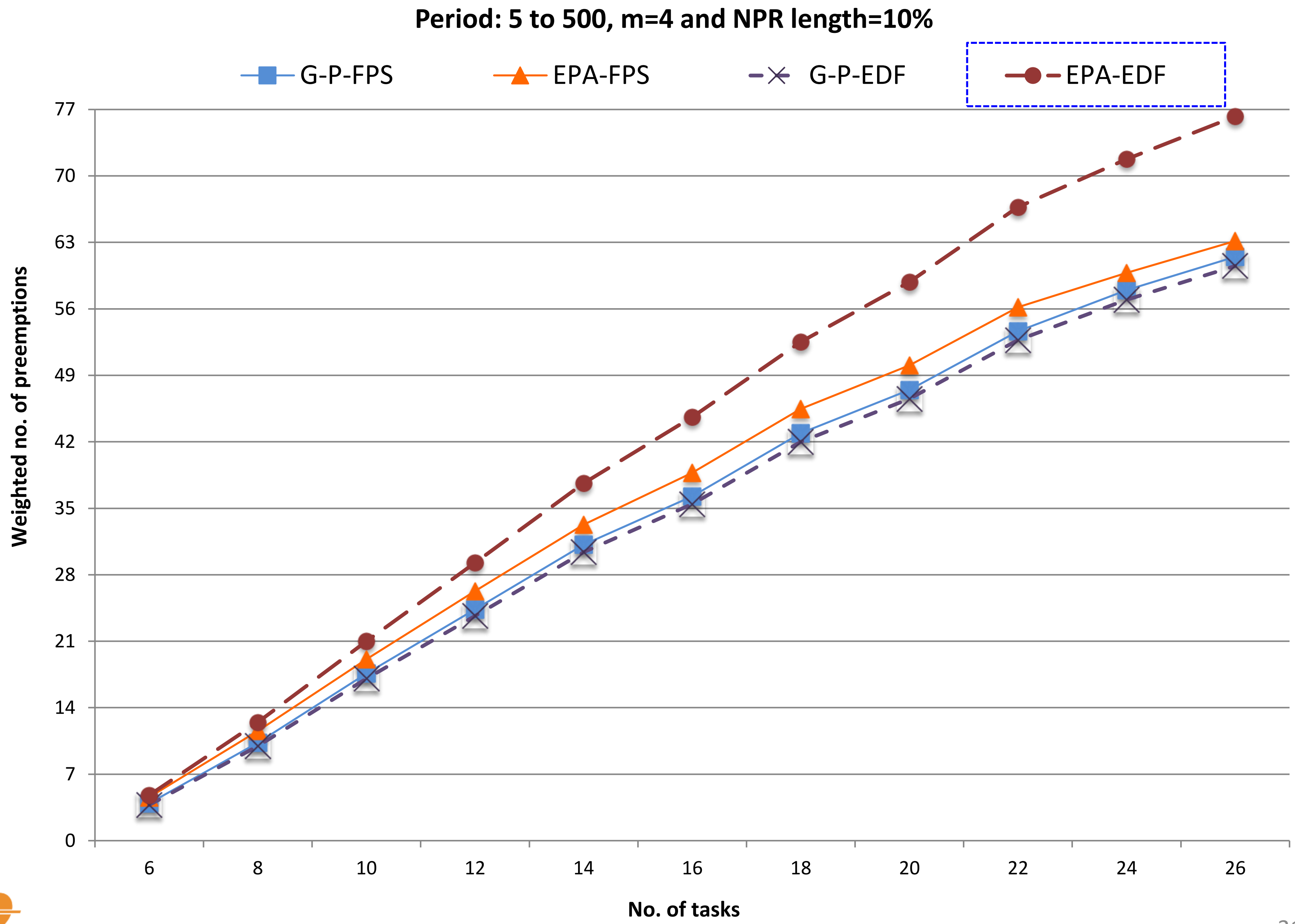

MÄLARDALEN UNIVERSITY SWEDEN

# Varying Number of Tasks



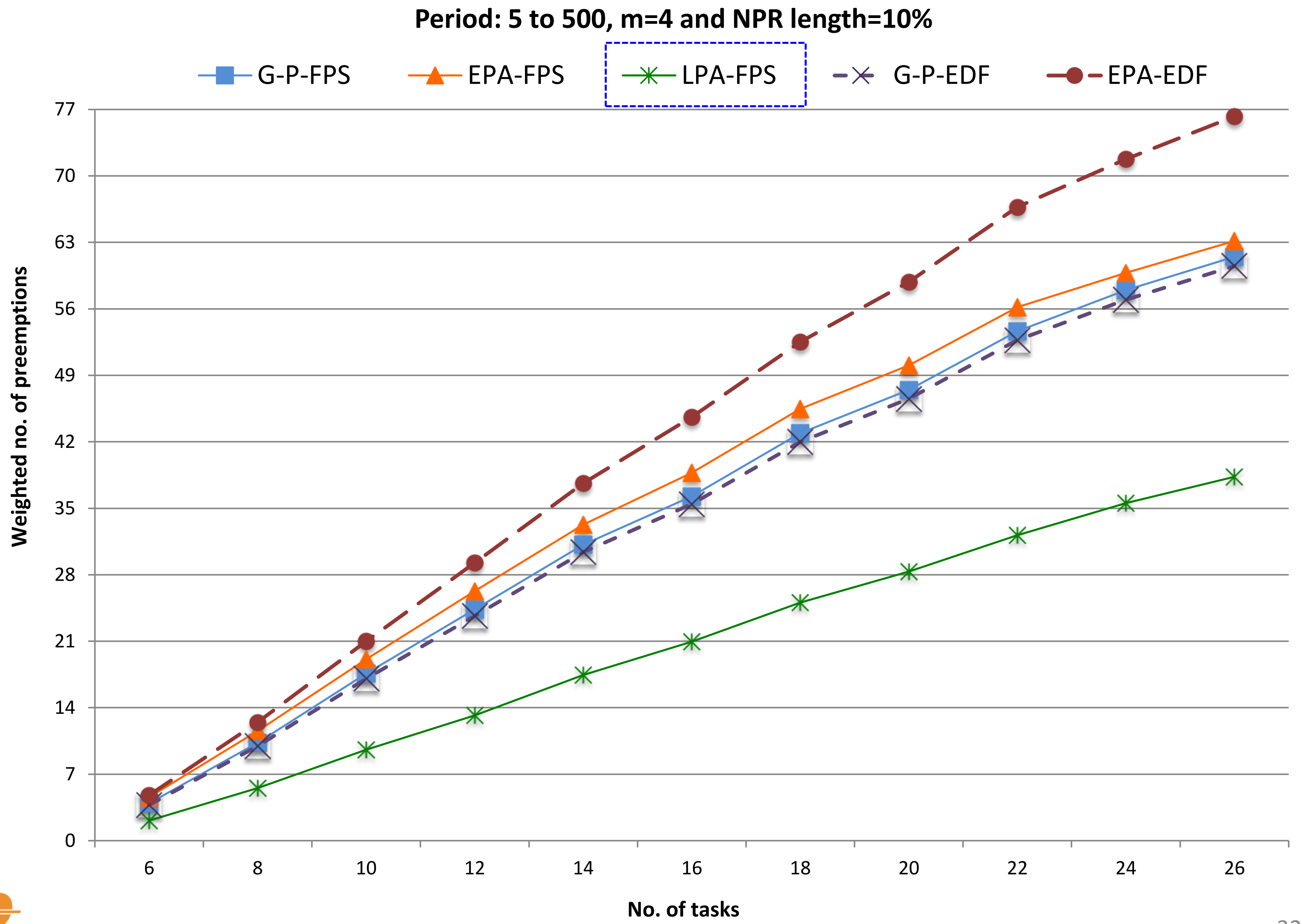

MÄLARDALEN UNIVERSITY SWEDEN

# Varying Number of Tasks



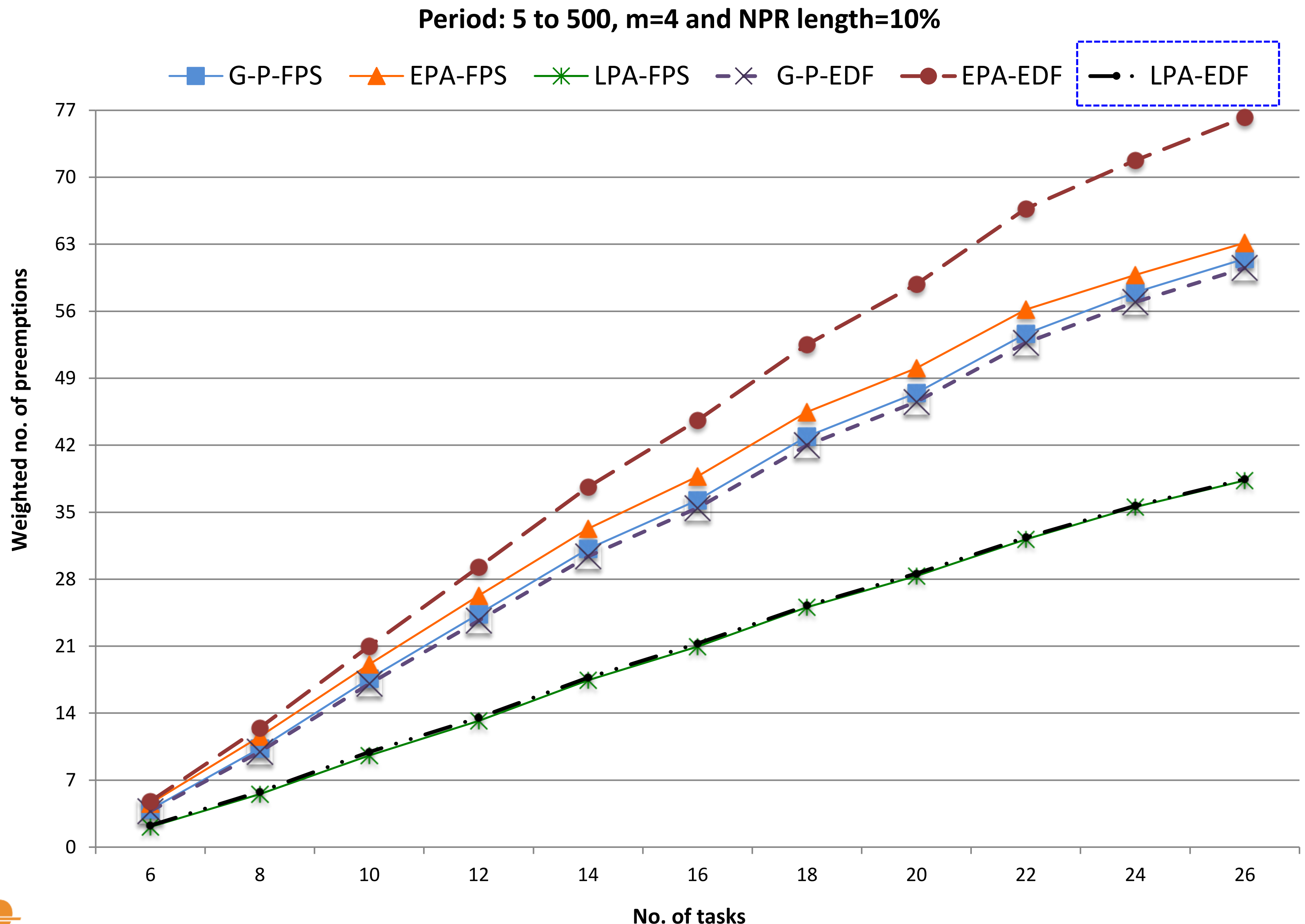

MÄLARDALEN UNIVERSITY SWEDEN

# Varying Number of Tasks




MÄLARDALEN UNIVERSITY SWEDEN

# Varying Number of Tasks




MÄLARDALEN UNIVERSITY SWEDEN

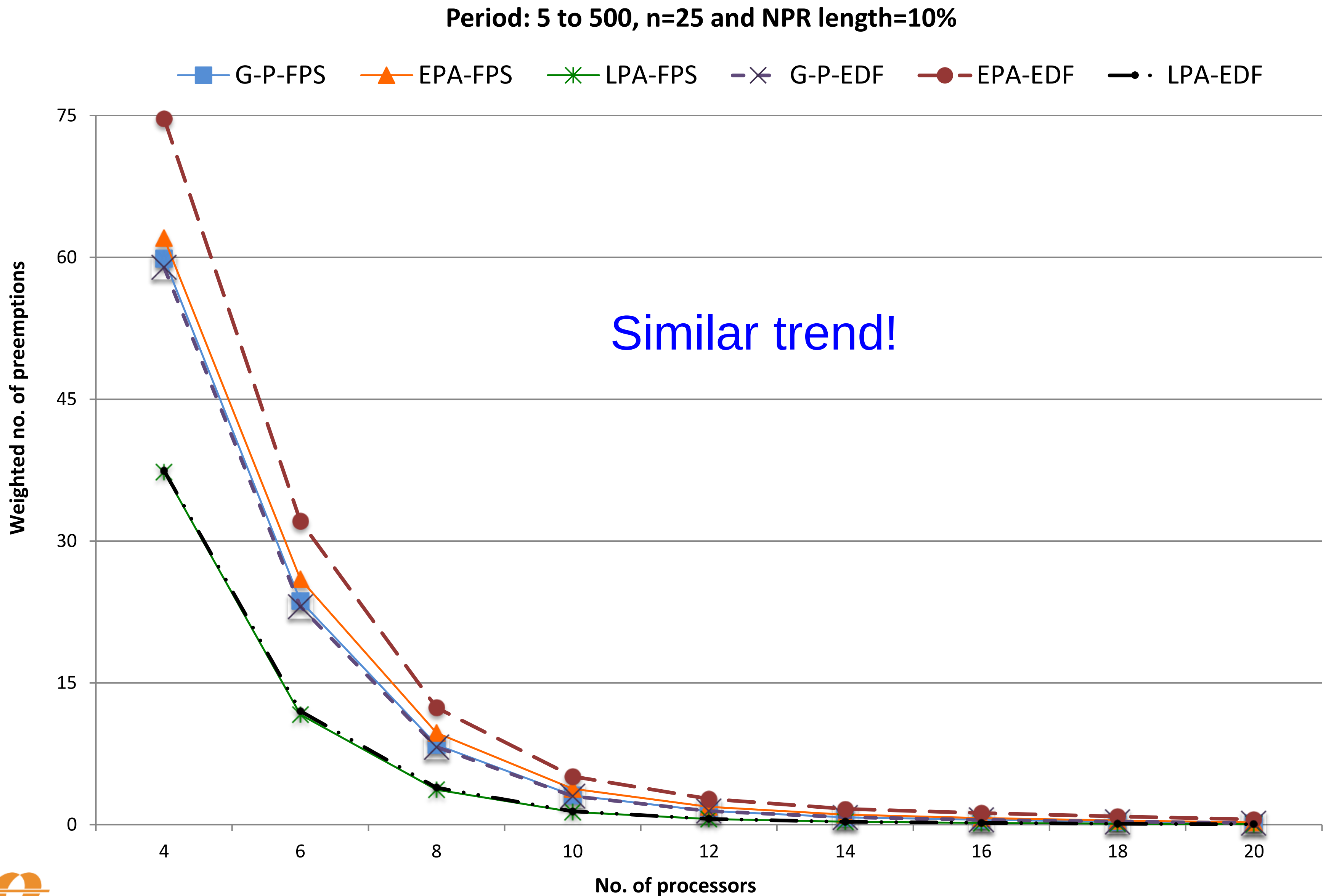
# Experiments

We investigated how *weighted preemptions* vary with:

1. Varying utilizations
2. Varying number of tasks
3. Varying number of processors
4. Varying NPR lengths

MÄLARDALEN UNIVERSITY SWEDEN

# Varying Number of Processors



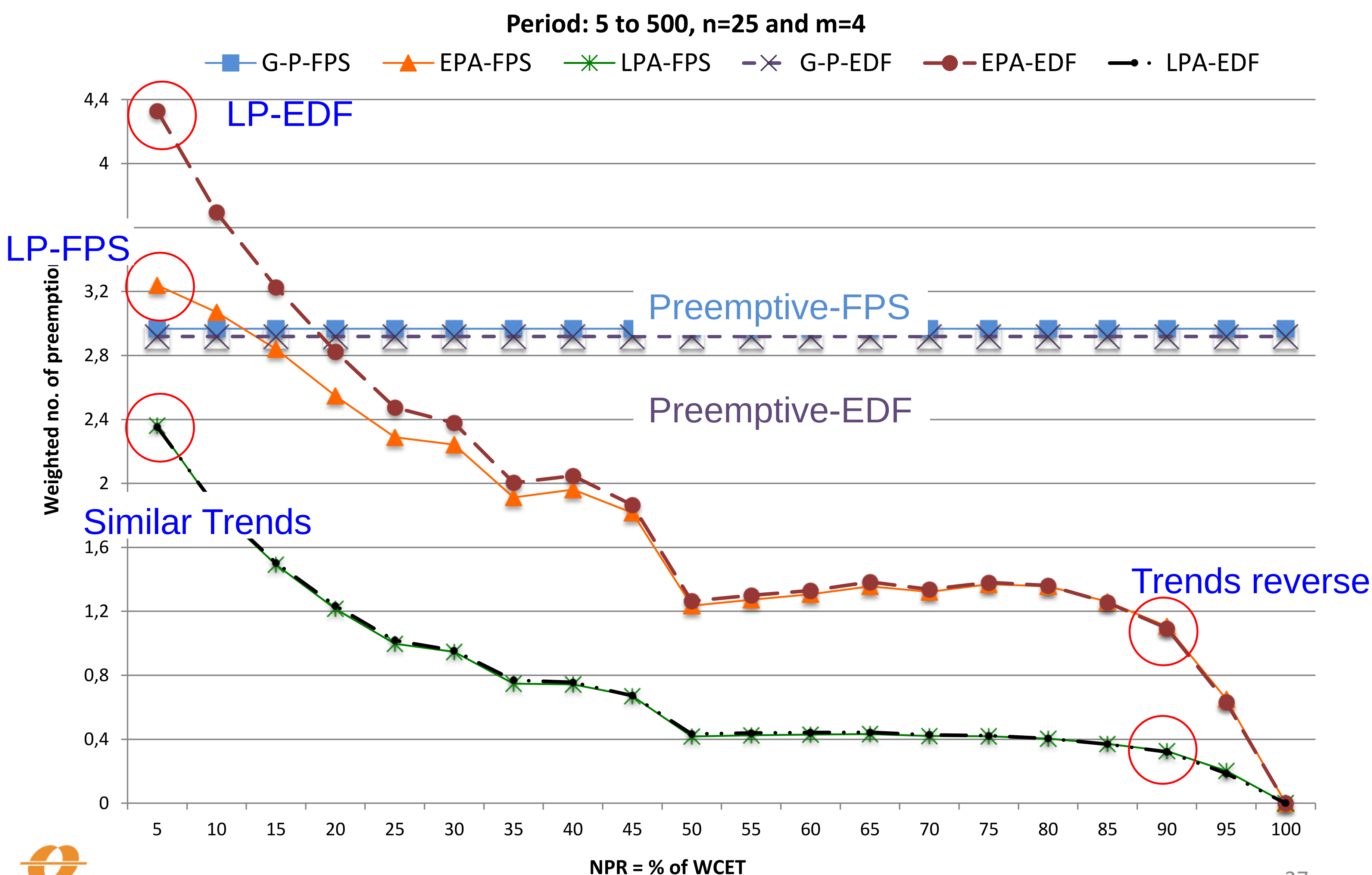
# Experiments

We investigated how *weighted preemptions* vary with:

1. Varying utilizations
2. Varying number of tasks
3. Varying number of processors
4. Varying NPR lengths

MÄLARDALEN UNIVERSITY
SWEDEN

# Varying Lengths of NPRs



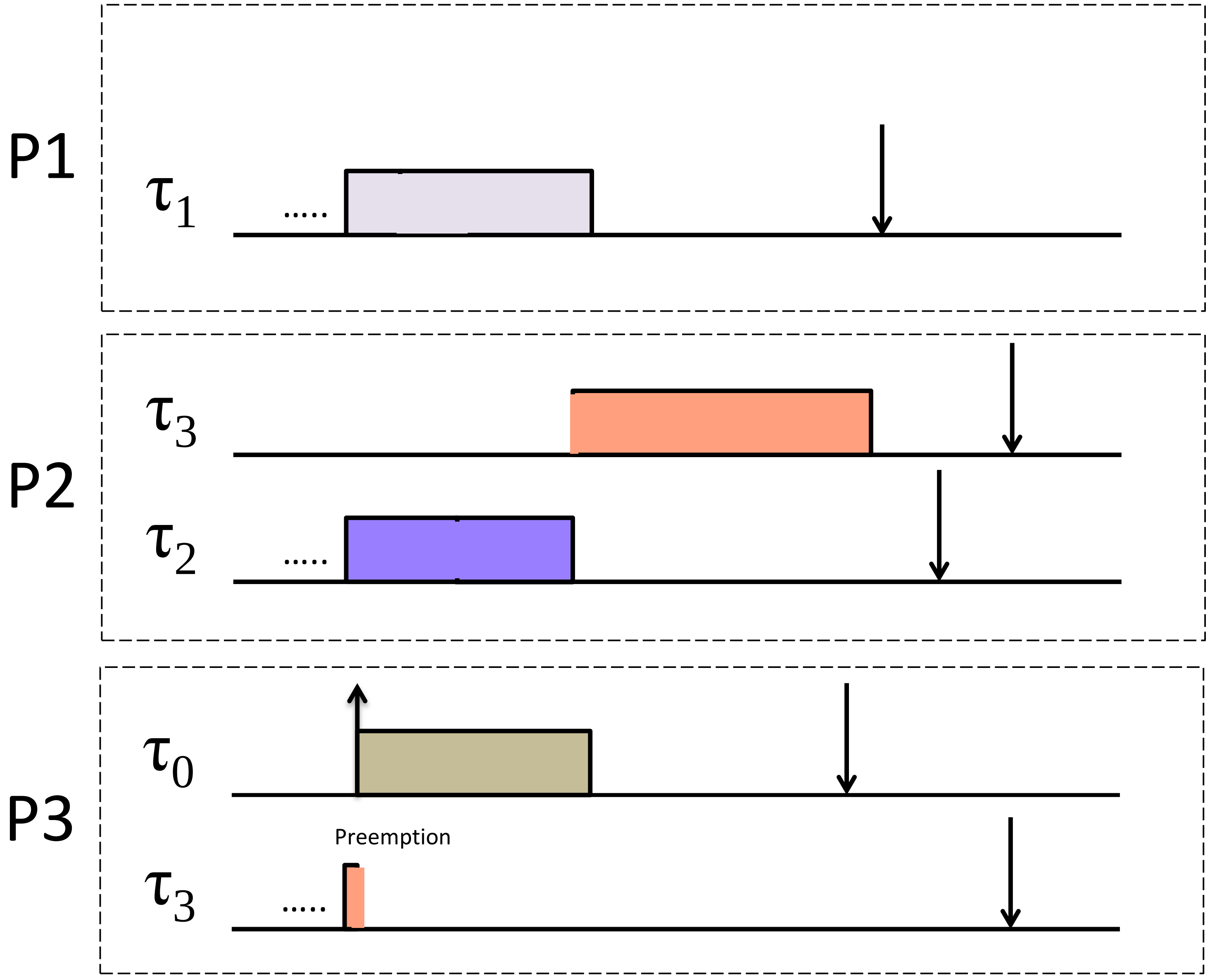

MÄLARDALEN UNIVERSITY SWEDEN

# Observations

1. Eager preemption approach generates most preemptions while lazy preemption approach generates least

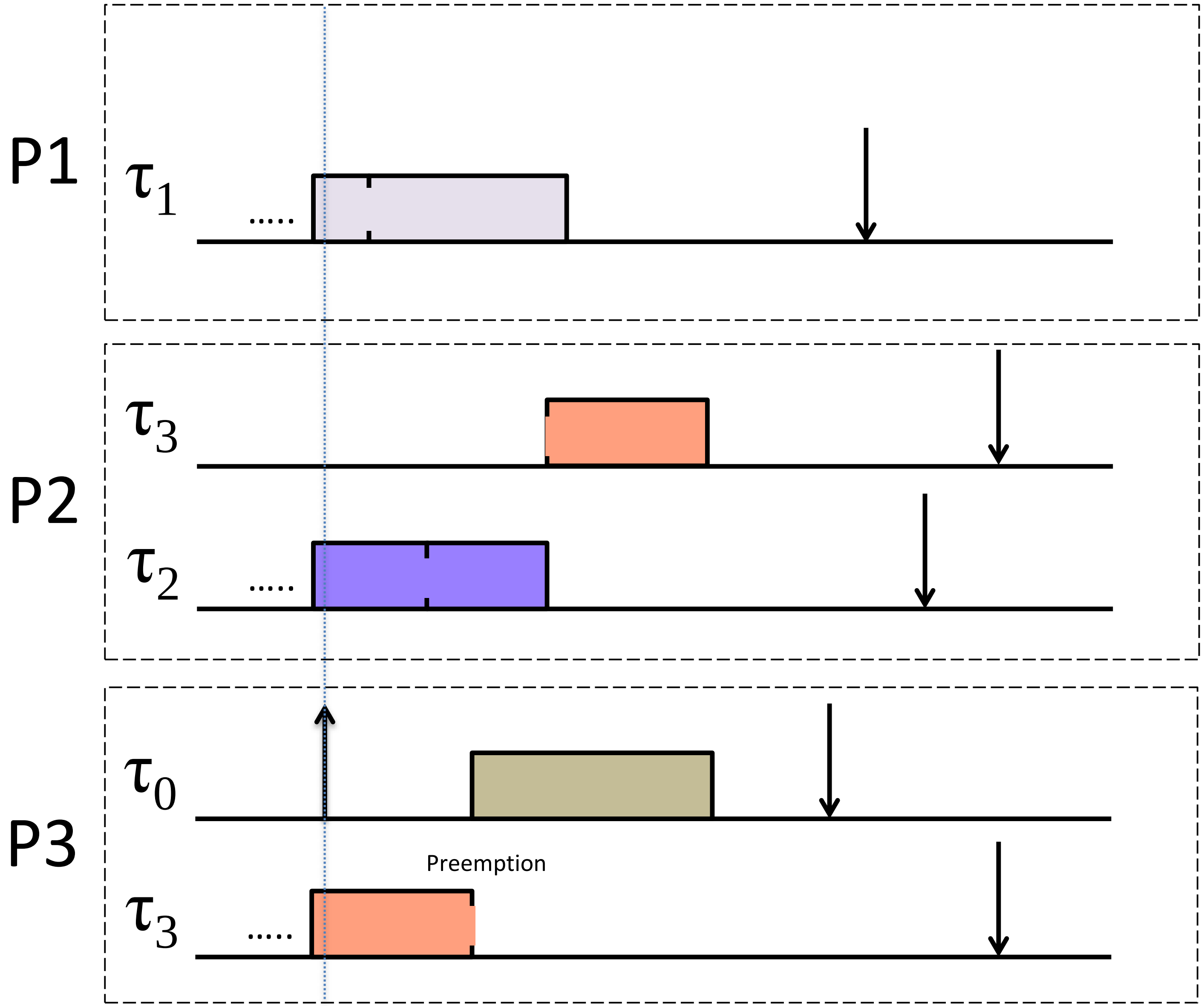
MÄLARDALEN UNIVERSITY SWEDEN

# Preemptive Scheduling

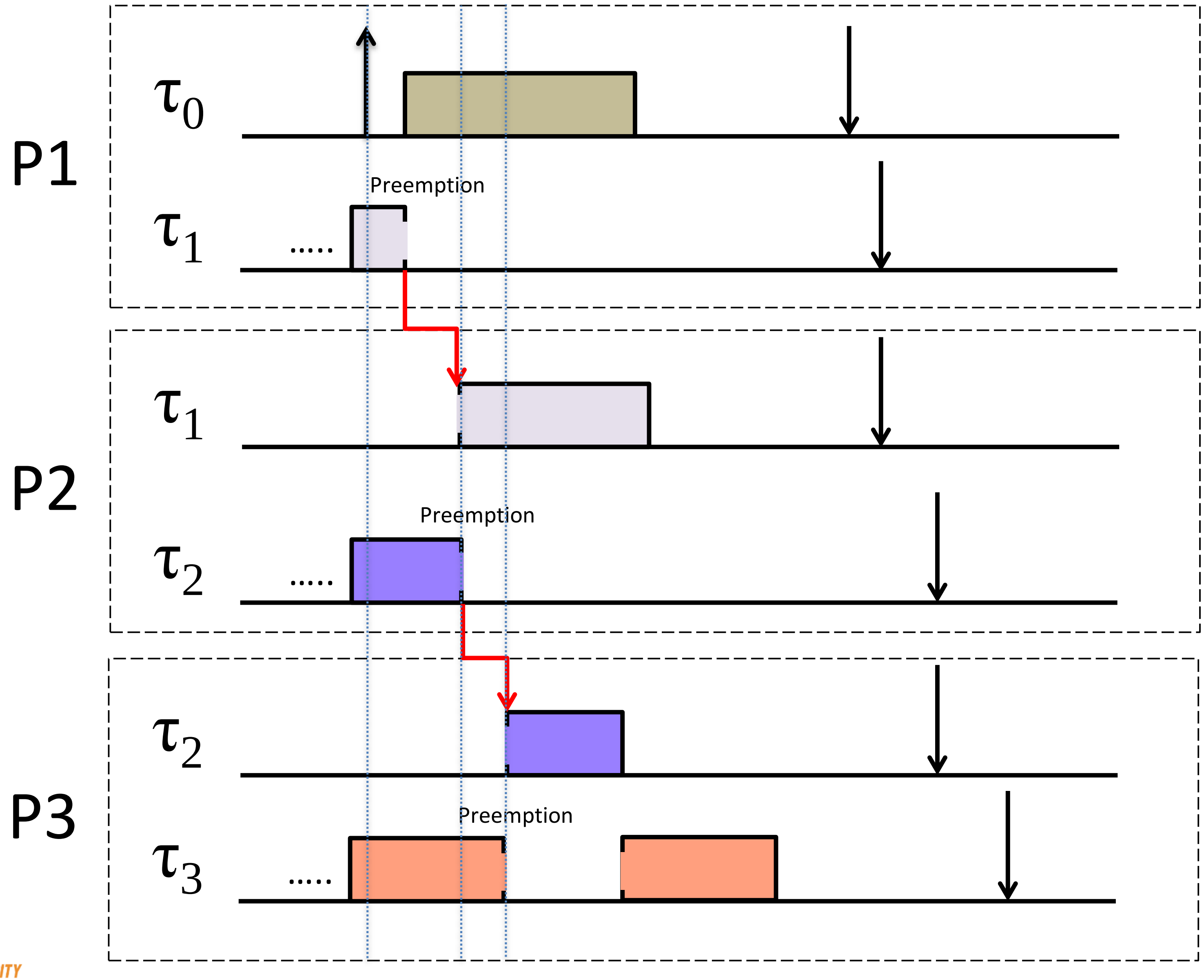



MÄLARDALEN UNIVERSITY SWEDEN

# Lazy Preemption Approach



# Eager Preemption Approach



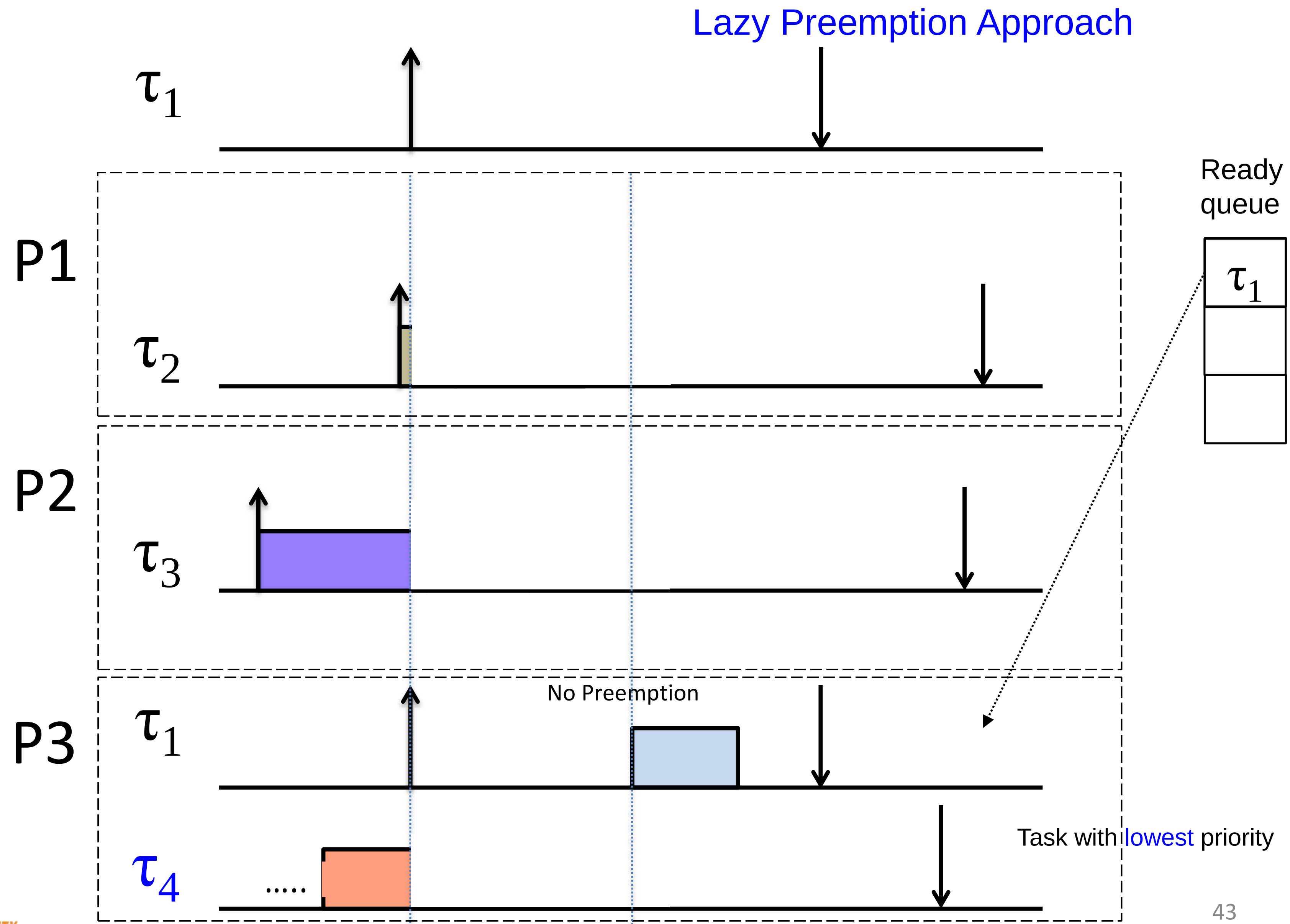
# Observations

1. Eager preemption approach generates most preemptions while lazy preemption approach generates least

2. Preemptive EDF generates fewer preemptions than preemptive FPS

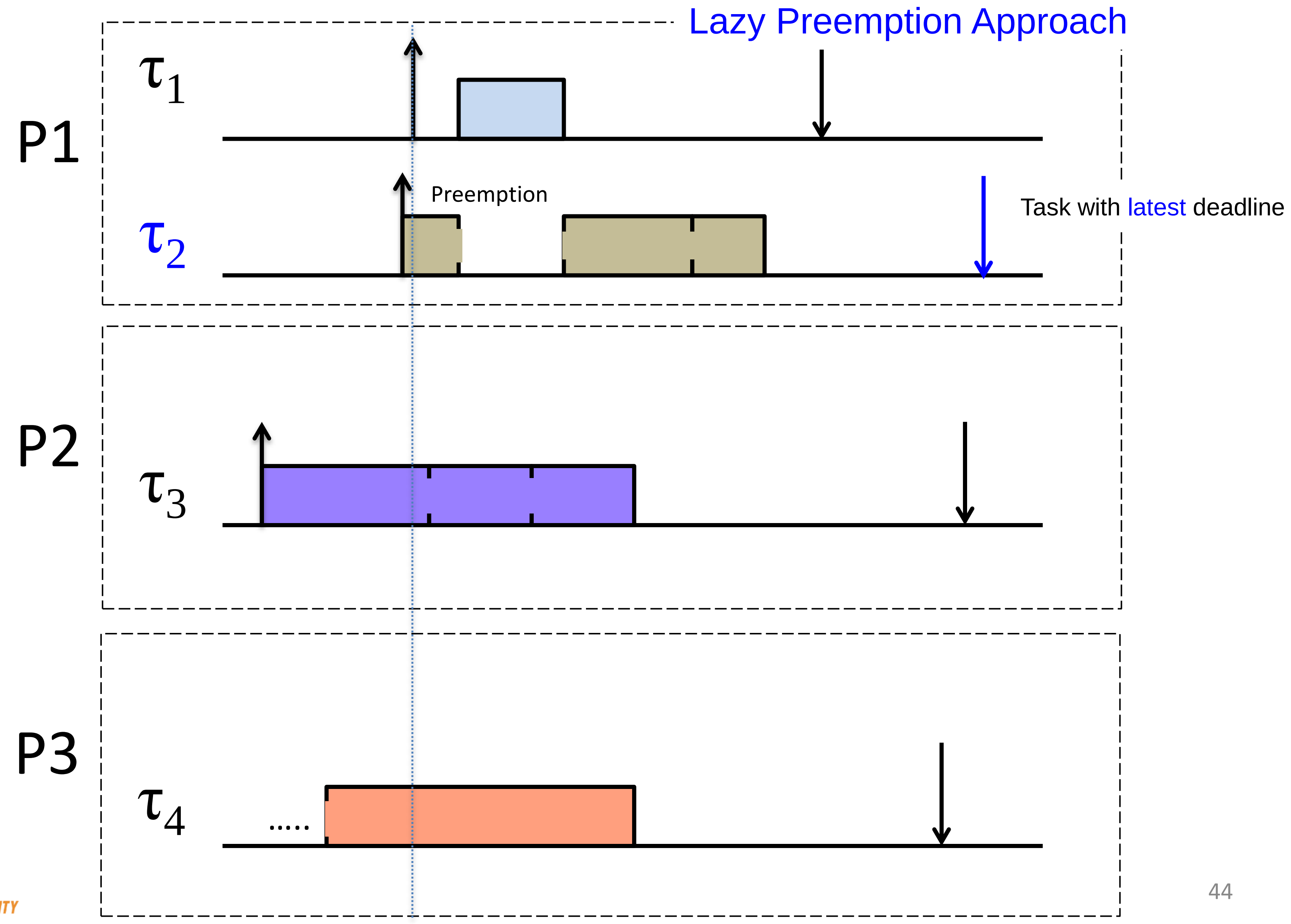
3. Limited preemptive EDF generates more preemptions that limited preemptive FPS

# Preemptive Behavior under FPS




MÄLARDALEN UNIVERSITY SWEDEN

# Preemptive Behavior under EDF



# Conclusions

1. Limited preemptive scheduling on multiprocessors may not reduce the number of preemptions
   - Number of preemptions larger than fully preemptive scheduling under the eager approach
2. The preemptive behavior of EDF on uniprocessors generalizes to multiprocessors
   - G-P-EDF generates fewer preemptions than G-P-FPS
3. This, however, does not generalize to global limited preemptive scheduling
   - G-LP-EDF generates more preemptions than G-LP-FPS
4. G-LP-FPS with LPA generates the least number of preemptions

MÄLARDALEN UNIVERSITY SWEDEN

# Future Work

- Hybrid approach to manage preemptions on multicores to get best of eager and lazy approaches: integration with preemption thresholds
- Probabilistic schedulability analysis for mixed-criticality tasks with fixed preemption points
- Improved preemption overhead accounting for tasks with fixed preemption points

# Thank you !

# Questions/Comments ?

MÄLARDALEN UNIVERSITY
SWEDEN



web images