DASYA

DAPHNE

RAD

IT UNIVERSITY OF COPENHAGEN

www.dasya.dk
@dasyaITU

https://daphne-eu.github.io/

https://rad.itu.dk

www.itu.dk

Sustainable Use of Hardware for Deep Learning – A case for Collocation

Pınar Tözün

Associate Professor

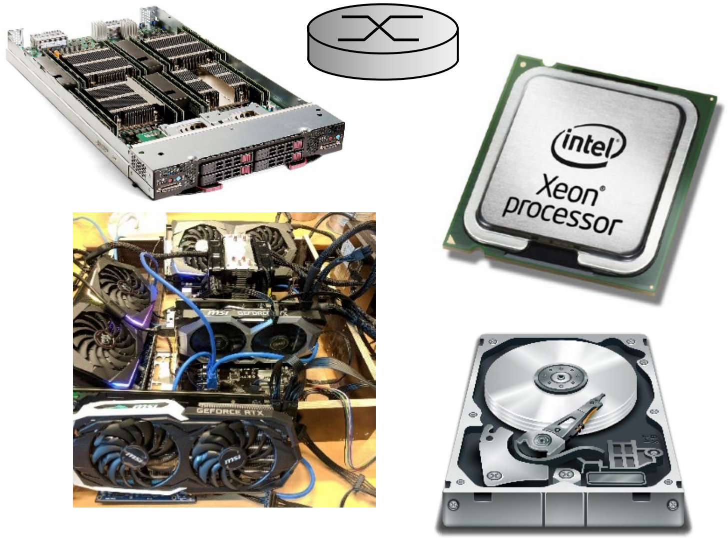
Computer Science Department

Data-Intensive Systems & Applications (DASYA) Lab

pito@itu.dk, www.pinartozun.com, @pinartozun

27/09/2022

hardware-conscious data(-intensive) systems



PyTorch
TensorFlow
MySQL
APACHE Spark

are data systems that utilize modern hardware well.

hardware:
processors (e.g., CPU, GPU),
storage (e.g., hard disk, SSD),
network (e.g., router)
…

data systems:
machine learning frameworks,
database management systems,
big data systems
…

why is it important to utilize hardware well for sustainability?
what does it mean to utilize hardware well?

agenda

- why is it important to utilize hardware well?
- are we utilizing hardware well?
- can we utilize hardware better?

hardware focus:
co-processors

software focus:
deep learning frameworks

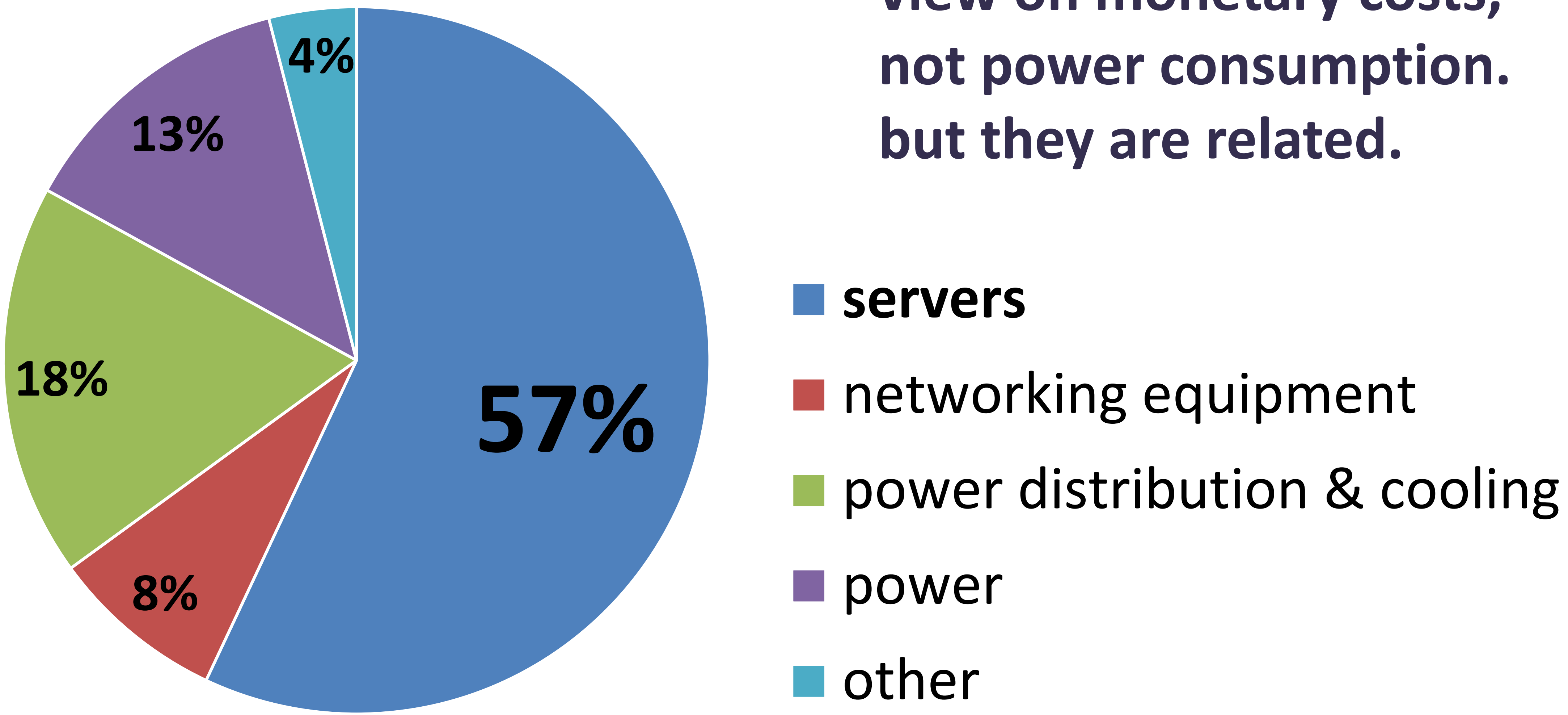
PyTorch

TensorFlow

monthly costs ($$) of a data center

source: James Hamilton, 2010

**view on monetary costs,
not power consumption.
but they are related.**



bad utilization of *servers* is a waste of both power & money

unsustainable growth of deep learning

2012 → **present**

- powerful hardware
- larger datasets
- deep learning frameworks

APIs (python, R, …)

deep learning software

CPU | GPU | TPU | FPGA | …

commodity hardware (CPU, GPU)

PyTorch

TensorFlow

300000x increase in ***computational need*** for deep learning models.

- computational efficiency is ignored

➔ **main performance metric = *accuracy***

- high computation (carbon) footprint

➔ **… with low transparency**

- throw new & expensive hardware at the problem?

➔ **no, there is no free lunch**

sources: https://openai.com/blog/ai-and-compute/, Strubell et al. ACL 2019, Schwartz et al. GreenAI 2019

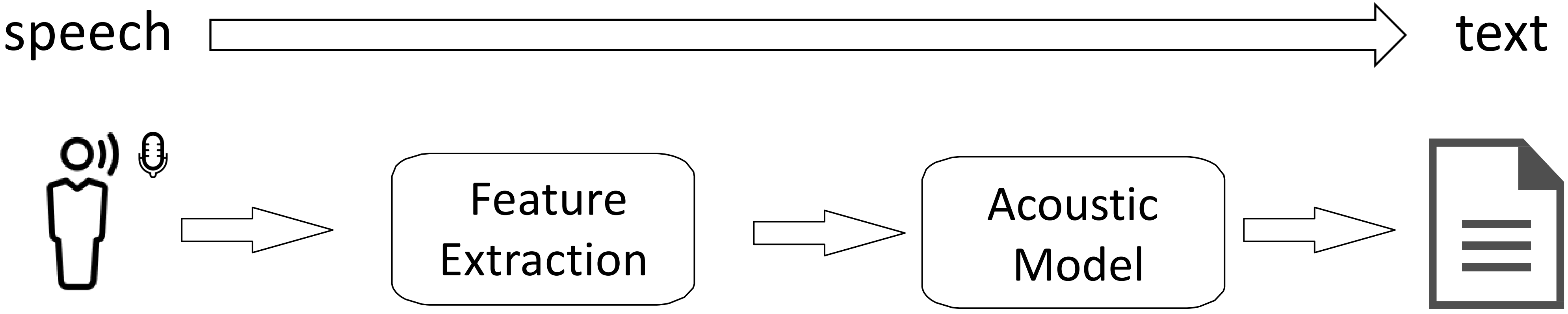
agenda

- why is it important to utilize hardware well?
- are we utilizing hardware well?
  - use case: speech recognition on CPU-GPU co-processors
- can we utilize hardware better?

sebastian benjamin wrede

sebastian baunsgaard

speech recognition

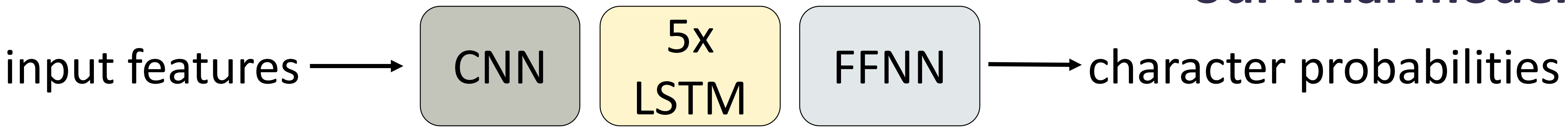


- human-computer & human-human interactions
- hospitals, call-centers, virtual assistants, etc.

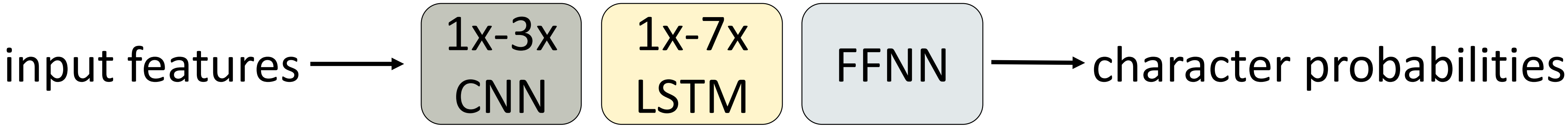
state-of-the-art *acoustic models* are based on neural networks in recent years → natural fit for GPUs

acoustic model

our final model



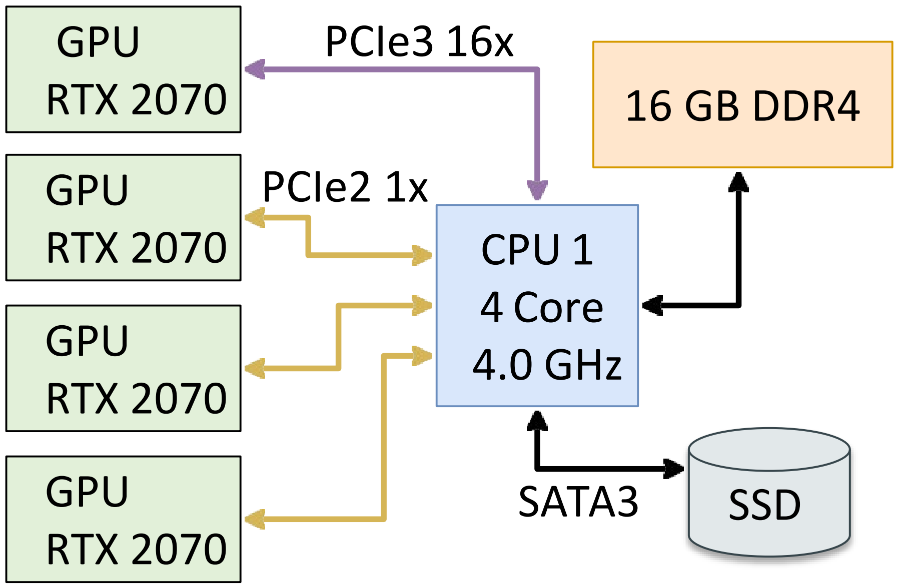
- inspired by Baidu Research, Deep Speech 2, ICML 2016
- basis for MLPerf’s speech recognition benchmark as well

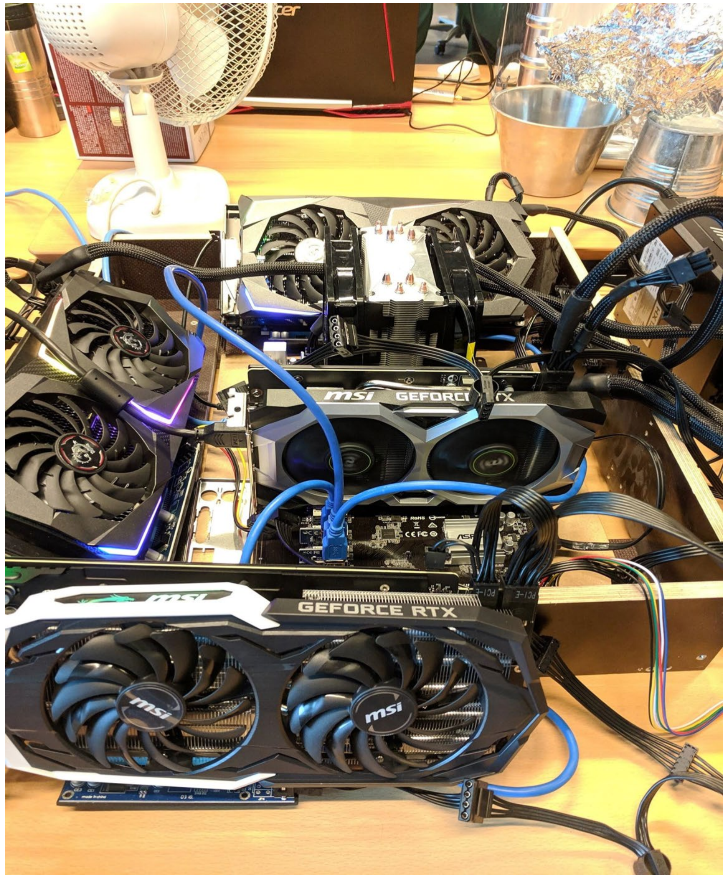


deep speech 2

process of determining the right set of layers is heavily based on trial-&-error

first, let’s introduce ***rebelrig*** = co-processor built in-house

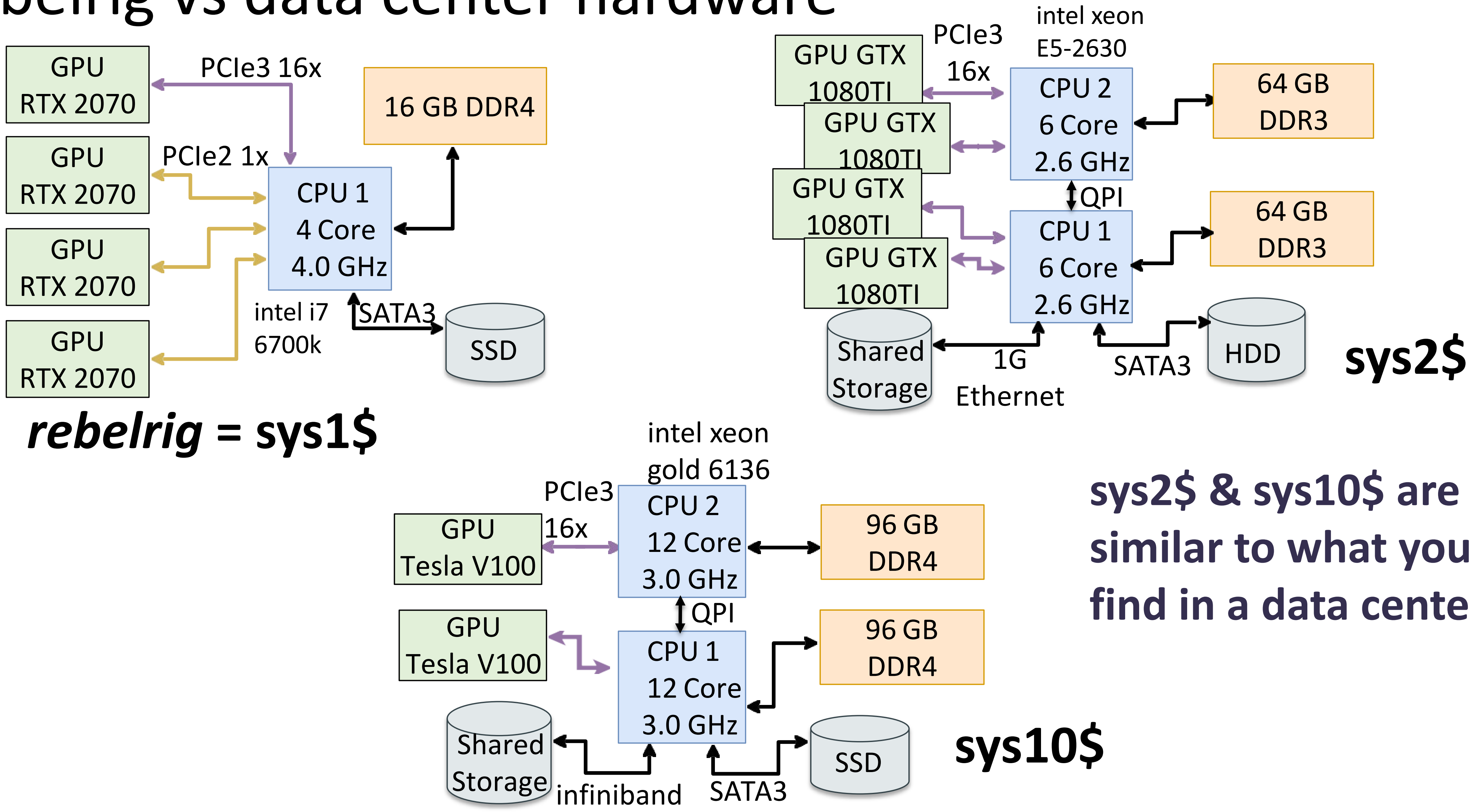




- motherboard = repurposed cheap crypto-mining rig
- not something you find in a data center

in DASYA lab, we try our best to repurpose old hardware rather than thrashing them

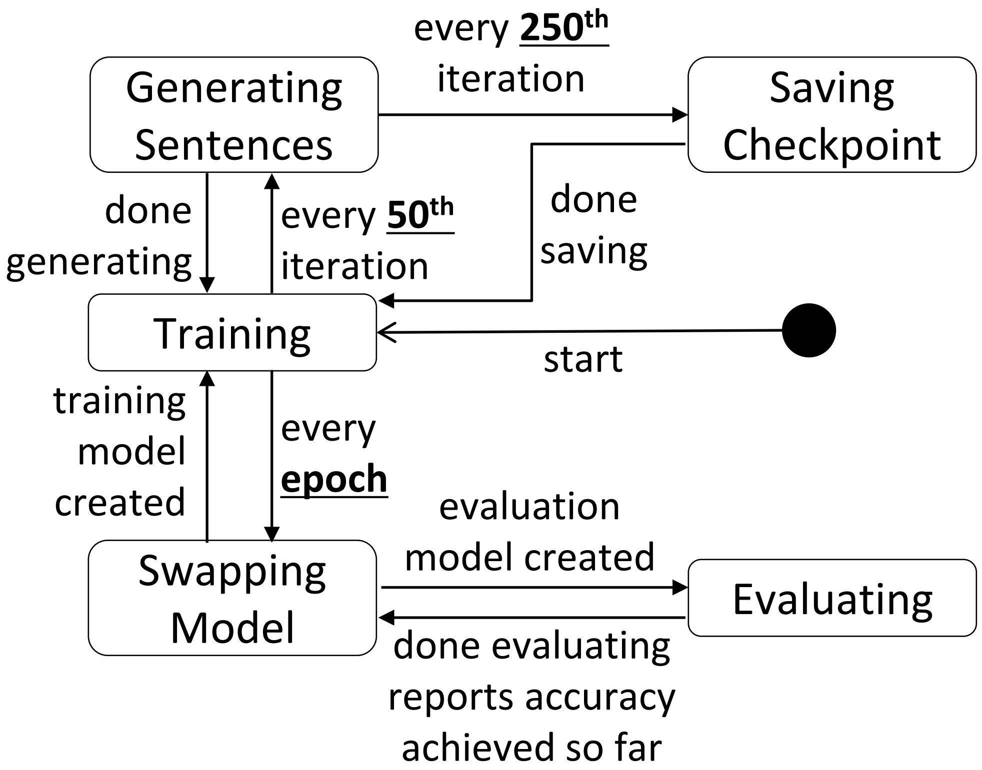
rebelrig vs data center hardware



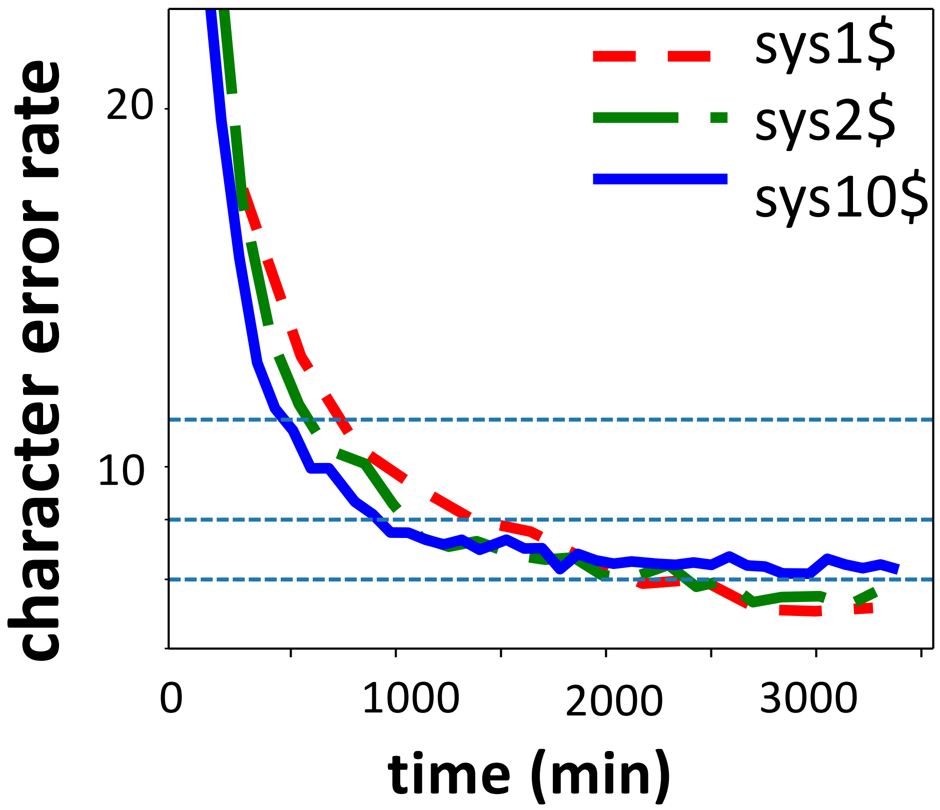
sys2$ & sys10$ are similar to what you find in a data center

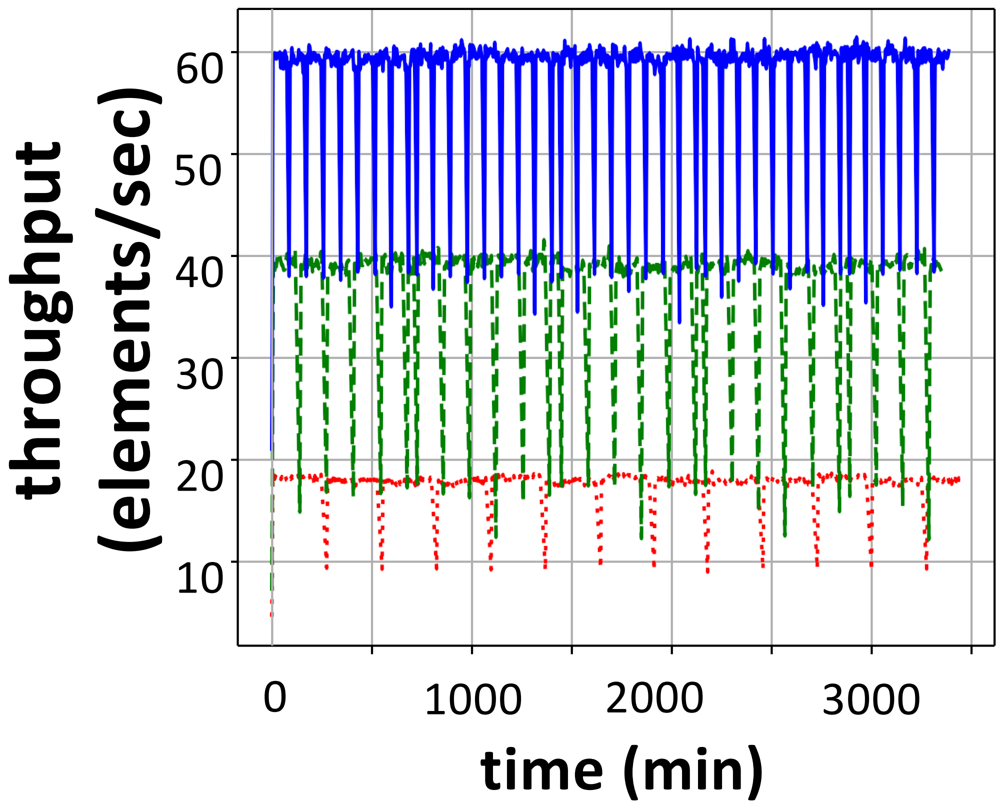
experimental setup

- acoustic model implemented using TensorFlow 1.14
- training over three hardware platforms
- dataset : LibriSpeech
  - audiobooks
  - ~1000 hours of speech (both clean & noisy)



price/performance results





no huge difference in accuracy across platforms

high throughput != faster time-to-accuracy

lessons learned

- very powerful co-processors more and more widely available for machine learning / deep learning
- but takes a lot to exploit, no free lunch as usual
- need to invest further in improving machine learning libraries and hardware resource managers
- on the other hand, low-budget platforms may be good enough for your needs

same old challenge for data-intensive systems, different workload & hardware

agenda

- why is it important to utilize hardware well?
- are we utilizing hardware well?
- can we utilize hardware better?
  - our direction: workload collocation

workload collocation

multiple workloads sharing hardware resources

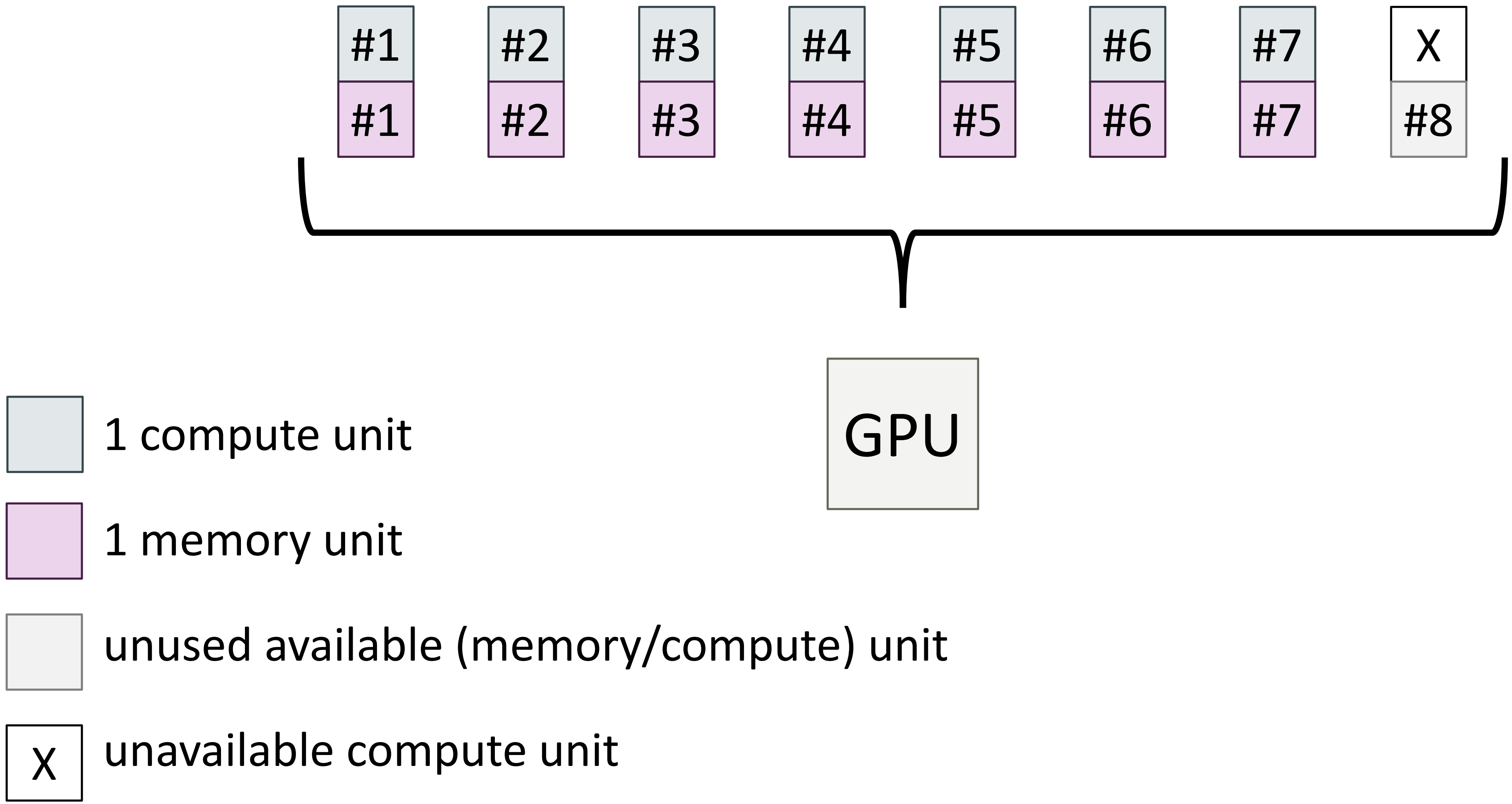
benefits when a single workload cannot utilize available resources

main high-level challenge = interference across workloads

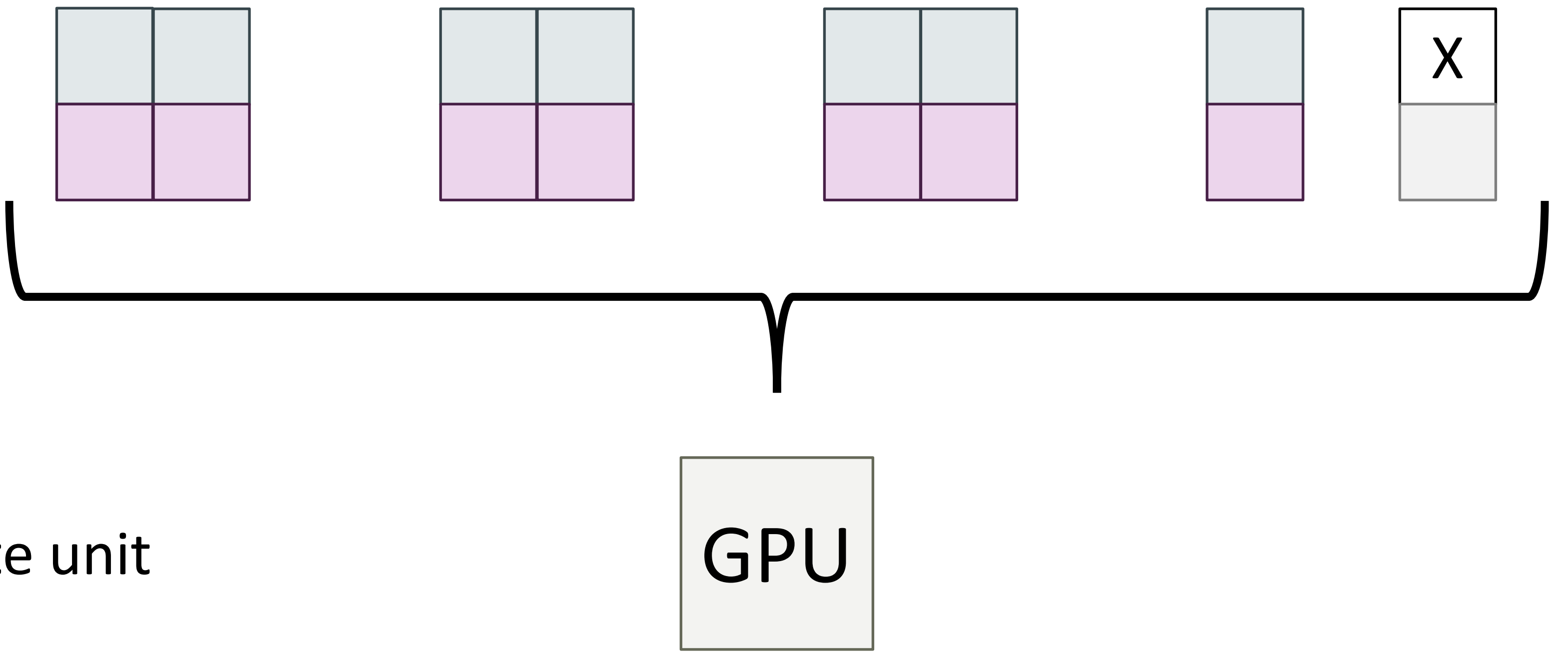
workload collocation on (NVIDIA) GPUs

- ***vanilla co-location***
  - kernels of different workloads are time-multiplexed (not concurrent)
- ***virtualization***
  - practical, but also based on time-sharing
- ***multi-process service (MPS)***
  - GPU resources are split (auto-magically) across collocated workloads
  - kernels of different applications can run simultaneously
  - allowed for one user (for safety reasons)
- ***multi-instance GPU (MIG)***
  - hardware support for resource split, introduced with NVIDIA A100
  - can do all of the above in a MIG partition

multi-instance GPU



multi-instance GPU



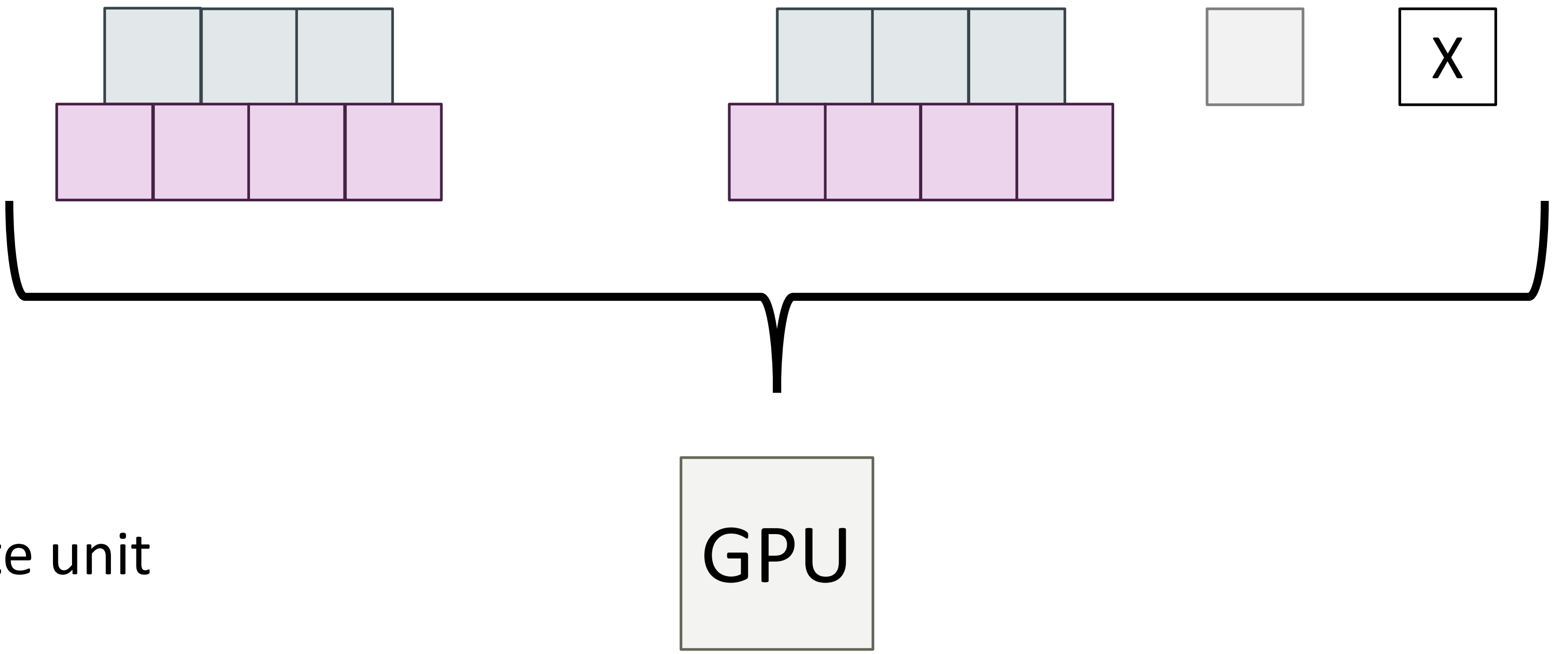
1 compute unit

1 memory unit

unused available (memory/compute) unit

X unavailable compute unit

multi-instance GPU



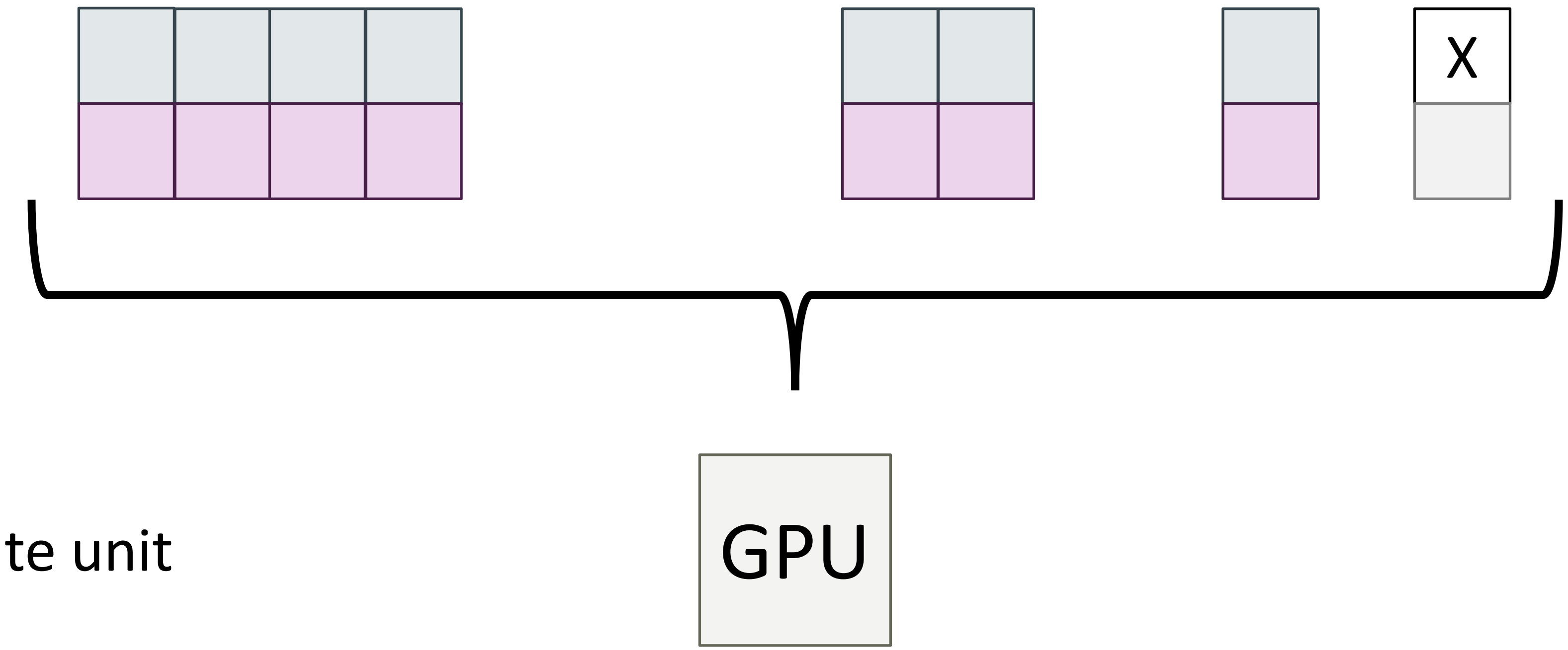
1 compute unit

1 memory unit

unused available (memory/compute) unit

X unavailable compute unit

multi-instance GPU



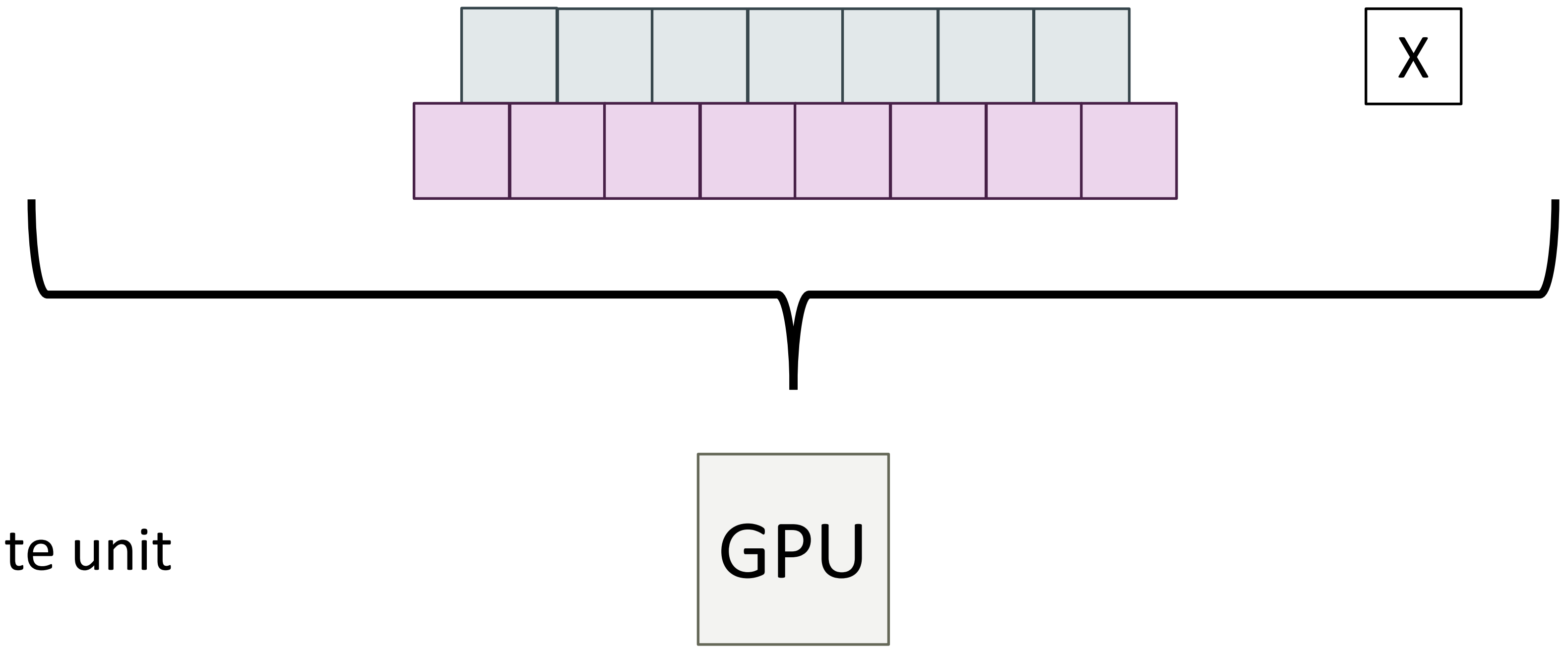
1 compute unit

1 memory unit

unused available (memory/compute) unit

X unavailable compute unit

multi-instance GPU



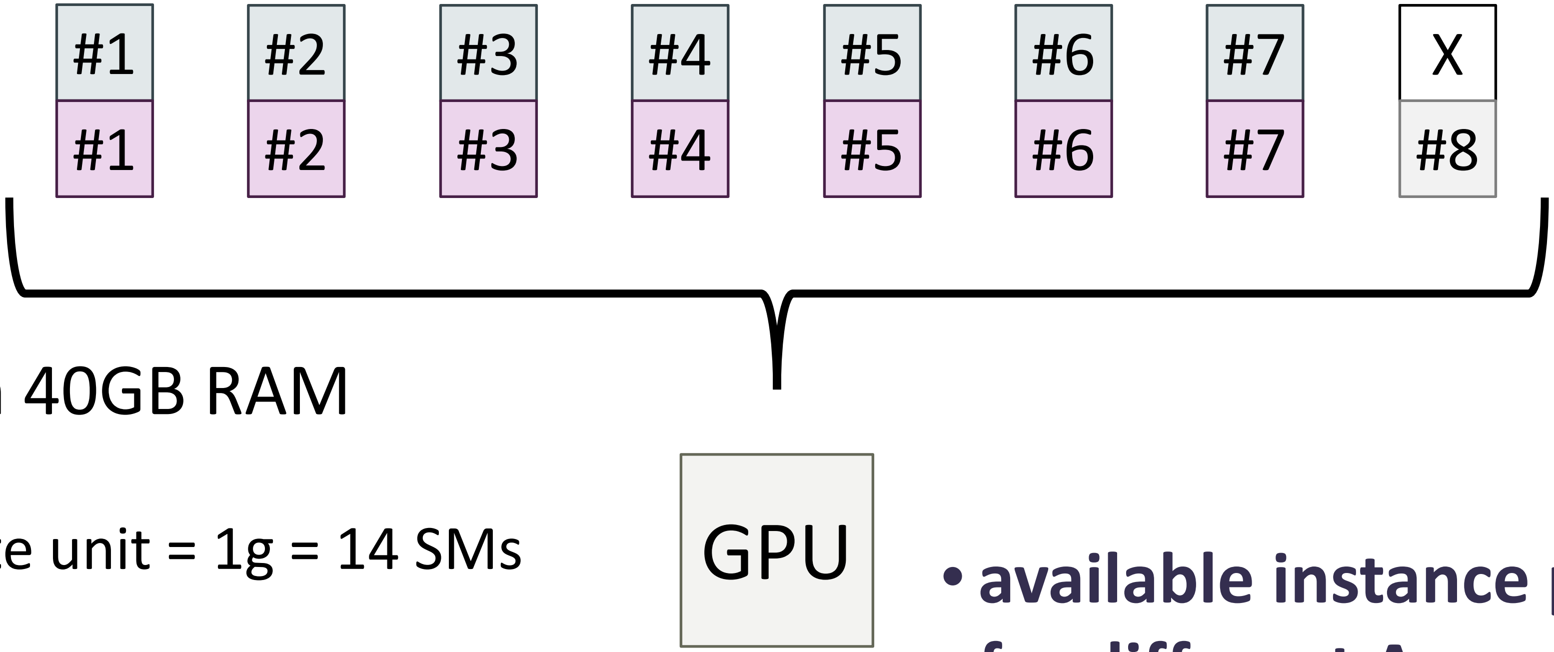
1 compute unit

1 memory unit

unused available (memory/compute) unit

X unavailable compute unit

multi-instance GPU



on A100 with 40GB RAM

1 compute unit = 1g = 14 SMs

1 memory unit = 5GB

unused available (memory/compute) unit

X unavailable compute unit = less than 14 SMs (SM = streaming multiprocessor)

- **available instance profiles differ for different Ampere GPUs**
- **doesn't allow distributed training**

performance impact of MIG-based co-location

NVIDIA DGX Station A100

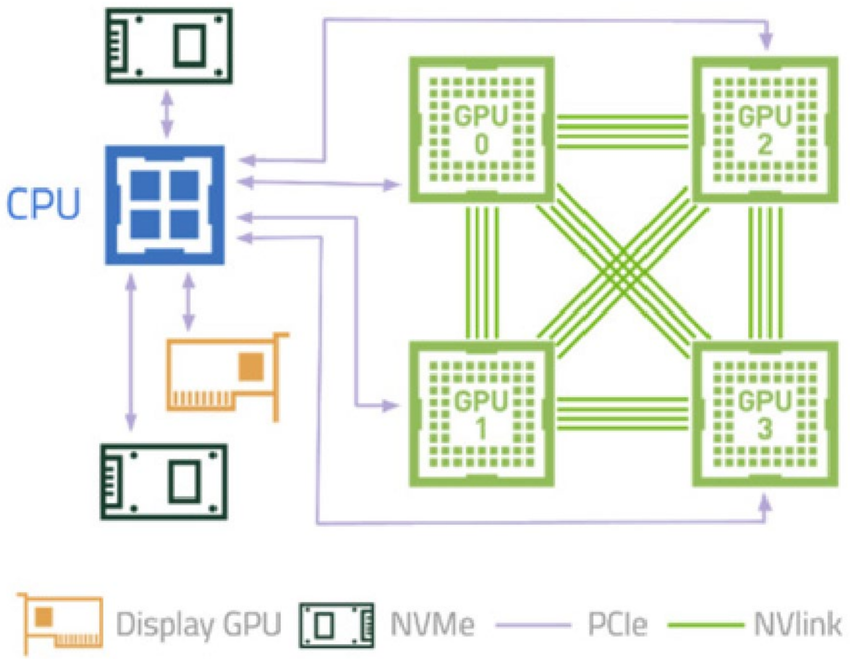


figure source

CPU = AMD 7742 – 512 GB RAM
64 physical cores
GPU = NVIDIA A100 – 40 GB RAM
allows ***multi-instance GPU (MIG)***

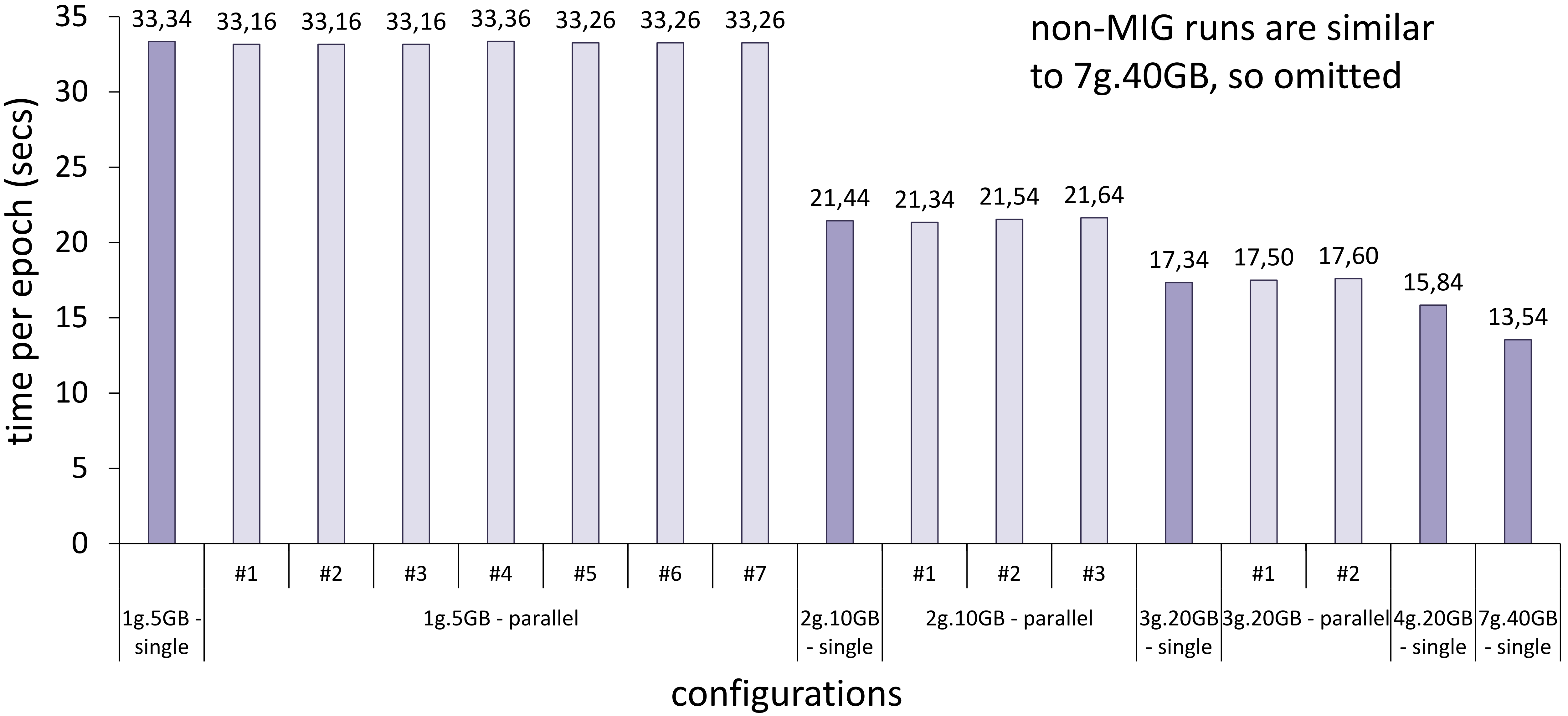
TensorFlow 2.7

workloads	model	dataset
small	ResNet26	CIFAR-10
medium	ResNet50	downsampled ImageNet*
large	ResNet152	ImageNet (2012)

batch size = 32 for all
runs on single GPU
- 25 epochs for small
- 5 epochs for medium & large

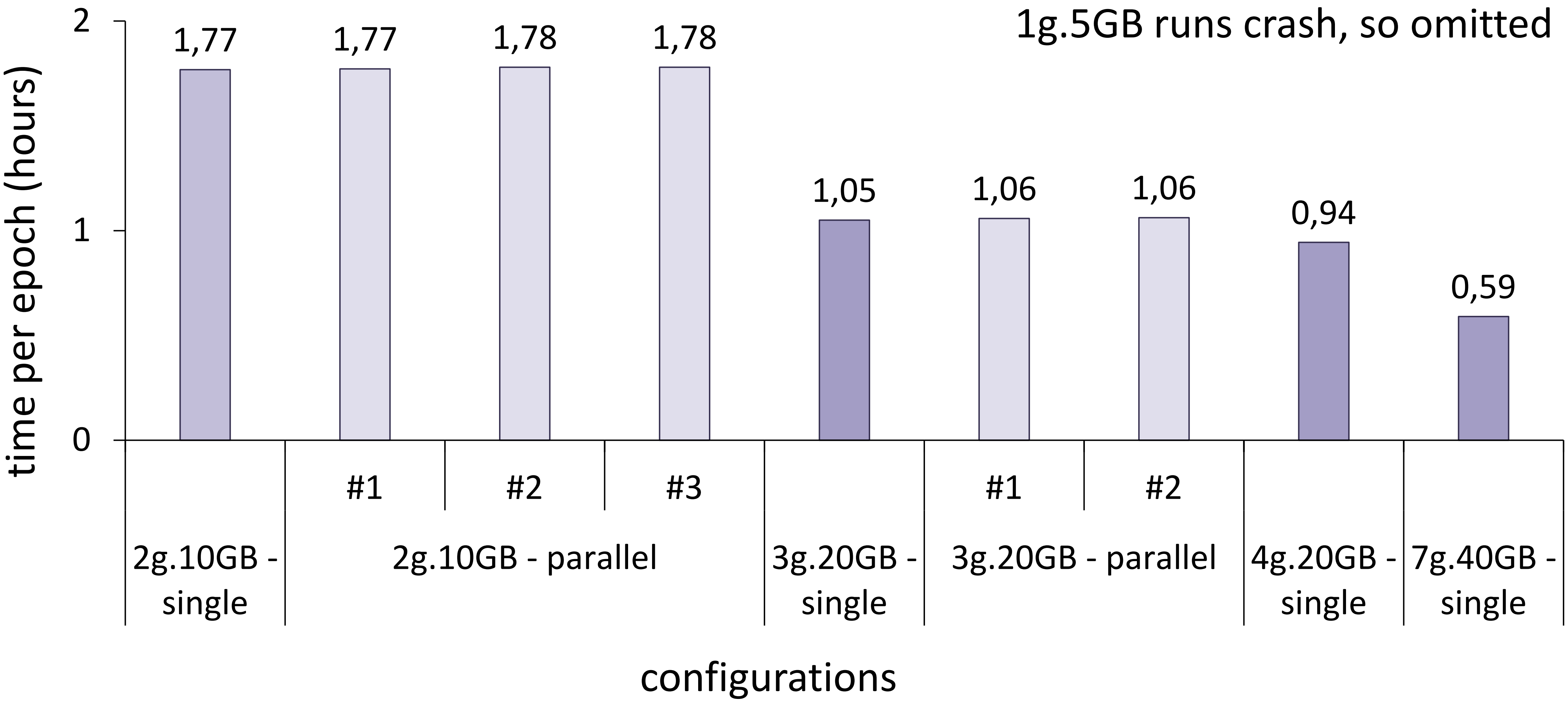
MSc thesis work of
Stilyan Petrov Paleykov
& Anders Friis Kaas

time per epoch – small case



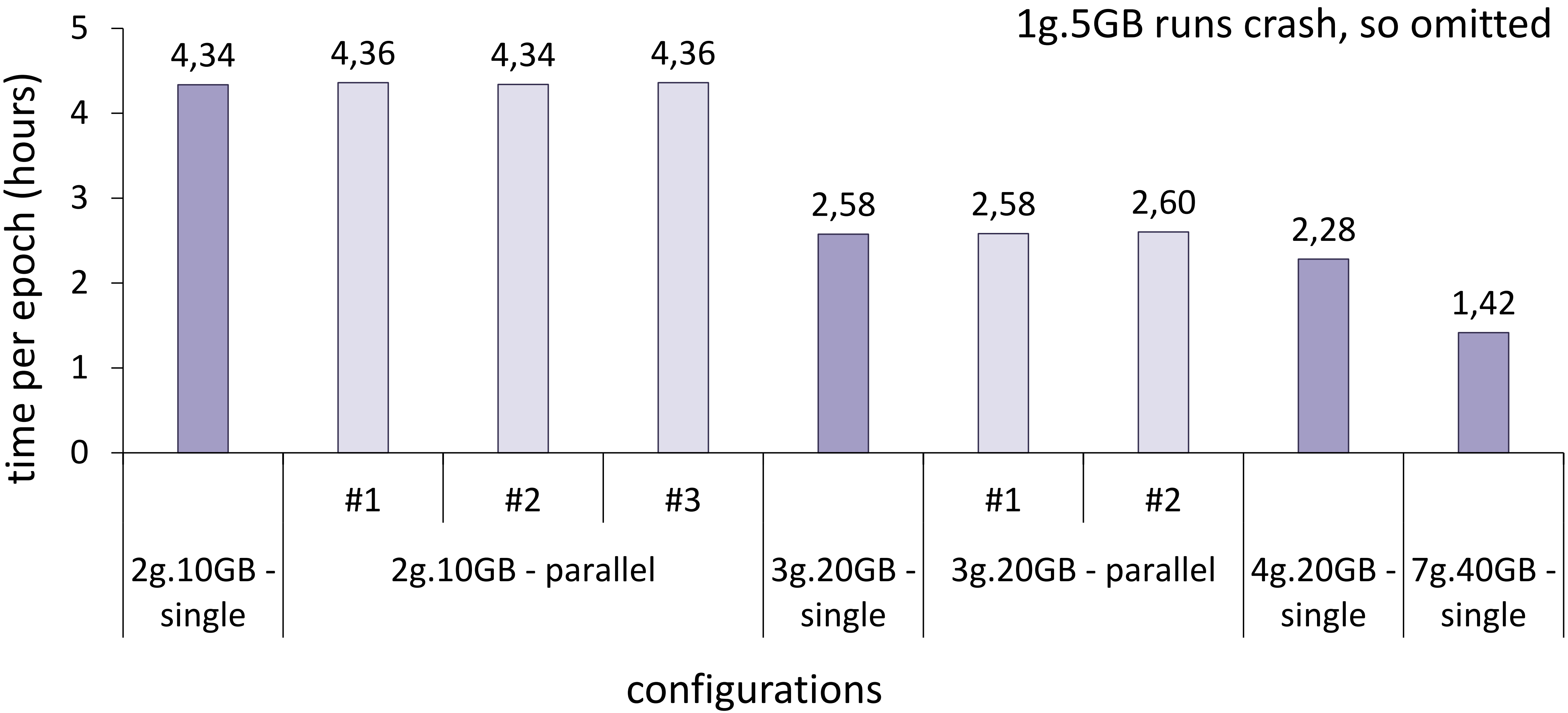
opportunity to collocate training runs with slight latency increase

time per epoch – medium case



1g.5gb case isn't feasible anymore due to insufficient memory, not much gain from collocation

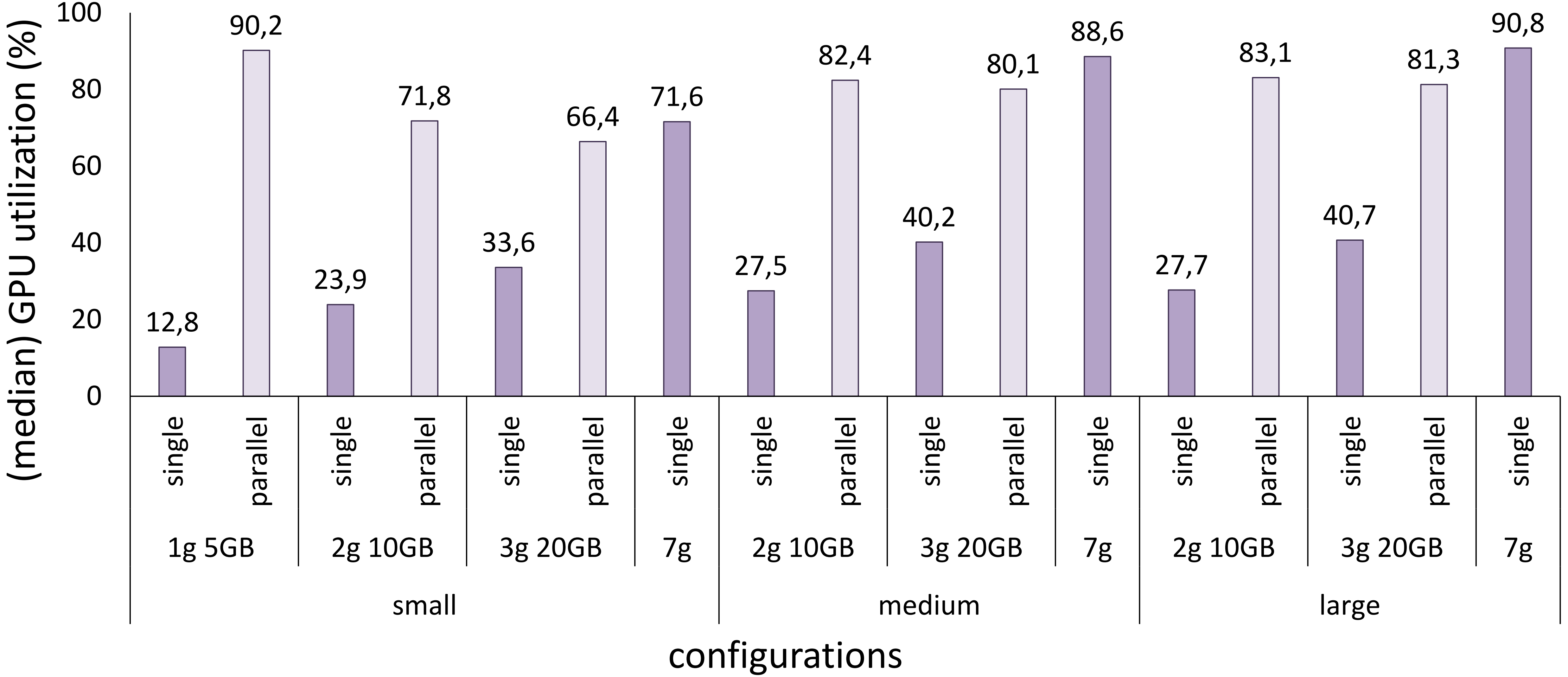
time per epoch – large case



similar to medium case in terms of co-location

overall, parallel runs don’t interfere as long as there is enough memory

GPU utilization



fine-grained parallel runs increase utilization for *small case*

***medium & large* cases utilize the whole GPU well without parallel runs**

challenges & opportunities

- hardware is a resource, must use it well
- many data-intensive systems (e.g., deep learning frameworks) do not use modern hardware well out-of-the-box

opportunities

- GPUs that allow finer-grained scheduling & space management
- diversity of applications, hardware, & end-users

➔ **creates opportunity for effective resource sharing on GPUs**

challenges

- representative workloads
- experimental duration
- profiling & collocation granularity

thank you!