Real-time scheduling of dynamic neural networks on multi-Edge TPU

Tomasz Kłoda

LAAS-CNRS / Insa de Toulouse
Toulouse, France

Huawei 2012 Labs Global Software Technology Summit 2024
Dresden 02.07.2024

Binqi Sun

Tomasz Kłoda

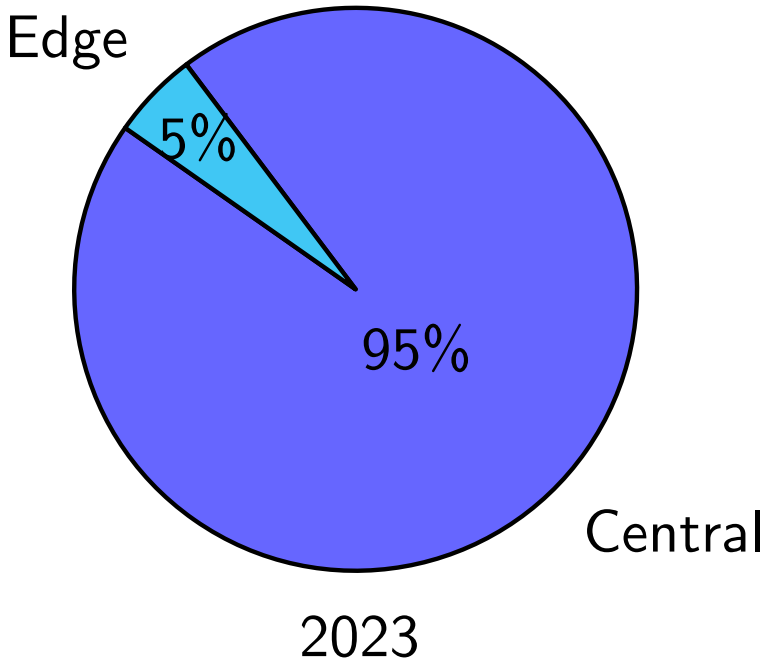
Marco Caccamo

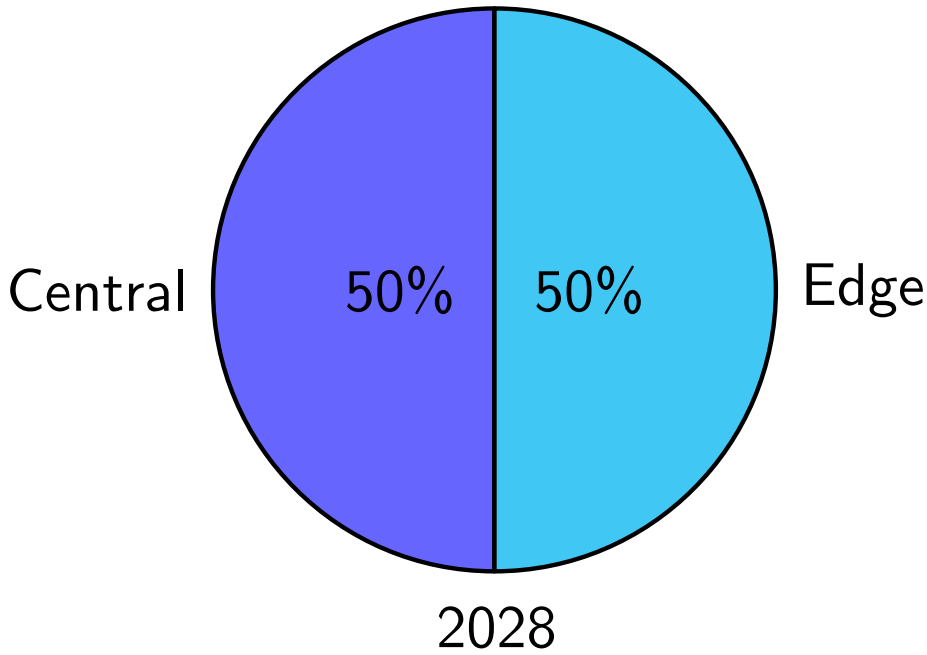
Chu-ge Wu, Jiyang Chen, Cen Lu, Bohua Zou

Technical University of Munich

LAAS CNRS

AI growth projection





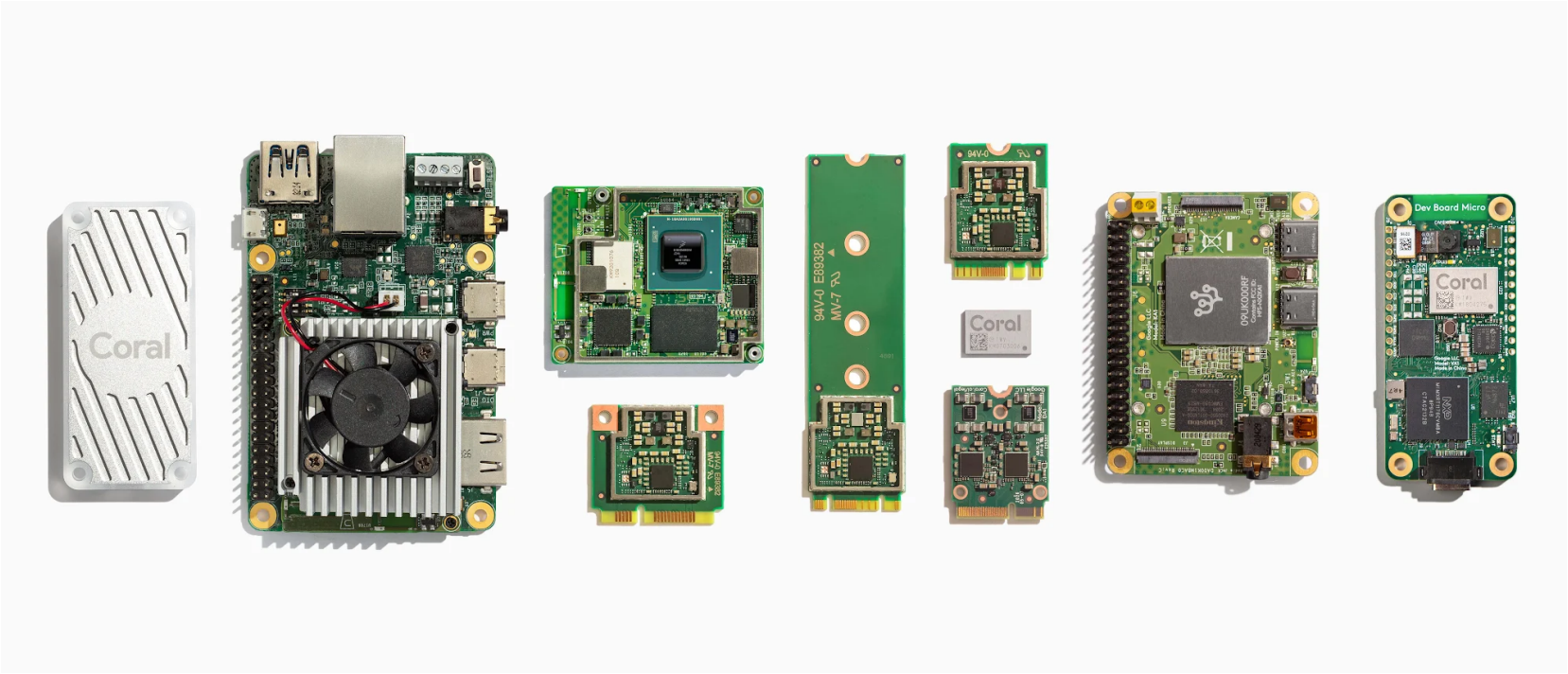
Schneider Electric

The AI Disruption: Challenges and Guidance for Data Center Design (2023)

Outline

Edge TPU

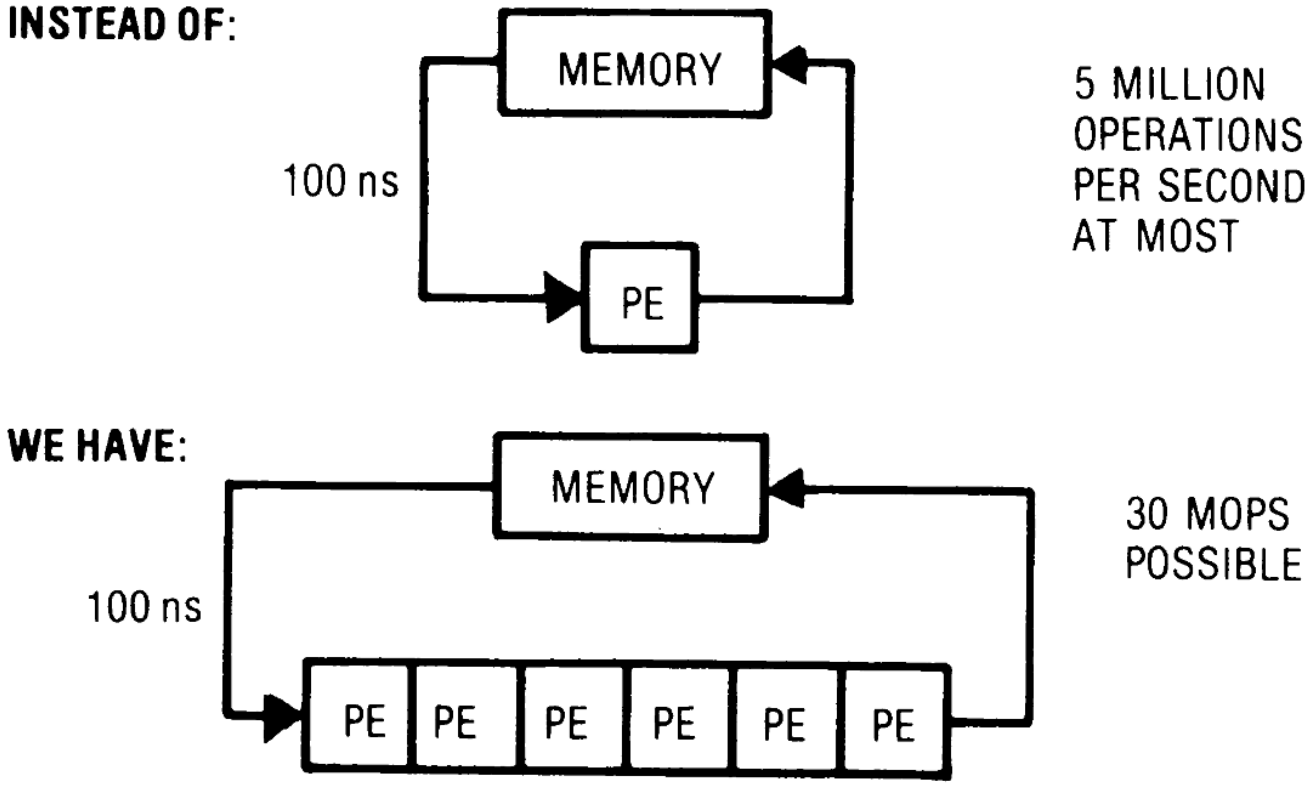
Edge TPU



	4 × Cortex-A53	Edge TPU
ResNet-50 V1	1763 ms	56 ms
Inception V1	392 ms	4.1 ms
MobileNet V1	164 ms	2.4 ms
DenseNet	1032 ms	25 ms

Edge TPU

Systolic array



H.T. Kung

Why systolic architectures? (1982)

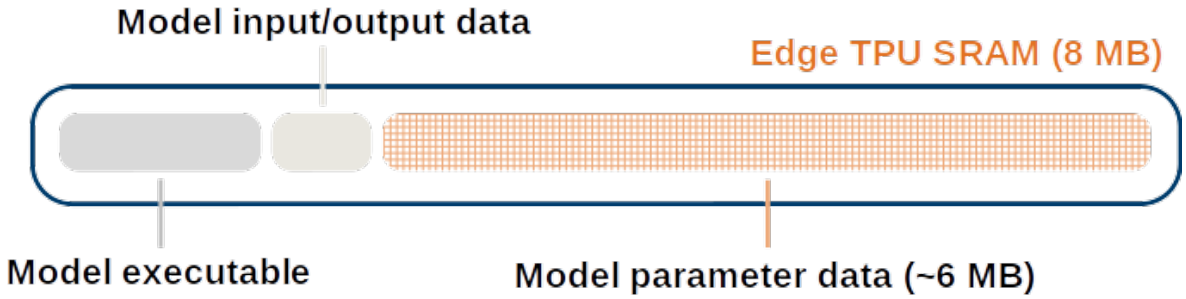
Edge TPU

SRAM On-chip memory

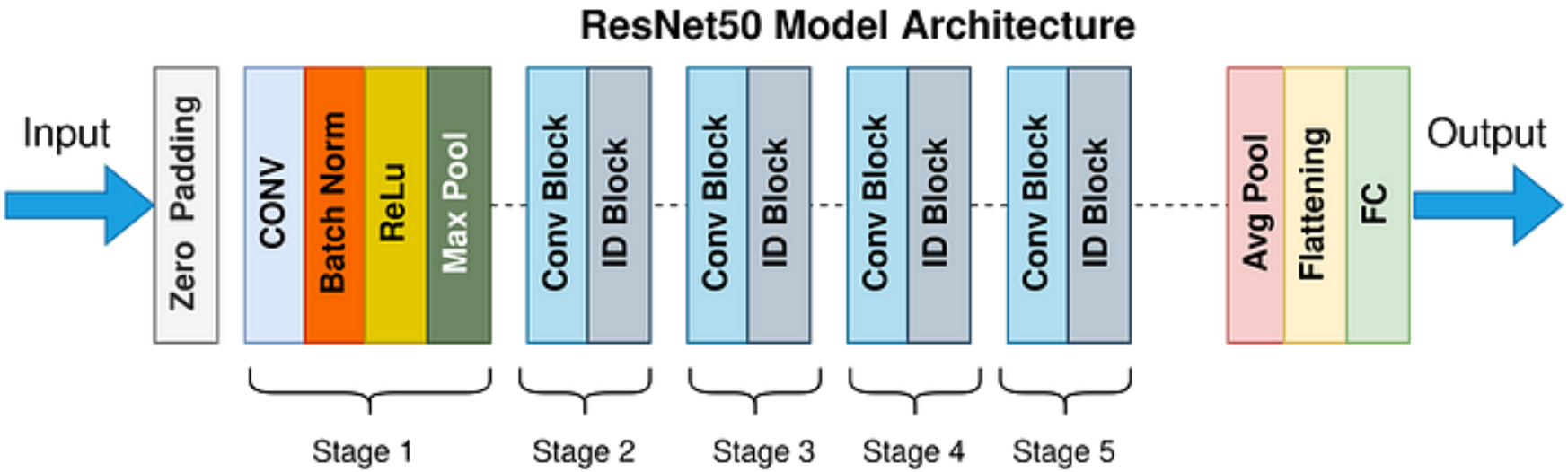
weights $\vec{w}$

inputs $\vec{x} \Rightarrow$ 

$\Rightarrow$outputs $\vec{y}$



Edge TPU pipelining



ResNet50

Input model: resnet_v1_50_100_quant.tflite

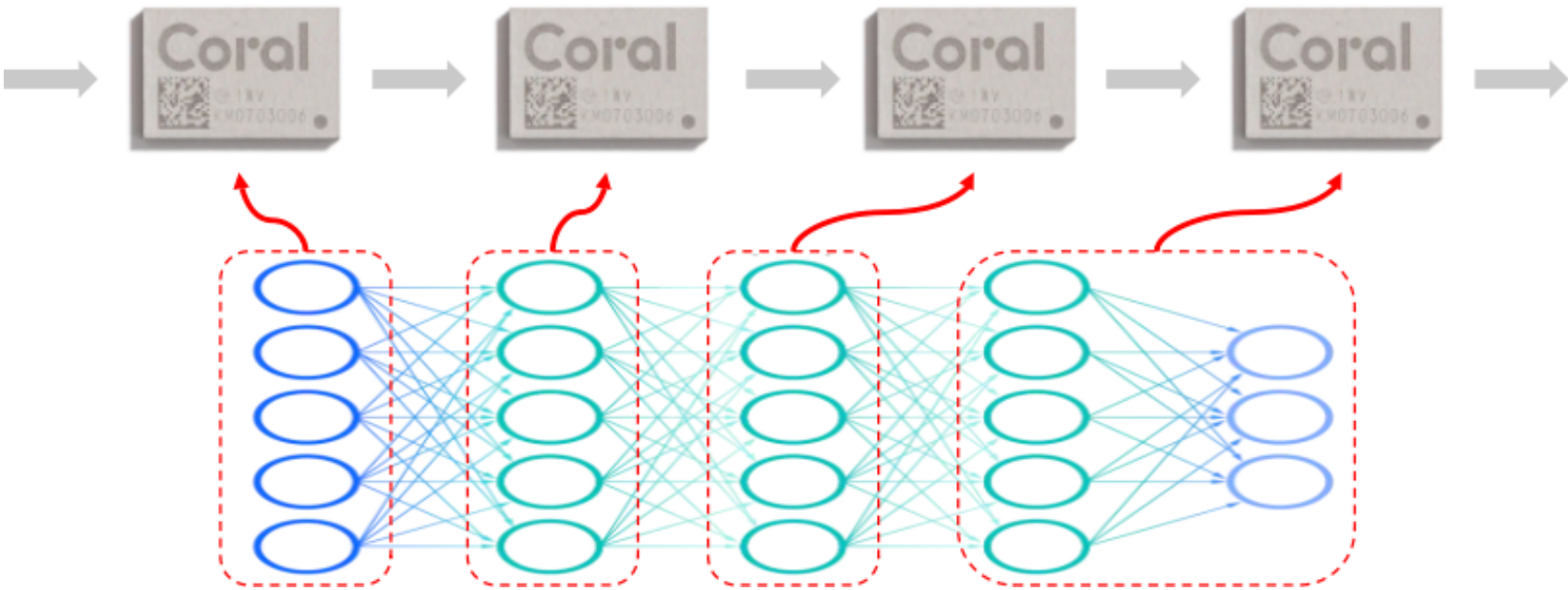
Output model: resnet_v1_50_100_quant_edgetpu.tflite

Output size: 22.92 MiB

On-chip memory used for caching model parameters: 7.35 MiB

Off-chip memory used for streaming uncached model parameters: 15.45 MiB

Edge TPU pipelining



Edge TPU pipelining

ResNet50

Input model: resnet_v1_50_100_quant_segment_0_of_4.tflite
On-chip memory used for caching model parameters: 5.33 MiB
Input model: resnet_v1_50_100_quant_segment_1_of_4.tflite
On-chip memory used for caching model parameters: 4.50 MiB
Input model: resnet_v1_50_100_quant_segment_2_of_4.tflite
On-chip memory used for caching model parameters: 5.30 MiB
Input model: resnet_v1_50_100_quant_segment_3_of_4.tflite
On-chip memory used for caching model parameters: 7.69 MiB
Off-chip memory used for streaming uncached model parameters: 0.00 B

Edge multi-TPU neural network benchmarks

ASUS AI Accelerator

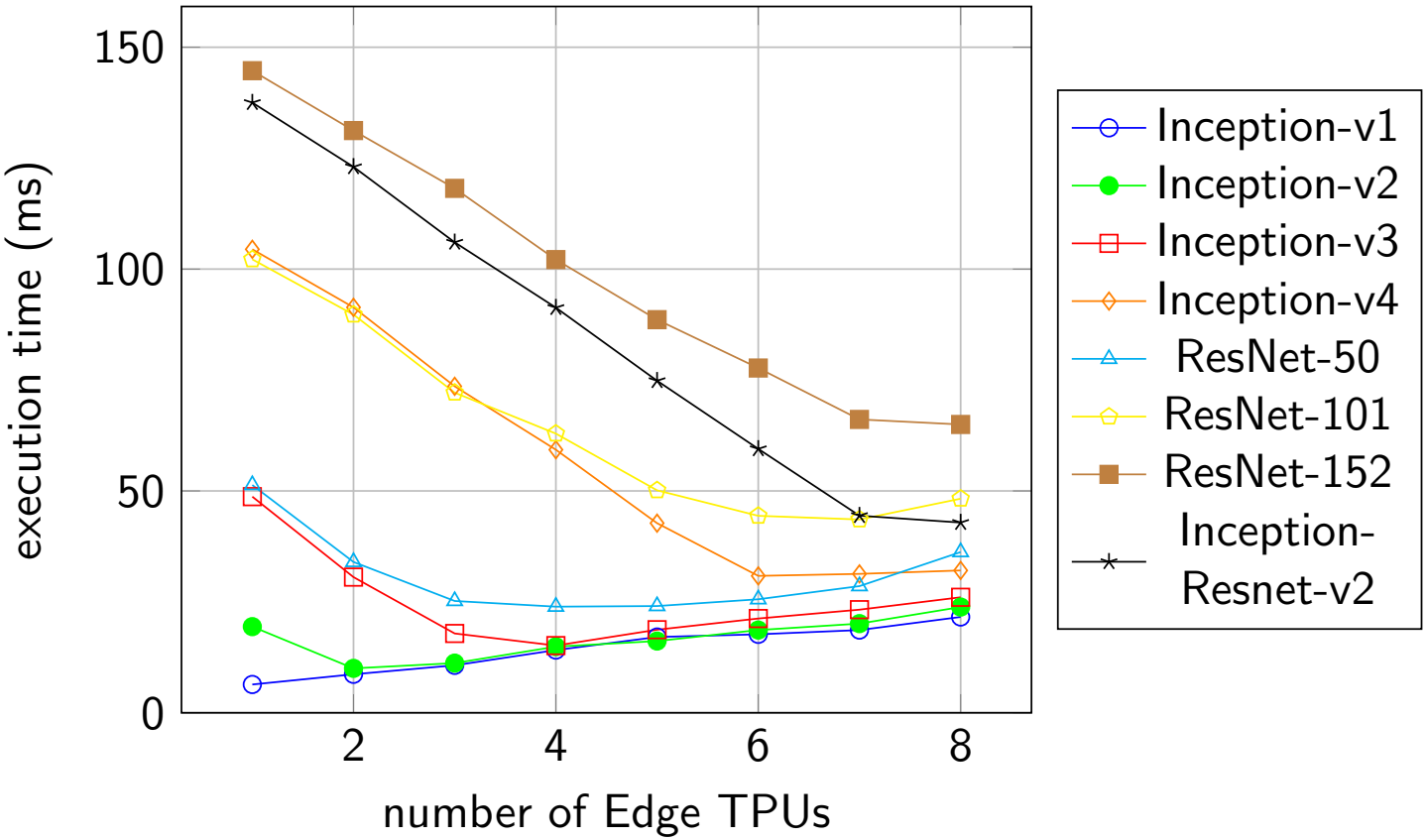
CRL-G18U-P3D / CRL-G116U-P3D

- Up to 8-16 Edge TPUs
- PCI Express 3.0
- TensorFlow Lite
- Precision INT8

Binqi Sun, Tomasz Kloda, Jiyang Chen, Cen Lu and Marco Caccamo.
Schedulability Analysis of Non-preemptive Sporadic Gang Tasks on Hardware Accelerators.
Real-Time and Embedded Technology and Applications Symposium (RTAS), San Antonio, TX, USA, 2023

Edge multi-TPU neural network benchmarks

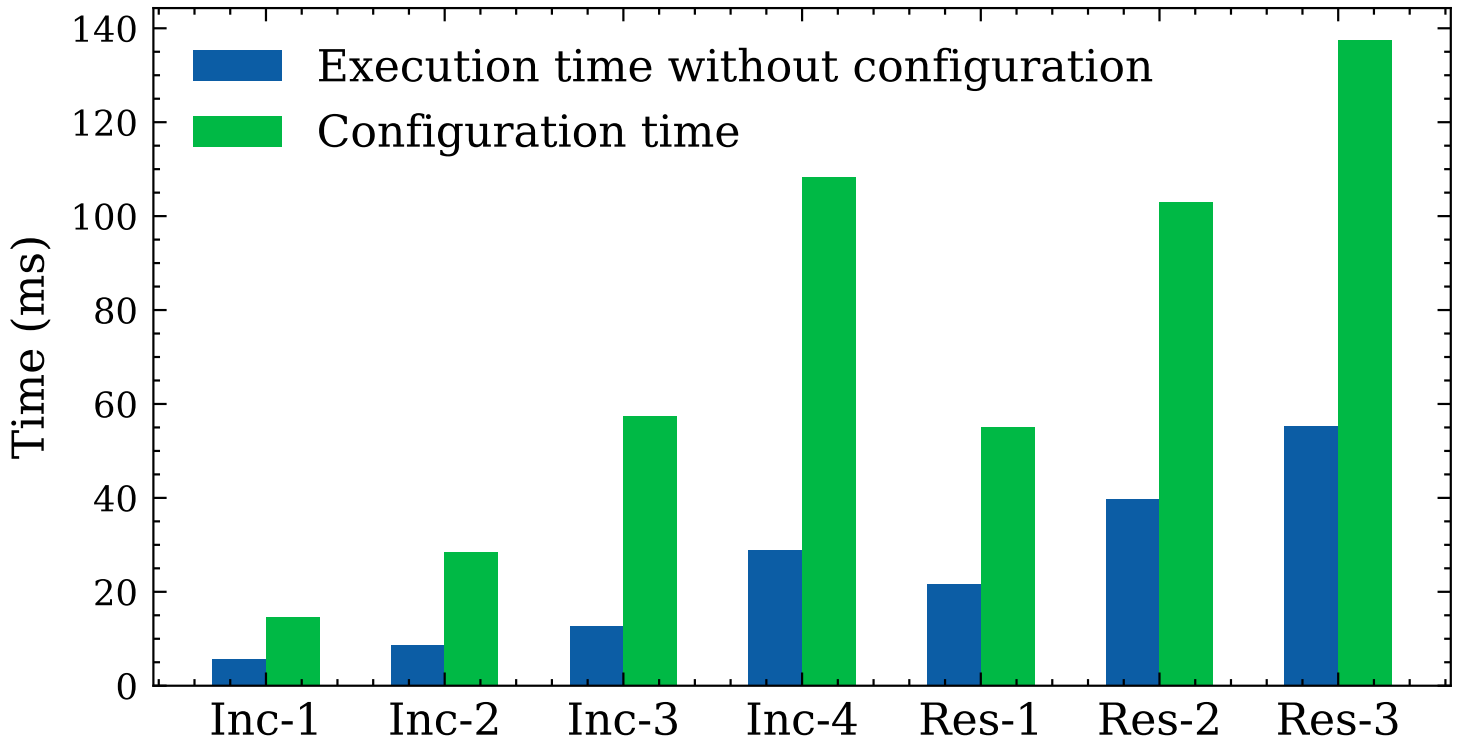
ASUS AI Accelerator



Binqi Sun, Tomasz Kloda, Jiyang Chen, Cen Lu and Marco Caccamo.

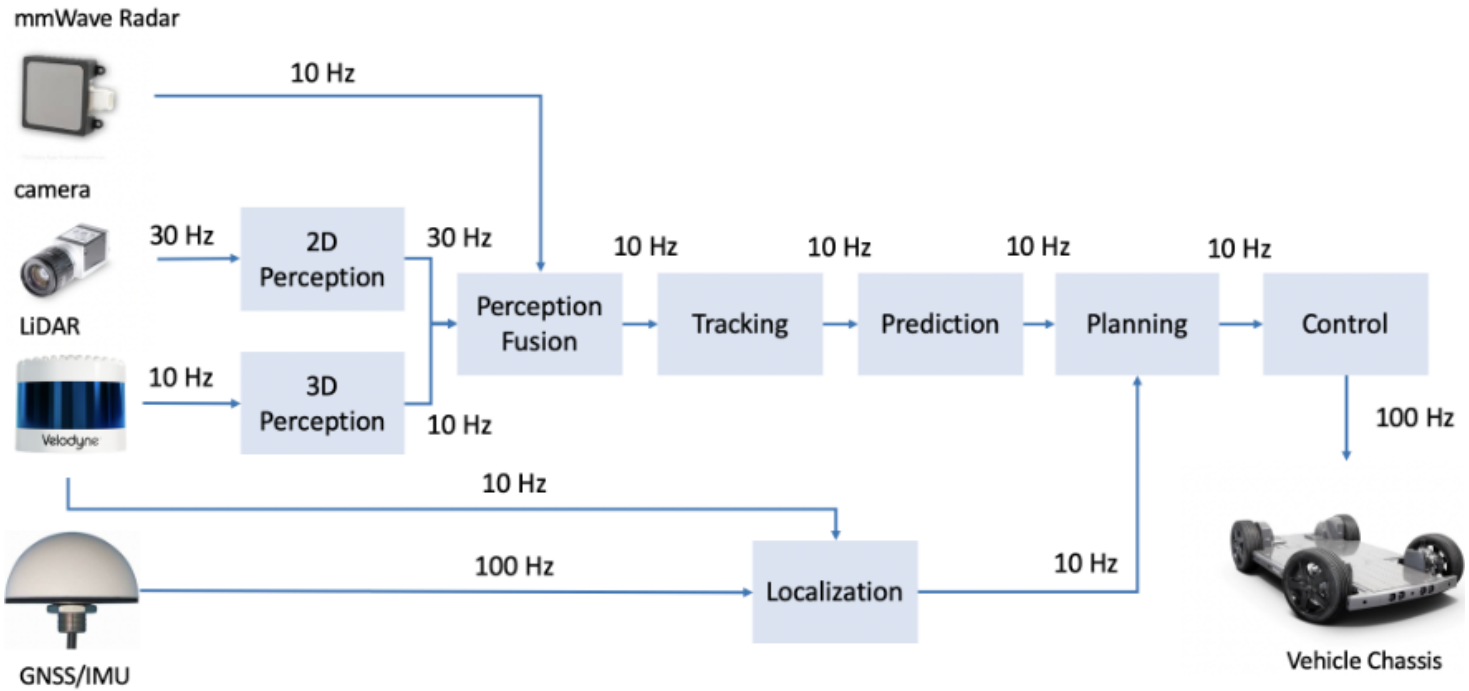
Schedulability Analysis of Non-preemptive Sporadic Gang Tasks on Hardware Accelerators.
Real-Time and Embedded Technology and Applications Symposium (RTAS), San Antonio, TX, USA, 2023

Parameter loading time overhead



Real-time scheduling

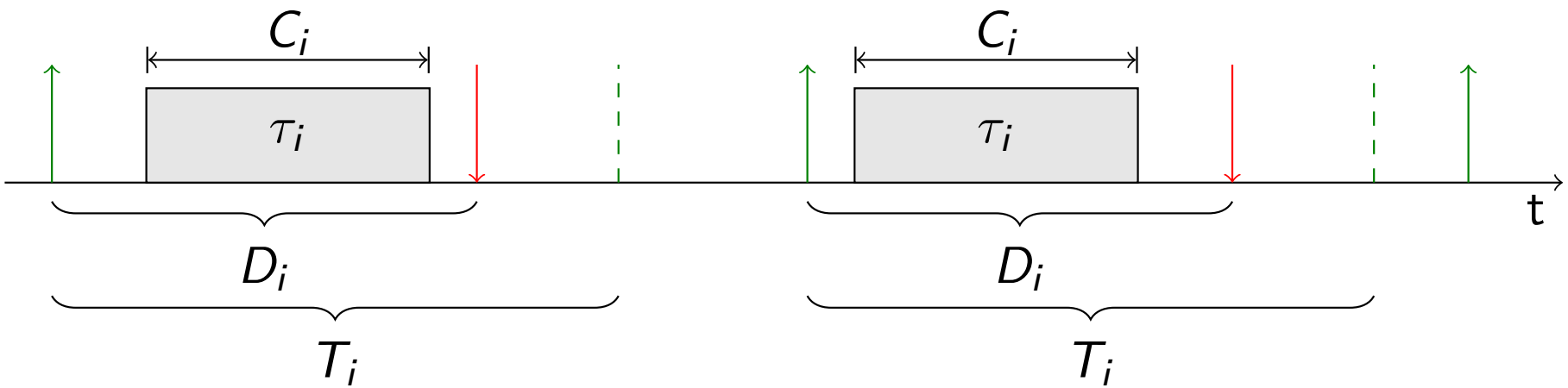
Real-time system



Real-time task model

Sequential task

- Each task τ_i can release its jobs anytime
- Two subsequent τ_i jobs are separated by at least T_i time units
- Each job of τ_i must complete within D_i time units



- Each job of τ_i executes for at most C_i time units ($WCET$)
- We define τ_i *processor utilization* as $u_i = C_i / T_i$

Real-time task scheduling

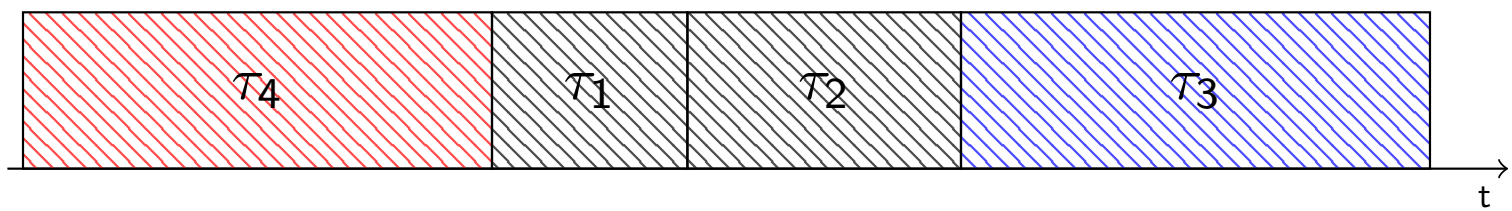
Fixed-priority non-preemptive scheduling

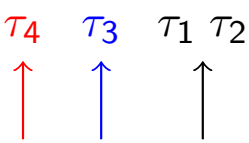
fixed-priority a priority is assigned to each task before execution and does not change over time

non-preemptive every job executes from its start uninterruptedly until completion

Schedulability test: $\forall i : R_i \leq D_i$
where is R_i is the task τ_i worst-case response time

Single, multi-processor and gang scheduling

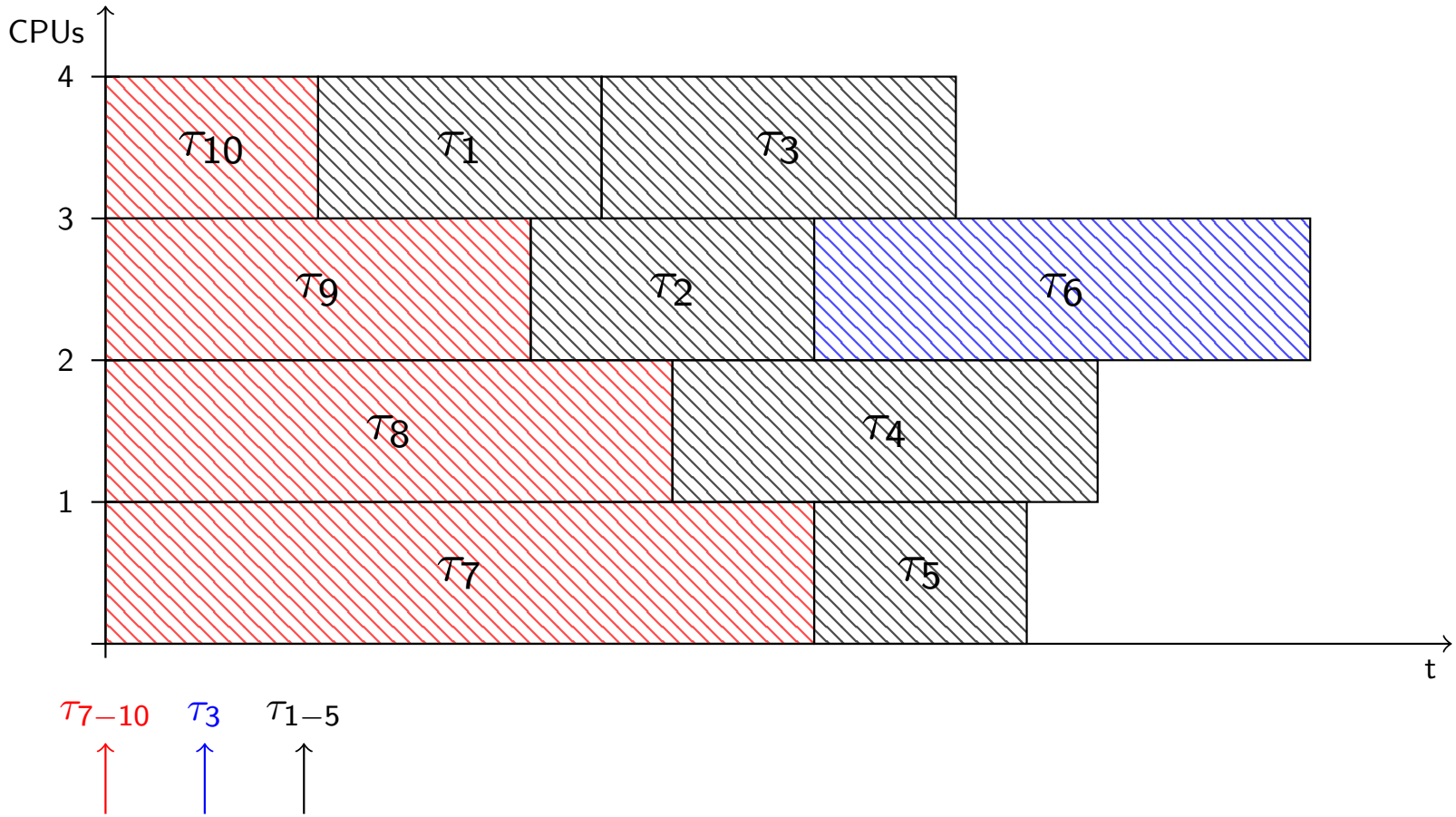




Robert I. Davis, Alan Burns, Reinder J. Bril, Johan J. Lukkien

Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised. (*Real Time Syst. 2007)*

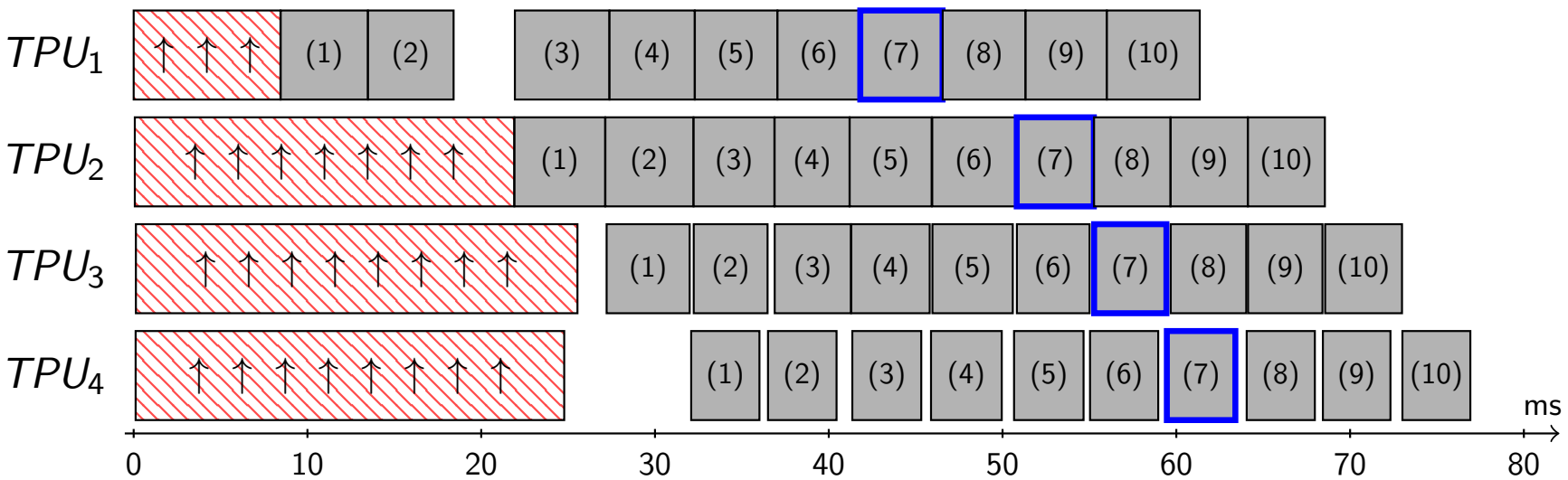
Single, **multi-processor** and gang scheduling



Nan Guan, Wang Yi, Qingxu Deng, Zonghua Gu and Ge Yu.
Schedulability analysis for non-preemptive fixed-priority multiprocessor scheduling. J. Syst. Archit. *2011*

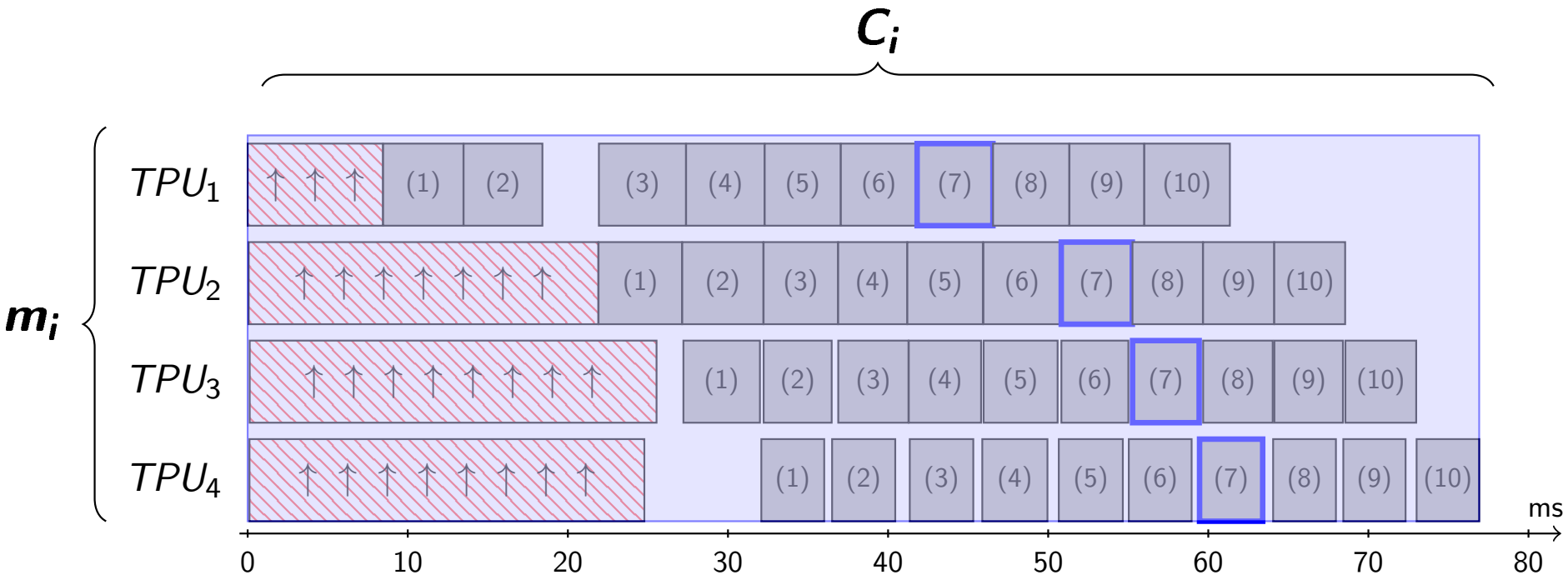
Single, multi-processor and **gang** scheduling

Gang task

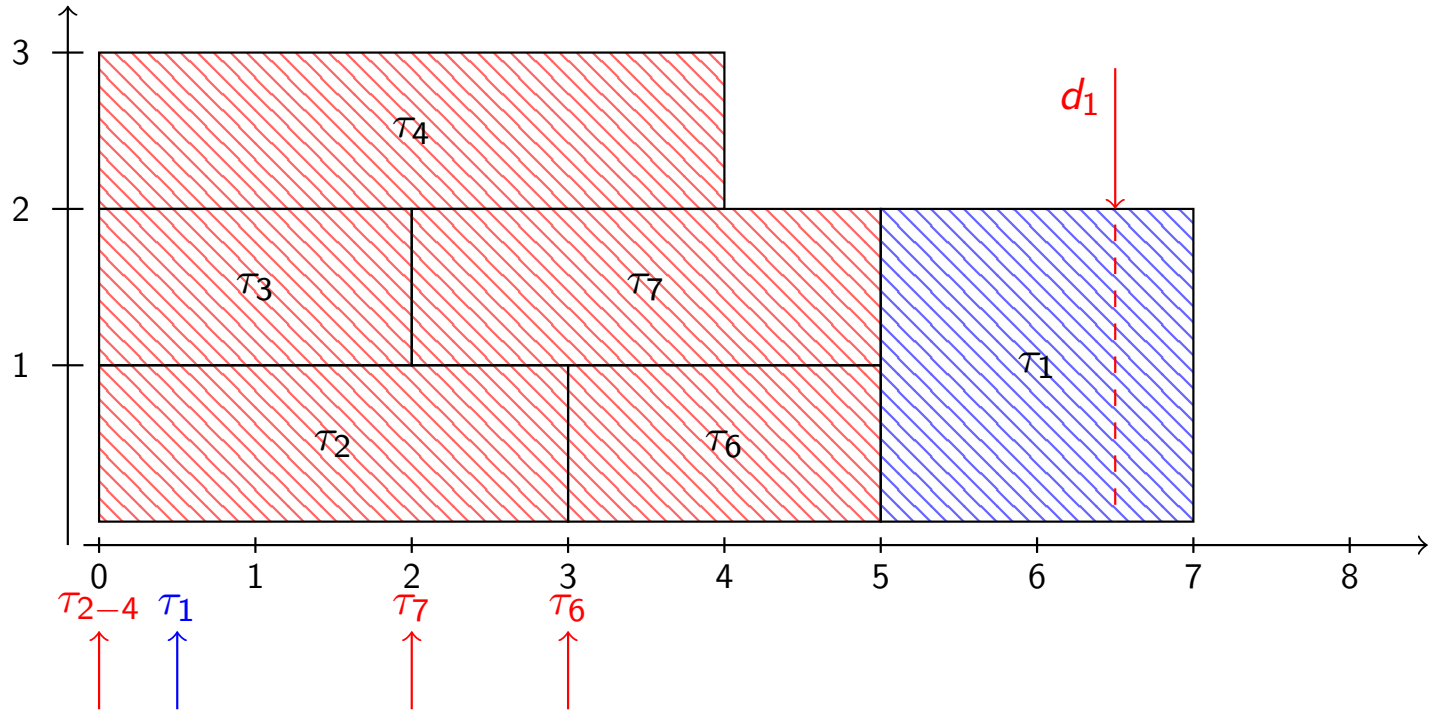


Single, multi-processor and **gang** scheduling

Gang task



Single, multi-processor and **gang** scheduling



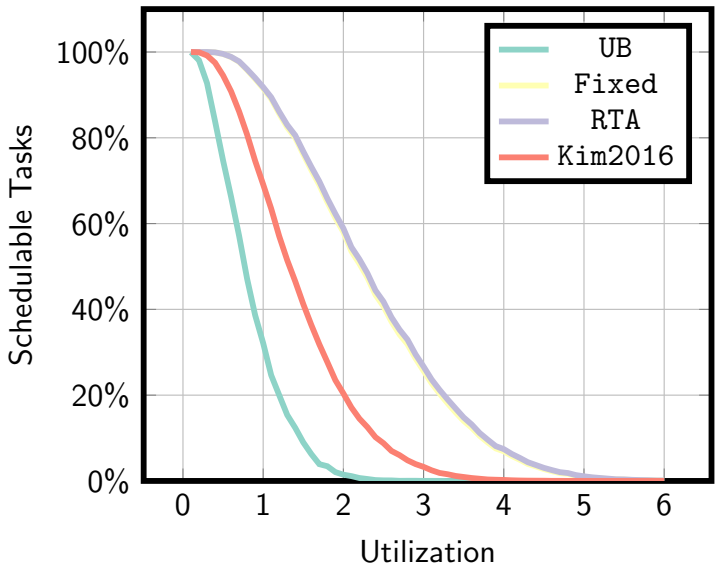
Z. Dong and C. Liu

Work-in-progress: Non-preemptive scheduling of sporadic gang tasks on multiprocessors. *RTSS 2019*

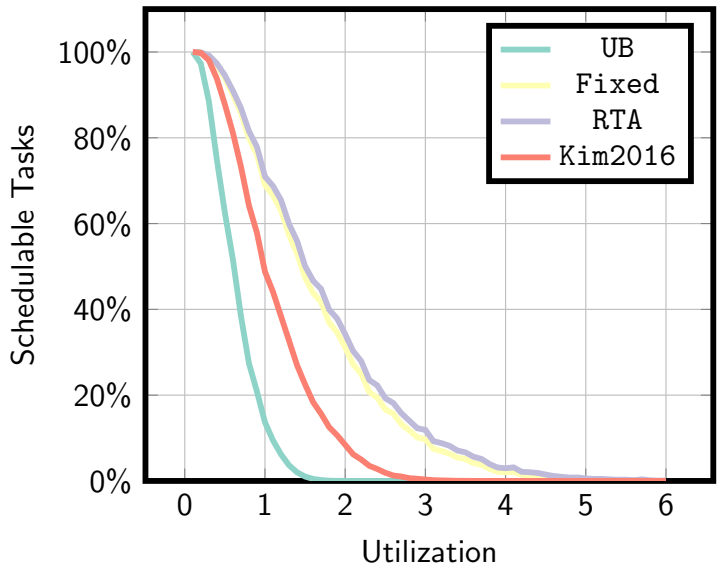
Binqi Sun, Tomasz Kloda, Jiyang Chen, Cen Lu, Marco Caccamo

Schedulability Analysis of Non-preemptive Sporadic Gang Tasks on Hardware Accelerators. *RTAS 2023*

Single, multi-processor and **gang** scheduling



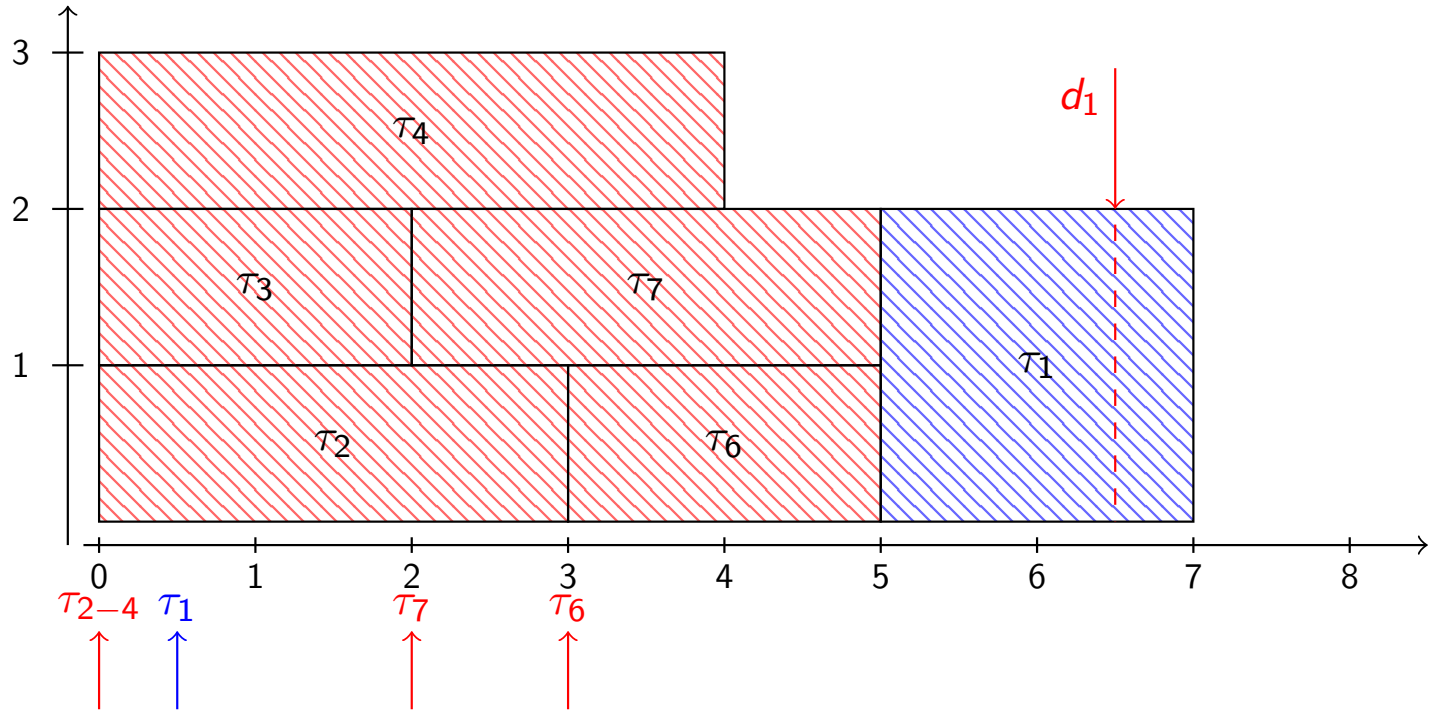
low parallelism level



high parallelism level

Binqi Sun, Tomasz Kloda, Jiyang Chen, Cen Lu and Marco Caccamo
Schedulability Analysis of Non-preemptive Sporadic Gang Tasks on Hardware Accelerators. *RTAS 2023*

Single, multi-processor and **gang** scheduling



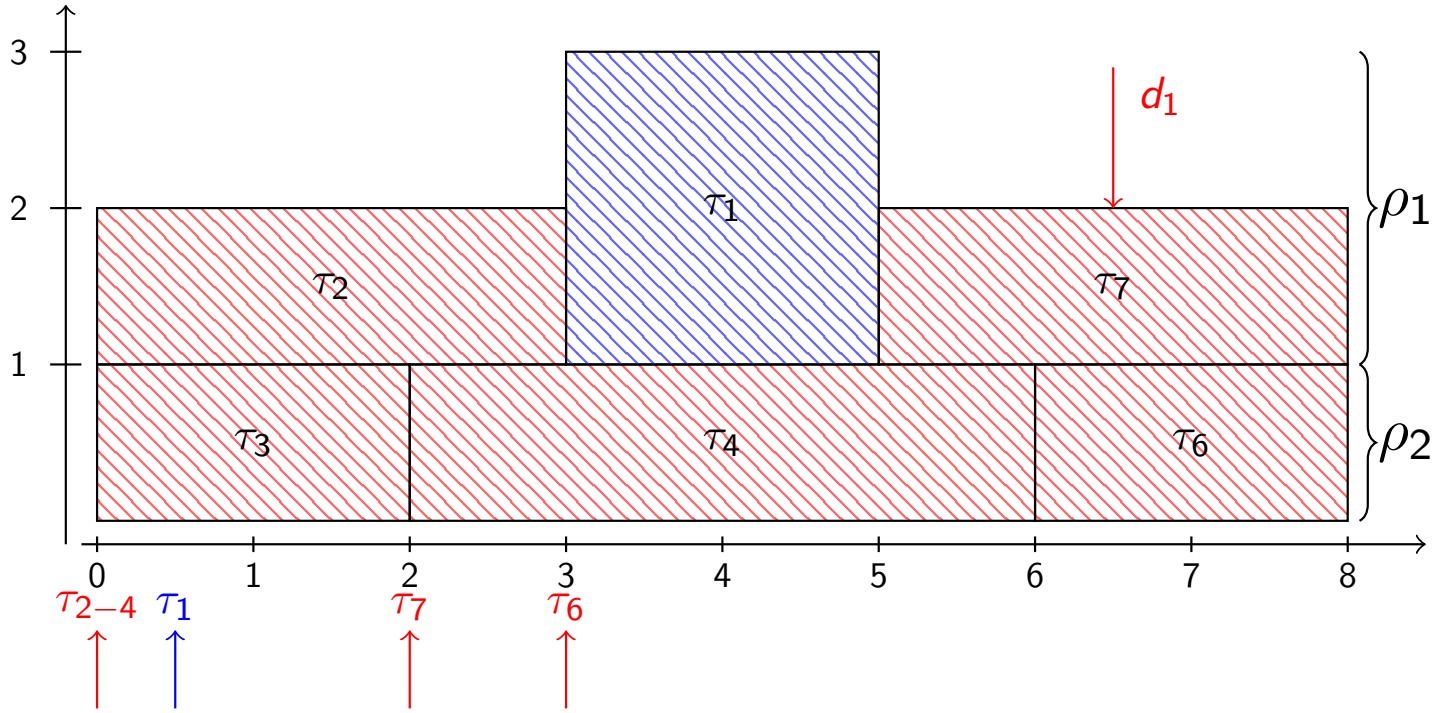
Z. Dong and C. Liu

Work-in-progress: Non-preemptive scheduling of sporadic gang tasks on multiprocessors. *RTSS 2019*

Binqi Sun, Tomasz Kloda, Jiyang Chen, Cen Lu, Marco Caccamo

Schedulability Analysis of Non-preemptive Sporadic Gang Tasks on Hardware Accelerators. *RTAS 2023*

Single, multi-processor and **gang** scheduling



Binqi Sun, Tomasz Kloda, Marco Caccamo

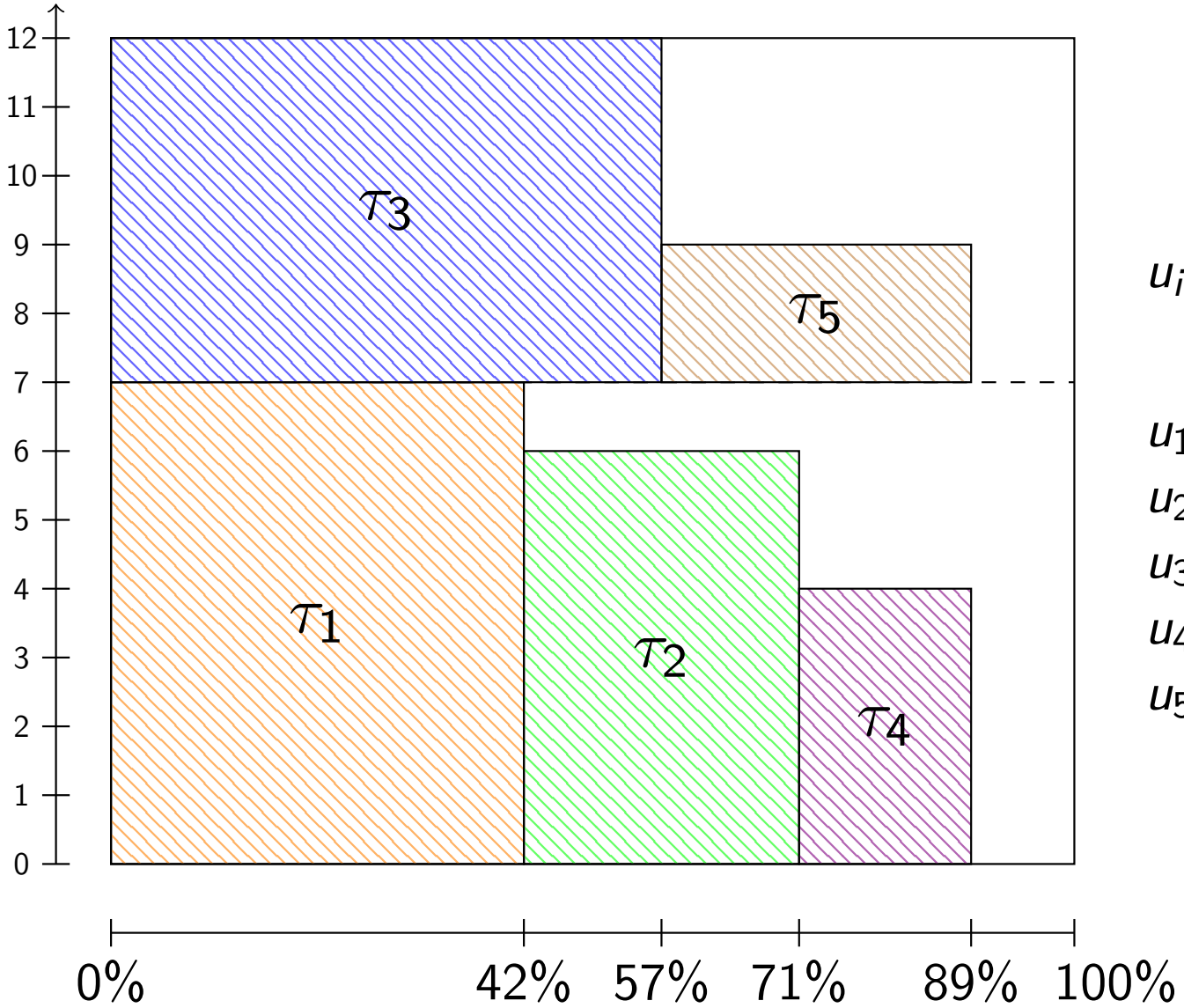
Strict Partitioning for Sporadic Rigid Gang Tasks *RTAS 2024*

Binqi Sun, Tomasz Kloda, Chu-ge Wu, Marco Caccamo

Partitioned Scheduling and Parallelism Assignment for Real-Time DNN Inference Tasks on Multi-TPU *DAC 2024*

Task to processor assignment

Strip packing



$u_i = C_i / T_i$

$u_1 = 21/50 = 42\%$
$u_2 = 29/100 = 19\%$
$u_3 = 114/200 = 57\%$
$u_4 = 45/250 = 18\%$
$u_5 = 160/500 = 32\%$

Task to processor assignment

Strip packing

- single processor non-preemptive fixed-priority:

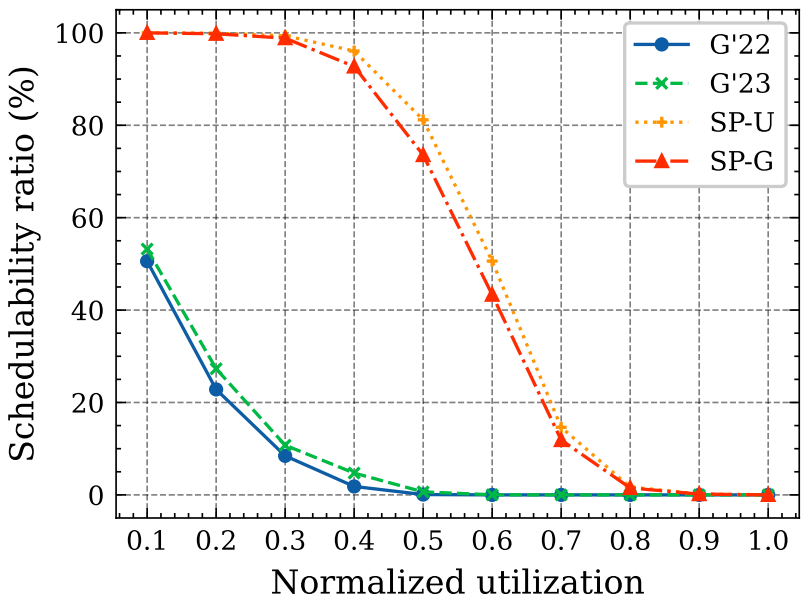
$$s_{i,l} = \max_{\tau_j \in lp(i)} C_j + (l-1) \cdot C_i + \sum_{\tau_j \in hp(i)} \left\lceil \frac{s_{i,l}}{T_j} \right\rceil \cdot C_j$$

$$R_i = s_{i,l} + C_i \leq D_i$$

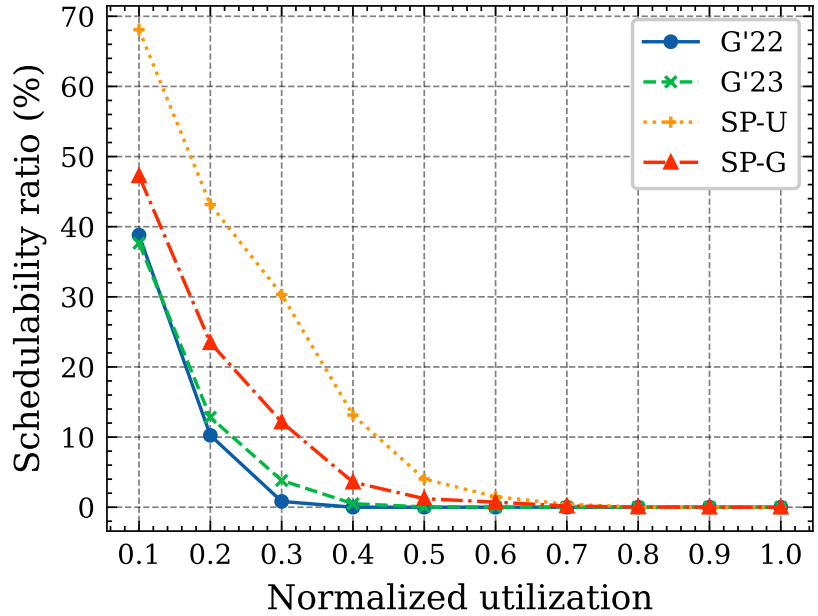
- can be any scheduling policy (*e.g.*, global)

Binqi Sun, Tomasz Kloda, Marco Caccamo
Strict Partitioning for Sporadic Rigid Gang Tasks *RTAS 2024*

Strict partitioning



low parallelism level

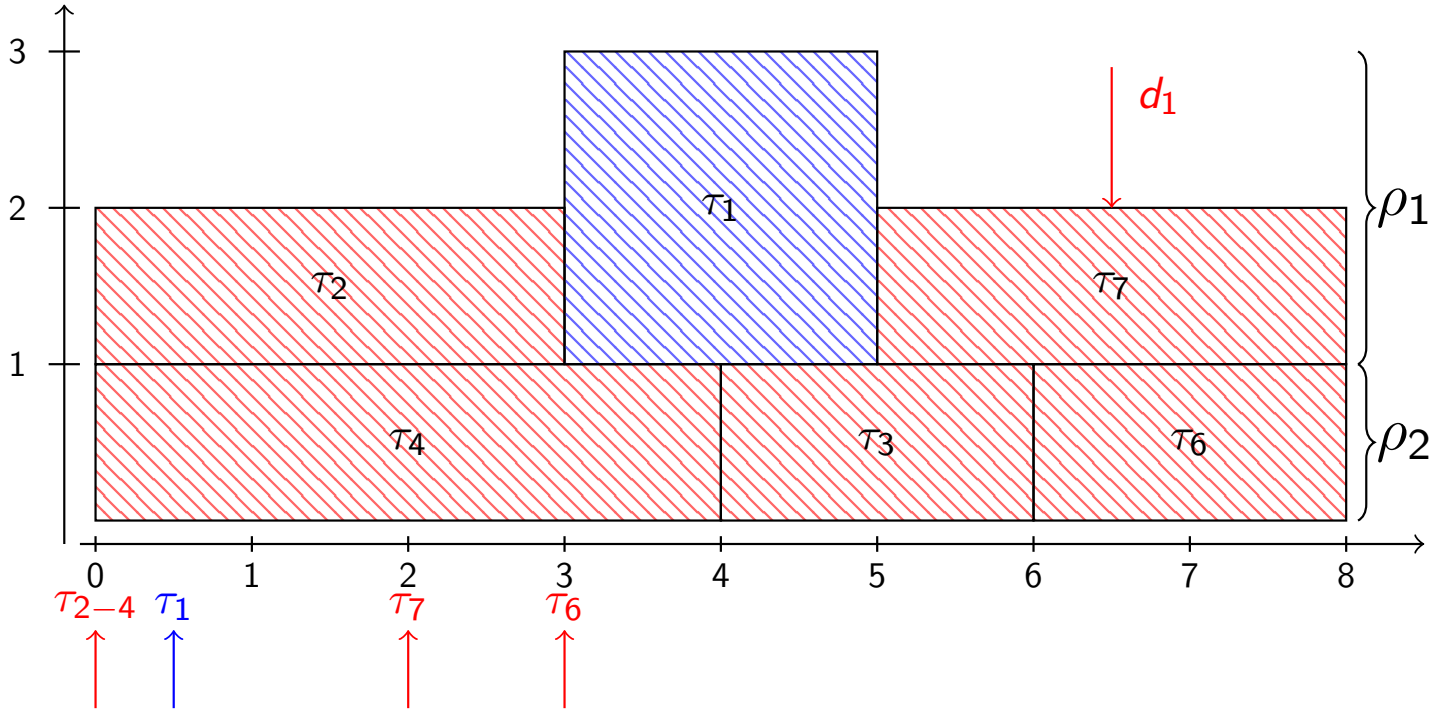


high parallelism level

Binqi Sun, Tomasz Kloda, Marco Caccamo

Strict Partitioning for Sporadic Rigid Gang Tasks *RTAS 2024*

Parallelism assignment

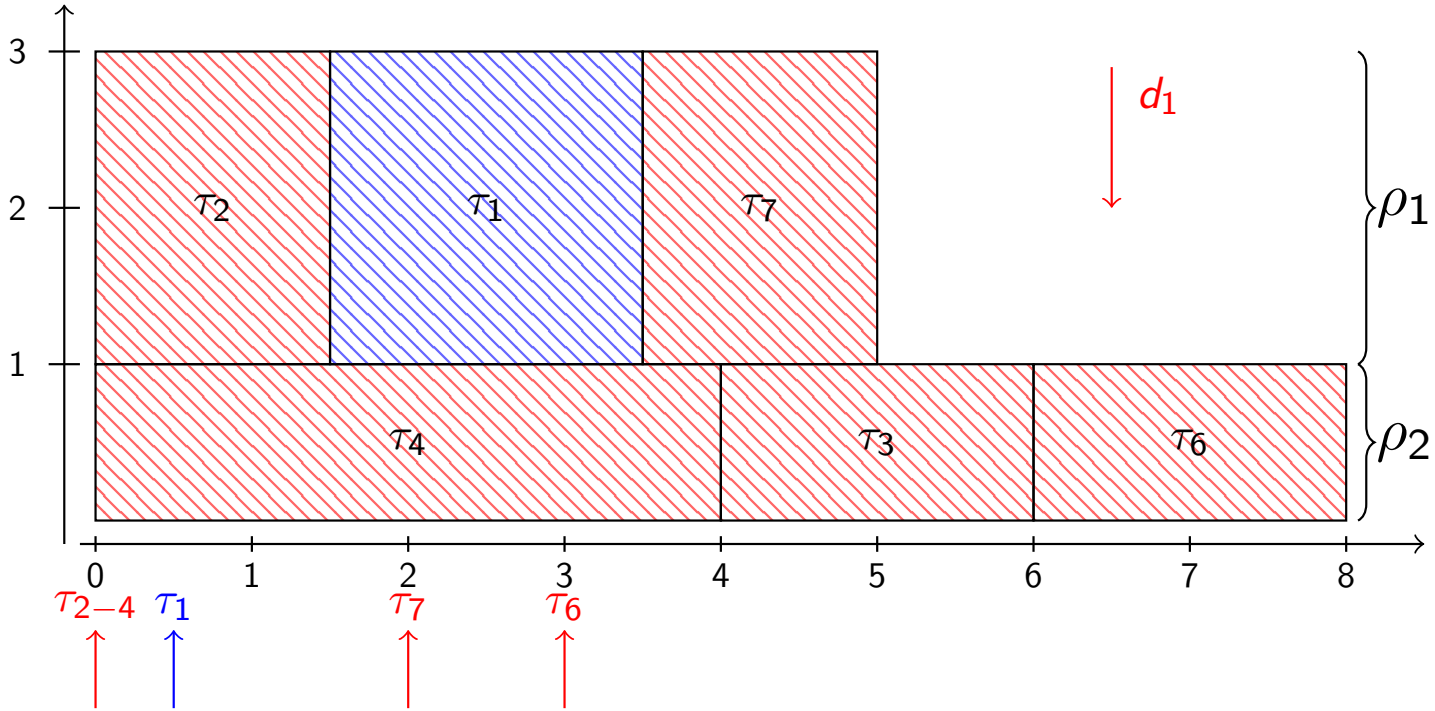


Binqi Sun, Tomasz Kloda, Chu-ge Wu, Marco Caccamo

Partitioned Scheduling and Parallelism Assignment for Real-Time DNN Inference Tasks on Multi-TPU

DAC 2024

Parallelism assignment



Binqi Sun, Tomasz Kloda, Chu-ge Wu, Marco Caccamo

Partitioned Scheduling and Parallelism Assignment for Real-Time DNN Inference Tasks on Multi-TPU

DAC 2024

Parallelism assignment

Heuristic

Input: M processors, a set of tasks

Create M partitions of size 1

for each task τ_i:

sort partitions in ascending order of parallelism efficiency
($WCET \cdot m$=10·2=20 is more efficient than $WCET \cdot m$=9·3=27)
for each partition ρ_j:

if τ_i is schedulable on ρ_j:

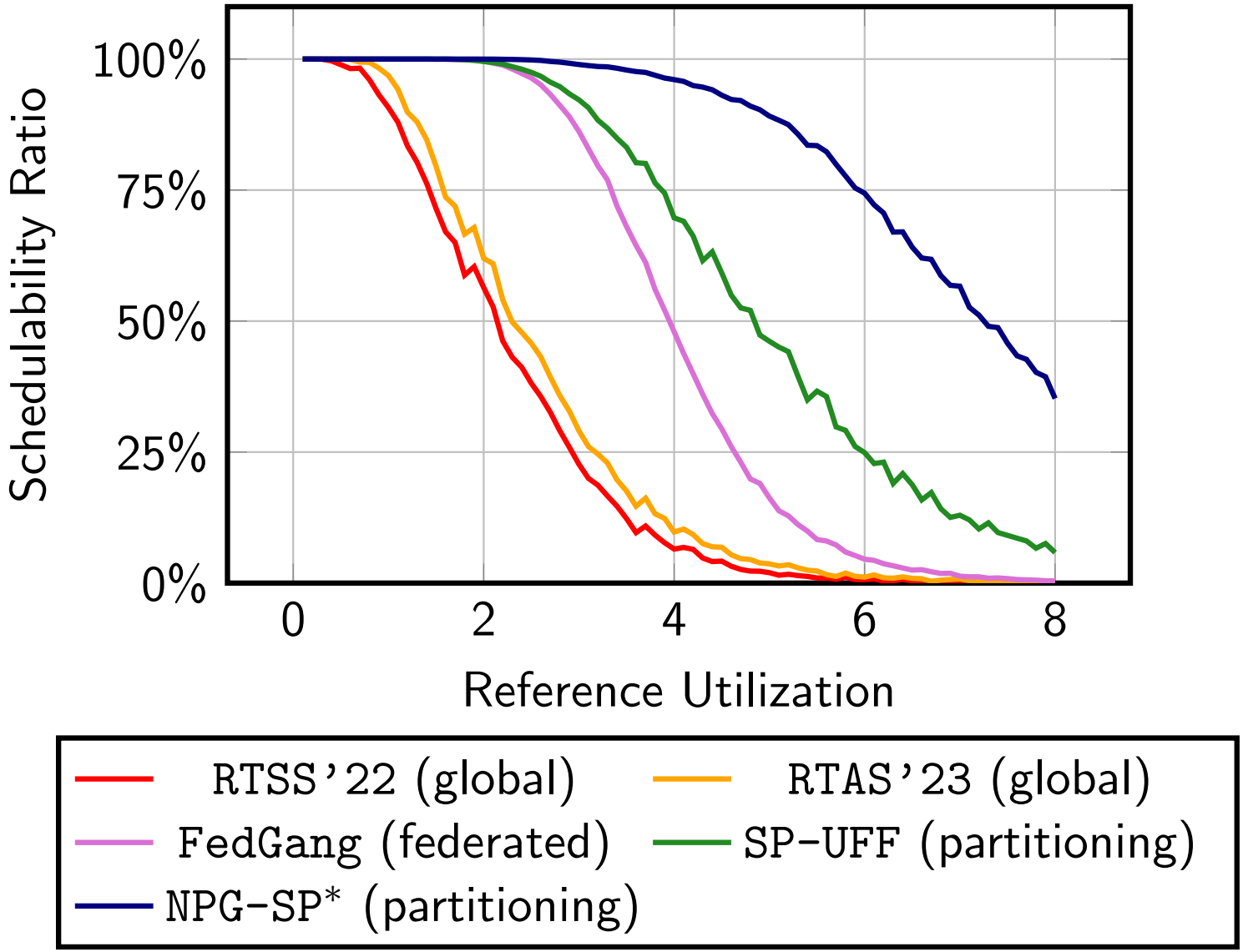
add τ_i to ρ_j and move to the next task

if τ_i is not schedulable on any partition do local search
if τ_i is still not schedulable:

merge two less efficient partitions and retry

Binqi Sun, Tomasz Kloda, Chu-ge Wu, Marco Caccamo
Partitioned Scheduling and Parallelism Assignment for Real-Time DNN Inference Tasks on Multi-TPU
DAC 2024

Parallelism assignment



Conclusions and future works

Conclusions and ongoing works

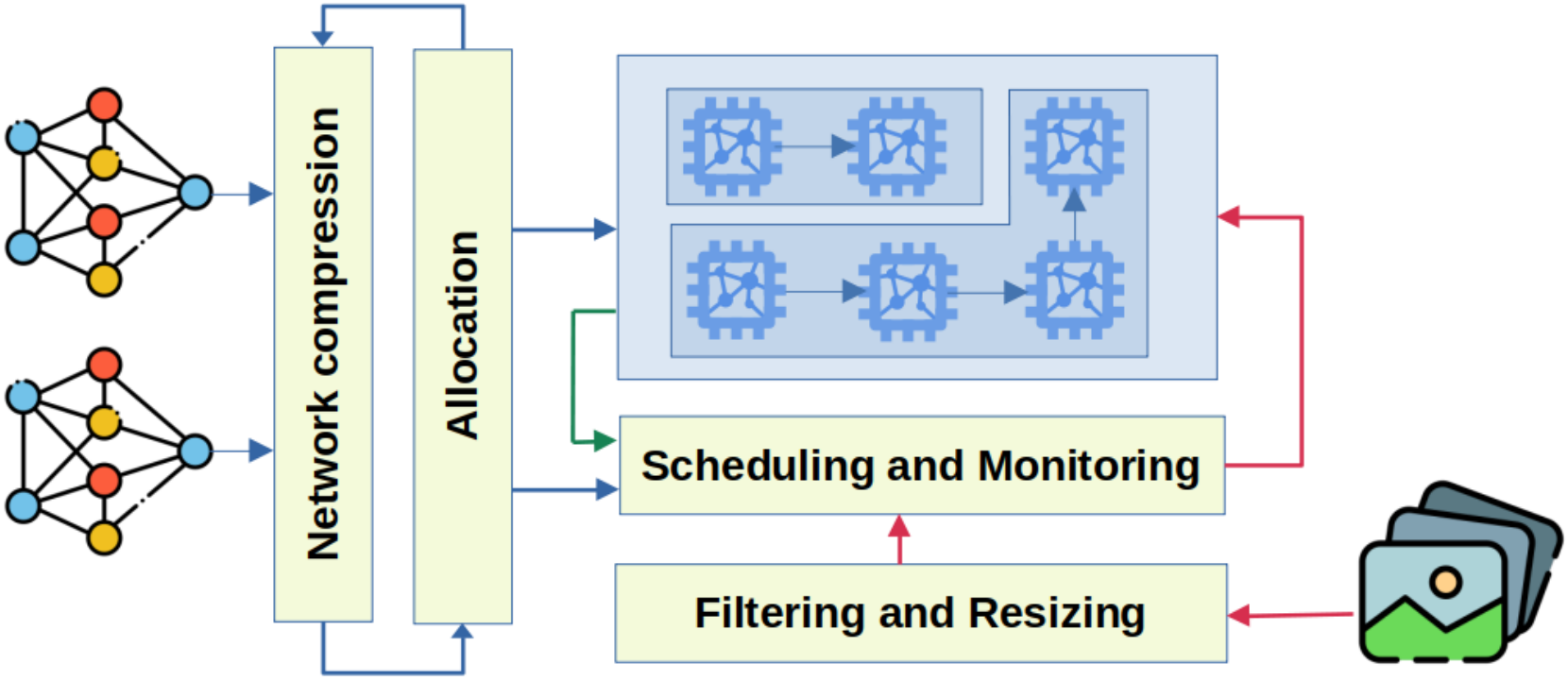
Conclusions

Partitioned approaches avoid scheduling anomalies, leverage well-known single-processor techniques and achieve high processor utilization.

Ongoing works

- CPU-TPU coordinated co-scheduling (*e.g.*, self-suspension)
- limited preemption and model preloading
- different setups (*e.g.*, parallel)
- dynamic neural networks (*e.g.*, early exit)
- model partitioner for multi-TPU

Future works



Thank you for your attention