DASYA
Data-Intensive Systems and Applications
dasya.itu.dk

RAD
rad.itu.dk

INDEPENDENT RESEARCH FUND DENMARK
dff.dk

IT UNIVERSITY OF COPENHAGEN
itu.dk

CARMA: Collocation-Aware Resource Manager with GPU Memory Estimator

Ehsan Yousefzadeh-Asl-Miandoab

ehyo@itu.dk

24th September

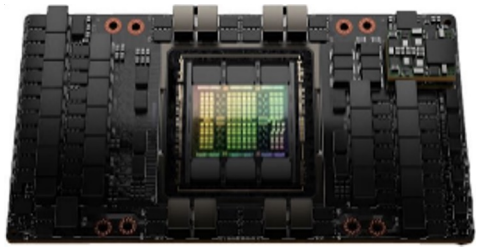
SUITS, Denmark, 2025

GPU Underutilization Challenge

An analysis of 100,000 jobs run by 100s of users for ~2 months on a real-world cluster shows ~52% GPU utilization on average ✷

Hardware problem

1. Compute/memory requirements of models don't always match with the GPUs
   - e.g., transfer learning, small models
2. No fine-grained sharing mechanism and virtual memory on GPUs



Mem: 141GB

BASIC-L: ~2.44billion params

MobileNetV2: ~3.4M params

Software problem

1. Exclusive GPU assignments
2. Mostly black-box view of tasks

kubernetes

hadoop YARN

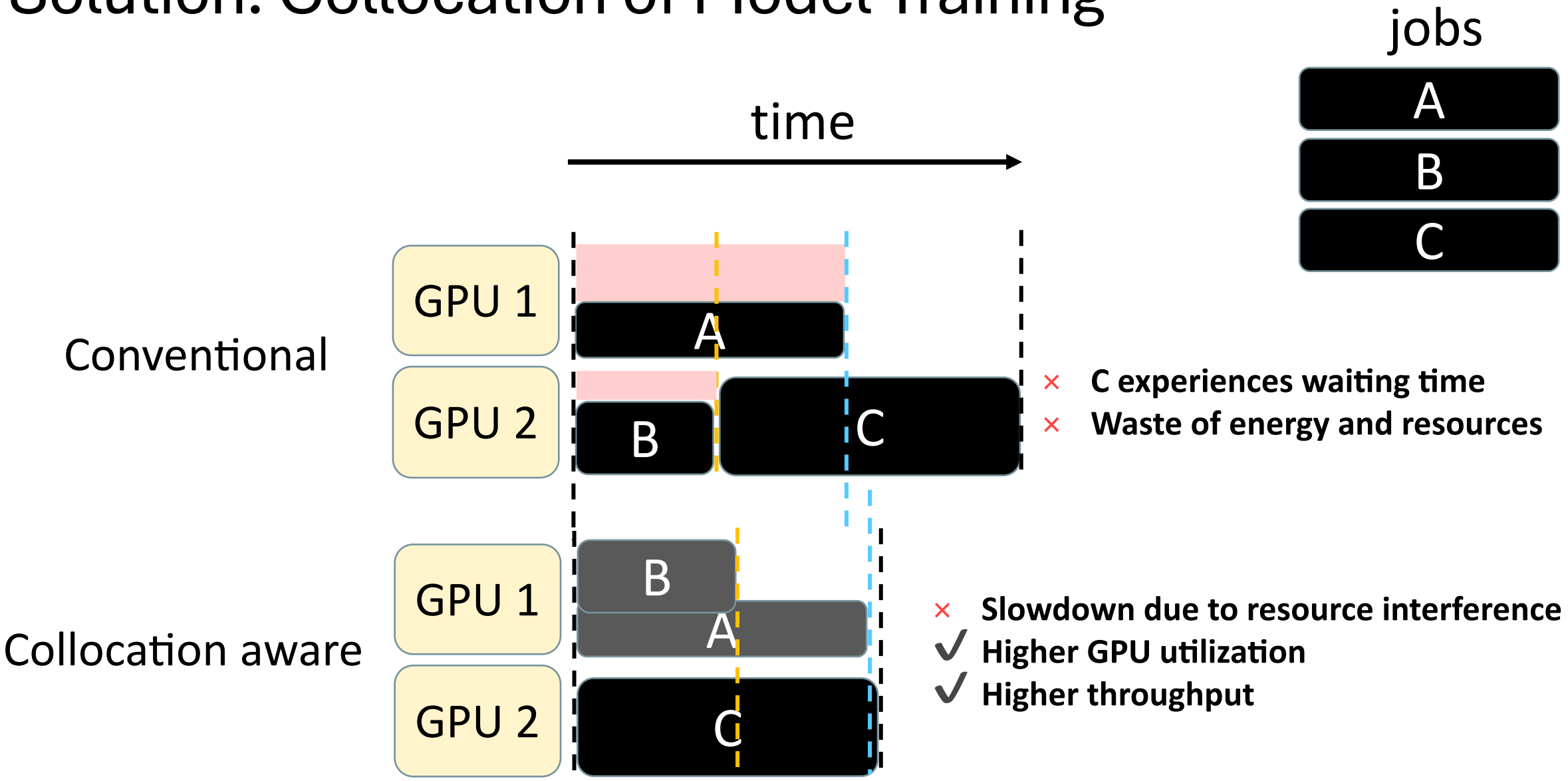
slurm workload manager

✷ Jeon, Myeongjae, et al. "**Analysis of Large-Scale Multi-Tenant GPU Clusters for DNN Training Workloads.**" USENIX Annual Technical Conference. 2019.
✷ Yanjie Gao et al. **"An Empirical Study on Low GPU Utilization of Deep Learning Jobs."** In: Proceedings of the IEEE/ACM 46th International Conference on Software Engineering. 2024.

Gefion AI Supercomputer

1528 NVIDIA H100
Tensor Core GPUs

Solution: Collocation of Model Training



Statement

Improving GPU Underutilization challenge and increasing GPU energy efficiency through task collocation.

Overview

Profiling & Monitoring (EuroMLSys'23) — 1

Collocation Analysis (EuroMLSys'24) — 2

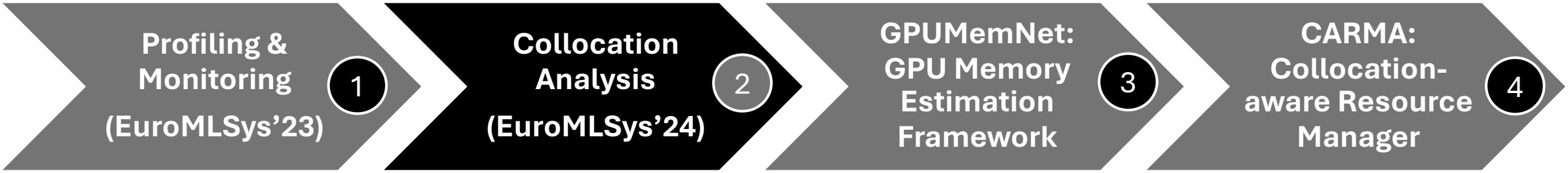
GPUMemNet: GPU Memory Estimation Framework — 3

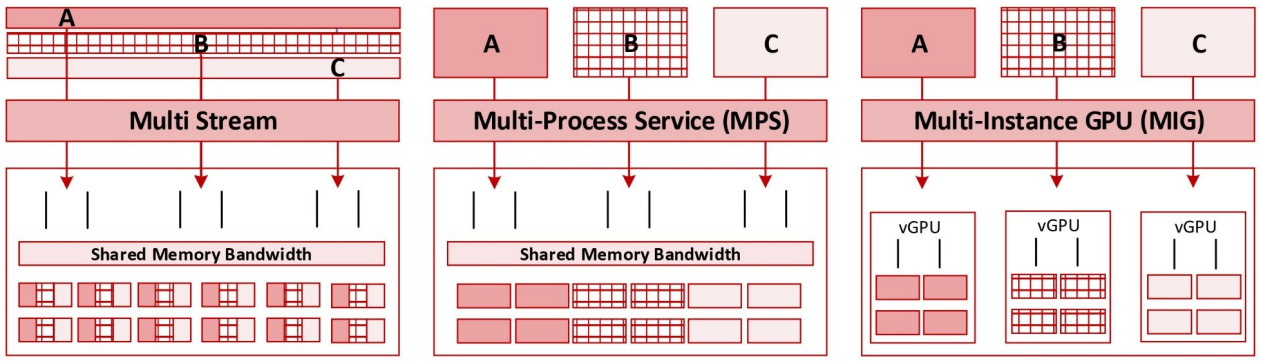
CARMA: Collocation-aware Resource Manager — 4

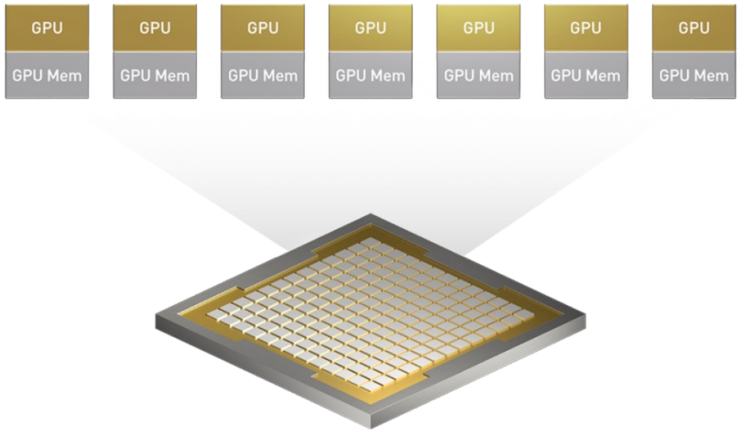
```
sh-4.2$ nvidia-smi
Wed Dec 11 09:52:10 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 418.87.01    Driver Version: 418.87.01    CUDA Version: 10.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla K80           On   | 00000000:00:1E.0 Off |                    0 |
| N/A   44C    P0    60W / 149W |  11292MiB / 11441MiB |     13%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    0      9389      C   ...naconda3/envs/tensorflow_p36/bin/python 11012MiB |
|    0     21268      C   ...naconda3/envs/tensorflow_p36/bin/python   267MiB |
+-----------------------------------------------------------------------------+
```

Overview



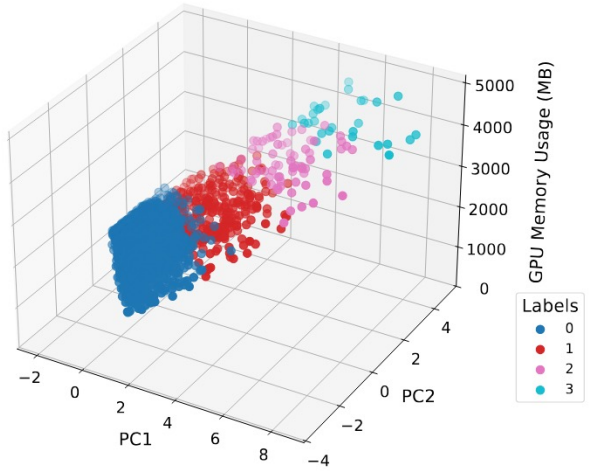




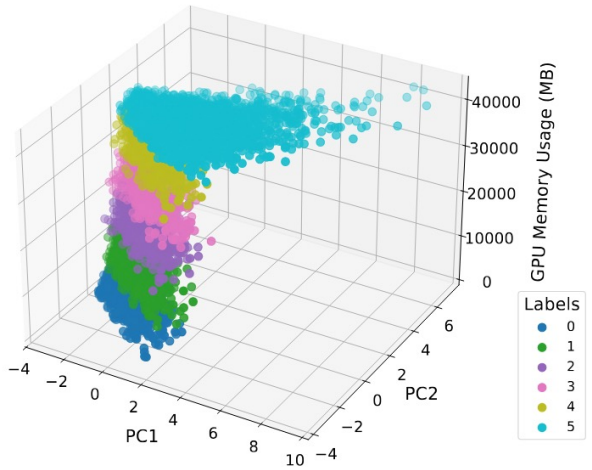
https://www.nvidia.com/en-us/technologies/multi-instance-gpu/

Overview

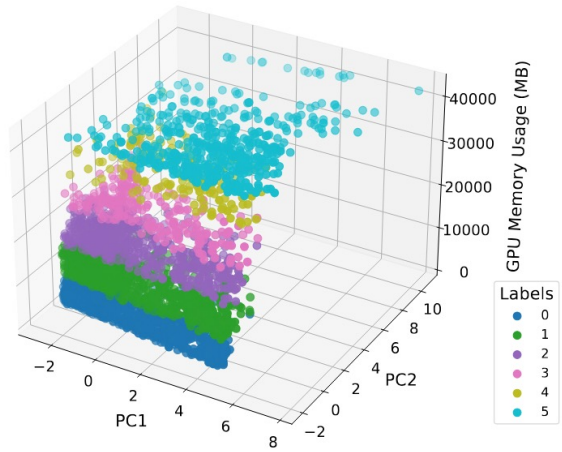
1. **Profiling & Monitoring (EuroMLSys'23)**
2. **Collocation Analysis (EuroMLSys'24)**
3. **GPUMemNet: GPU Memory Estimation Framework**
4. **CARMA: Collocation-aware Resource Manager**



MLPs



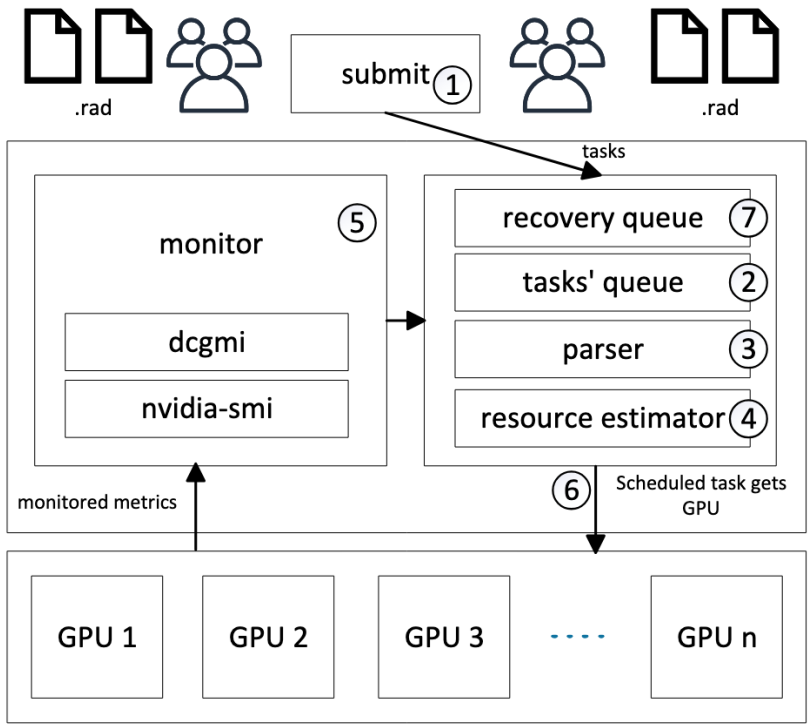
CNNs

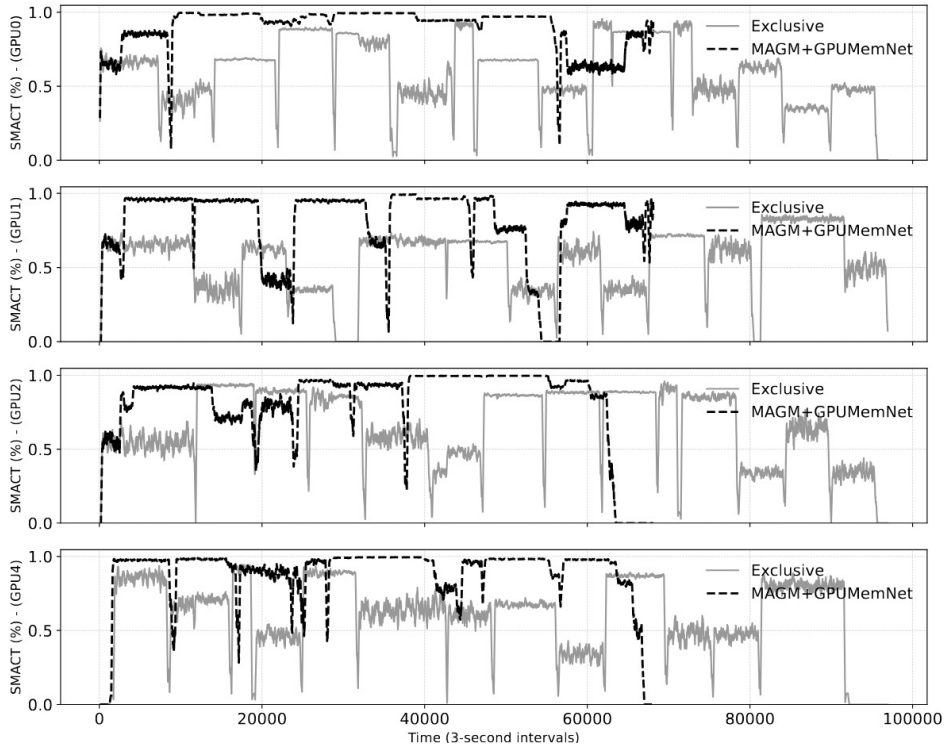


Transformers

Overview

Profiling & Monitoring (EuroMLSys'23) 1 → **Collocation Analysis (EuroMLSys'24)** 2 → **GPUMemNet: GPU Memory Estimation Framework** 3 → **CARMA: Collocation-aware Resource Manager** 4



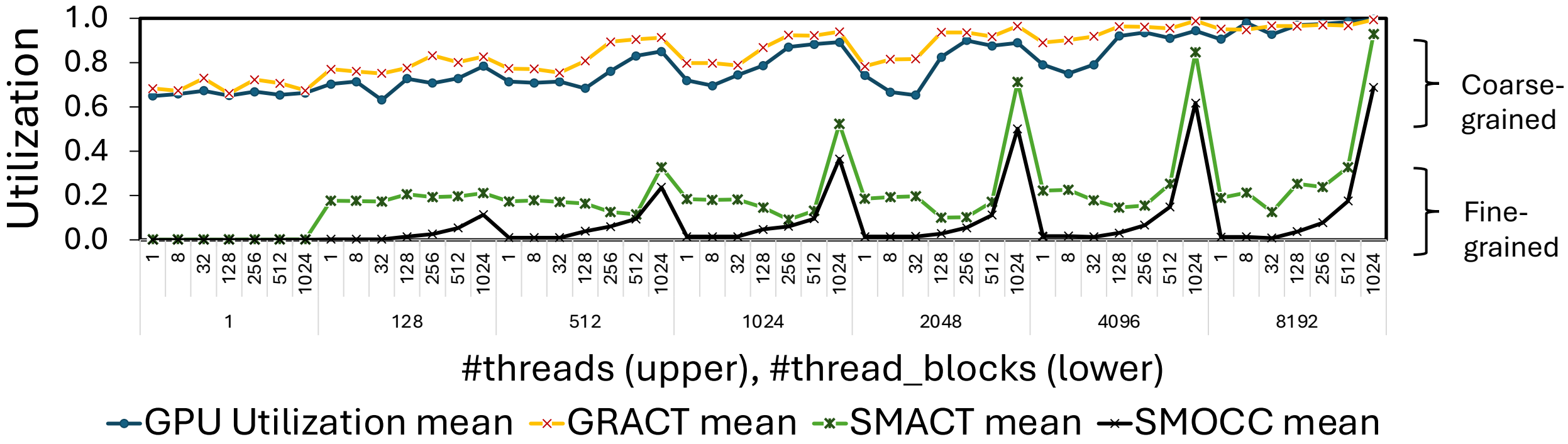


1

Profiling and Monitoring Deep Learning Training Tasks

GPU Utilization

- **GPU Utilization**: % of time one or more kernels were executing on the GPU
- **GRACT**: % of time any portion of the graphics or compute engines were active
- **SMACT:** the fraction of active time on an SM, averaged over all SMs
- **SMOCC**: degree of parallelism / max supported parallelism on SM



Coarser-grained utilization metrics can be misleading.

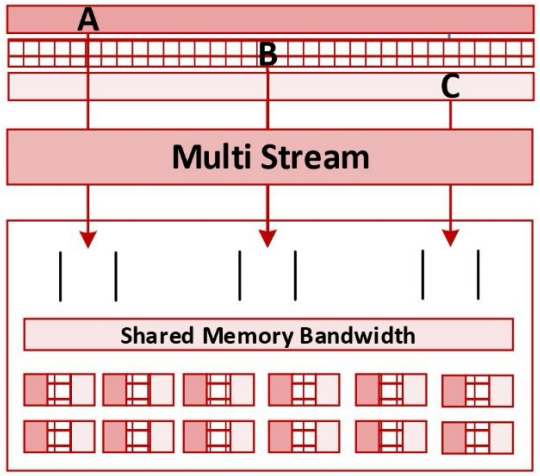
Digest

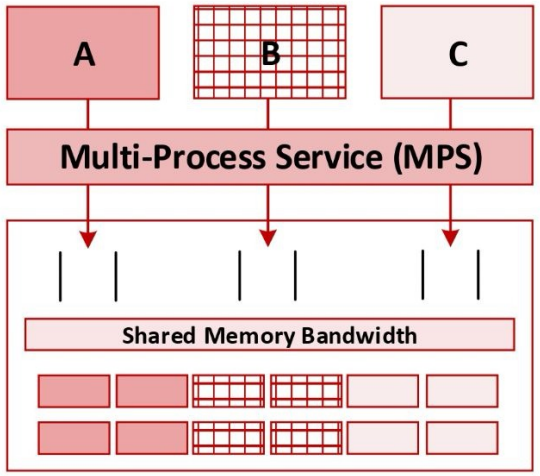
- For model level optimization purposes
  - Use framework specific profilers
- For digging deeper into OS and system
  - Use Nsight Systems
- For kernel-level optimizations
  - Use Nsight Compute
- Profile the needed amount of code for a reasonable range of time
  - Profiling for an iteration might be enough to show the behavior of training a model
- For online decision-making purposes
  - Use monitoring tools with representative fine-grained metrics

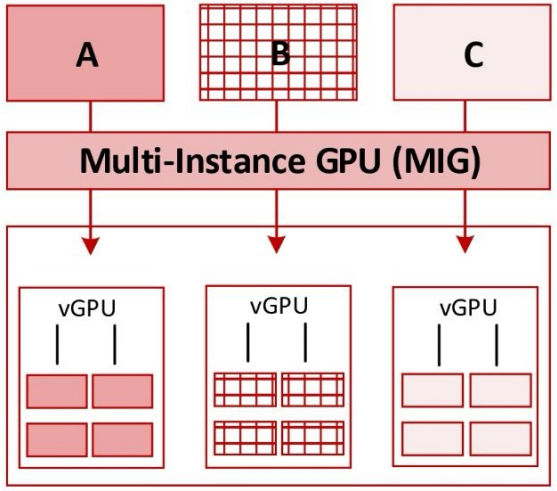
2

An Analysis of Collocation on GPUs for Deep Learning Training

Collocation on NVIDIA GPUs

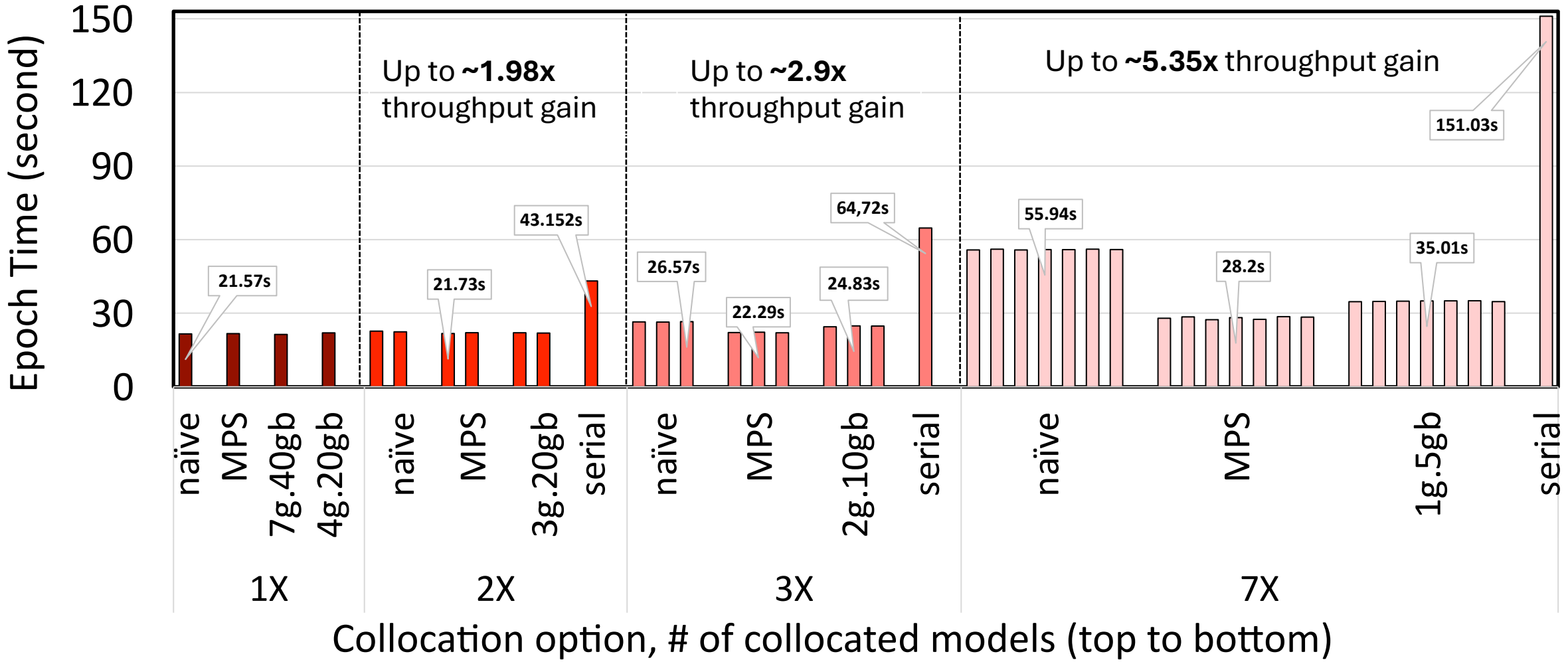






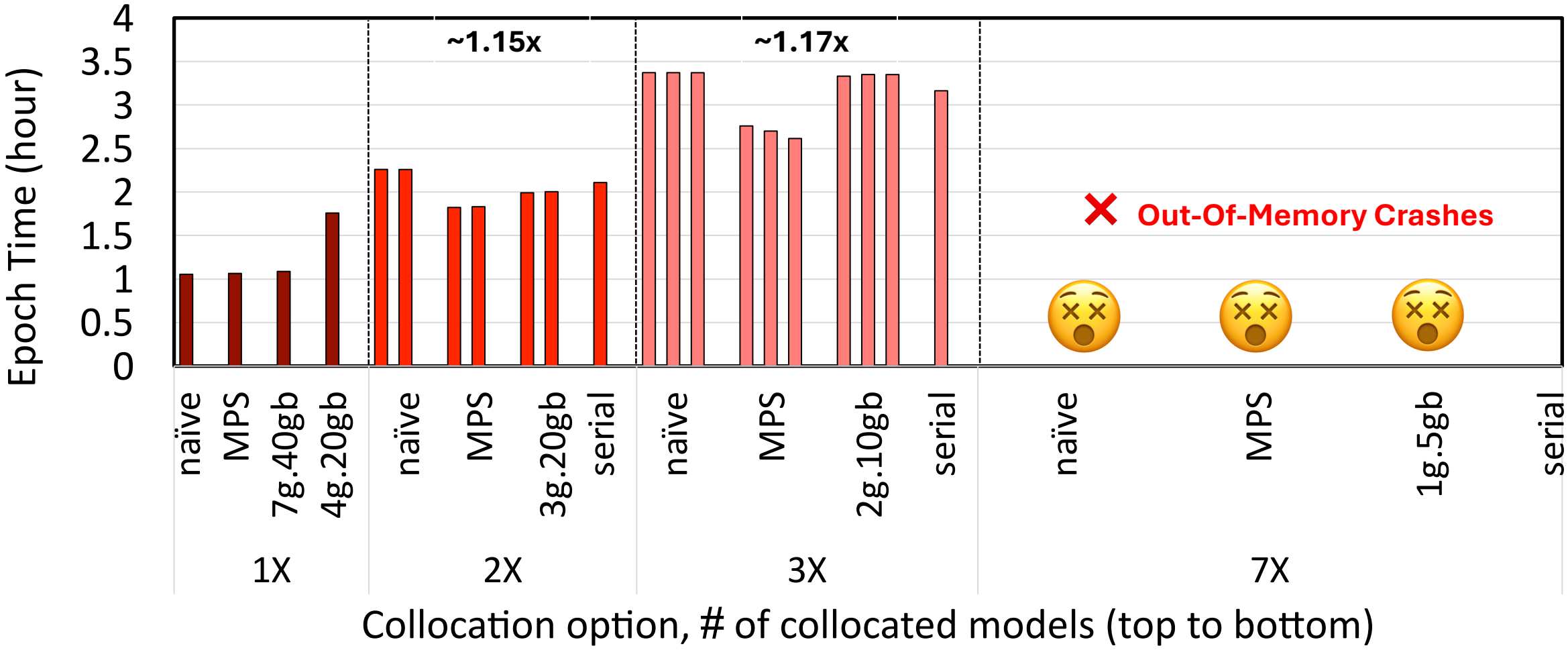
Streams (naïve)	Multi-Process Service (MPS)	Multi-Instance GPU (MIG)
➢ Easy to use	➢ Launching a daemon process	➢ Configuration overhead
False dependencies limit parallelism	➢ Takes care of false dependencies Interference still can degrade performance	➢ Complete isolation No interference
		✕ Fewer resources per instance may result in performance degradation

Small: ResNet26 on Cifar10 (batch size: 32)



Up to ~5.35x throughput gain despite increased epoch time
MPS > MIG > streams (naïve)

Large: ResNet152 on ImageNet (batch size: 32)



Negligible throughput gain with intolerable slow downs.

We cannot have 7X because of Out-Of-Memory (OOM) crashes.

Summary

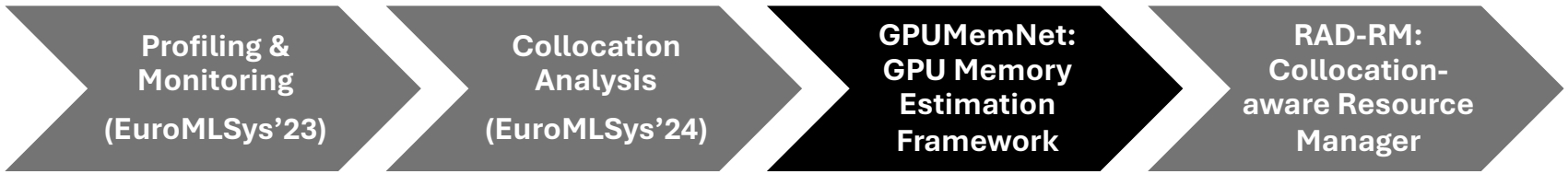
- Collocation is beneficial
  - Aggregate memory need of all collocated tasks is less than GPU memory
  - GPU utilization (SMACT) is less than 80%

- MPS offers higher performance

- MIG’s rigid partitioning may lead to sub-optimal performance!

3

GPUMemNet + Other Estimators



Having an estimation for GPU Memory => More Reliable Collocation

Estimating GPU Memory is challenging!

- Optimizations
  - Applied by Default:
    - Activation reuse, dynamic memory management
  - May be enabled by the user:
    - Layer fusion, gradient checkpointing, mixed precision, etc.

These introduce levels of unpredictability to GPU memory estimation.

Existing Estimators

1. Analytical
    1. Horus formula (***TPDS '21***)
    2. DNNMem, Microsoft's Analytical work (***ESEC/FSE '20***)
    3. LLMem (***IJCAI '24***)

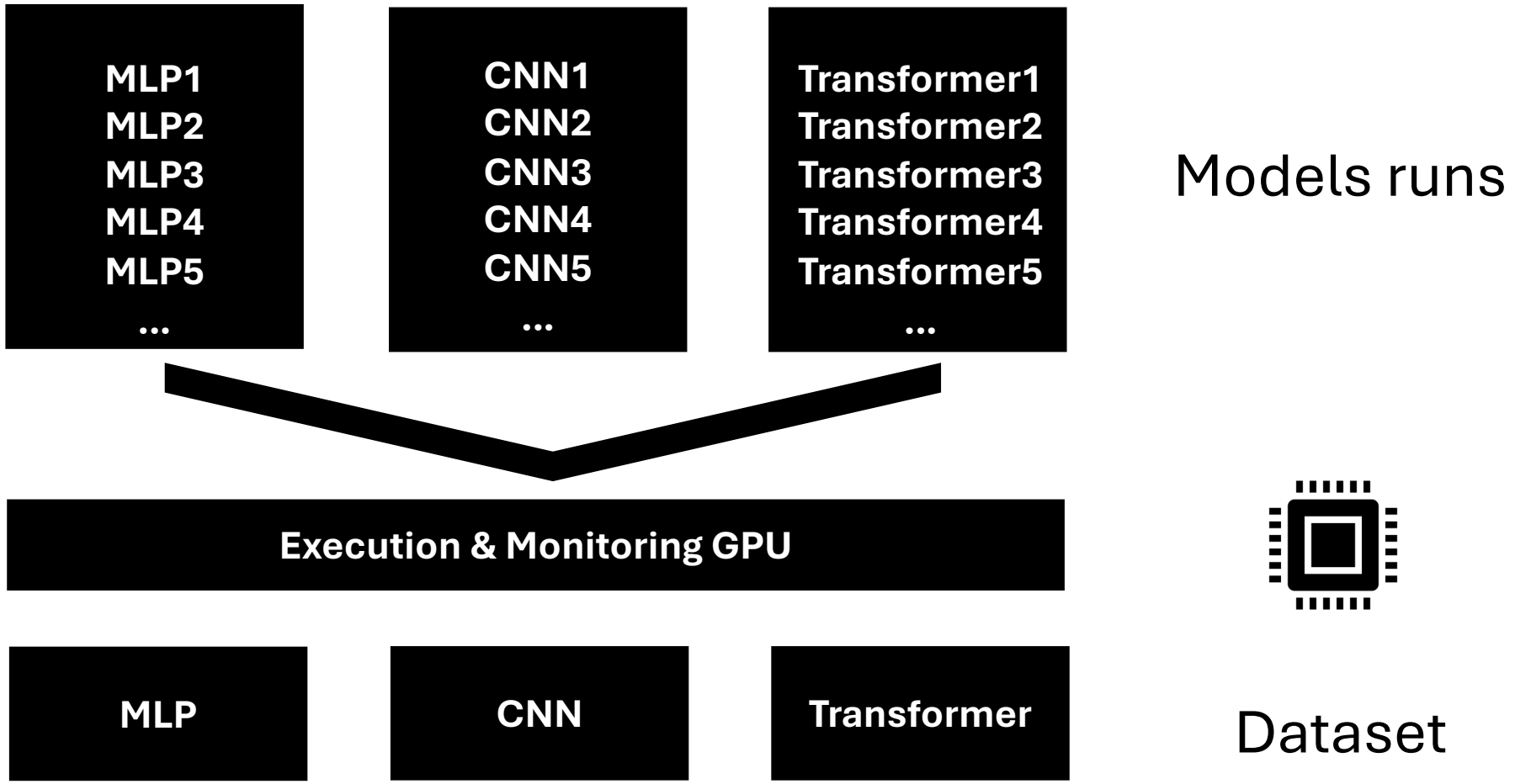
2. Libraries
    1. Fake Tensor
    2. DeepSpeed

3. ML-based approach
    - DNNPerf, graph neural networks (***ICSE-SEIP '23***)

https://pytorch.org/docs/stable/torch.compiler_fake_tensor.html
https://deepspeed.readthedocs.io/en/latest/memory.html

Challenges of Using ML for Estimation

Dataset

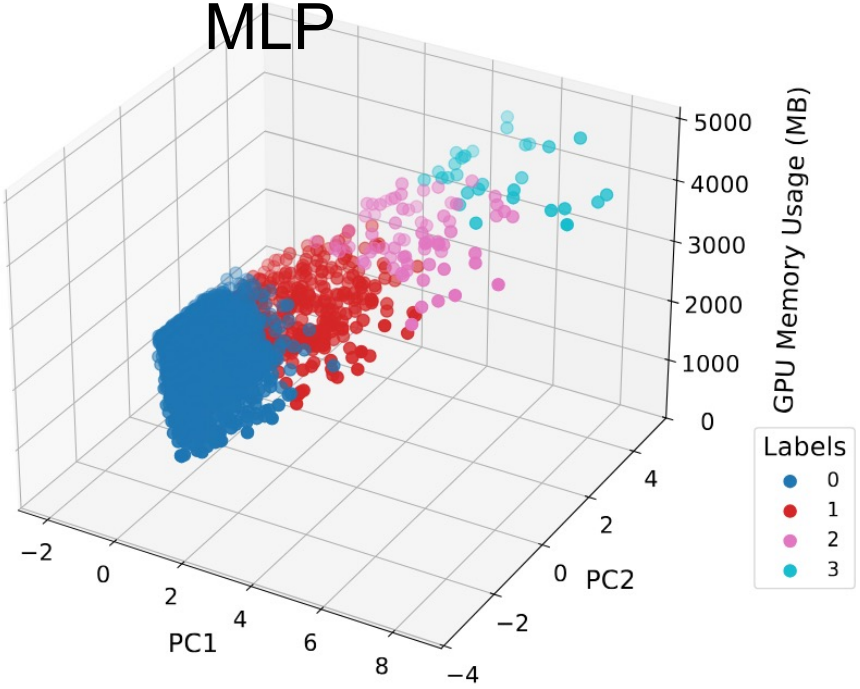


Classification Formulation

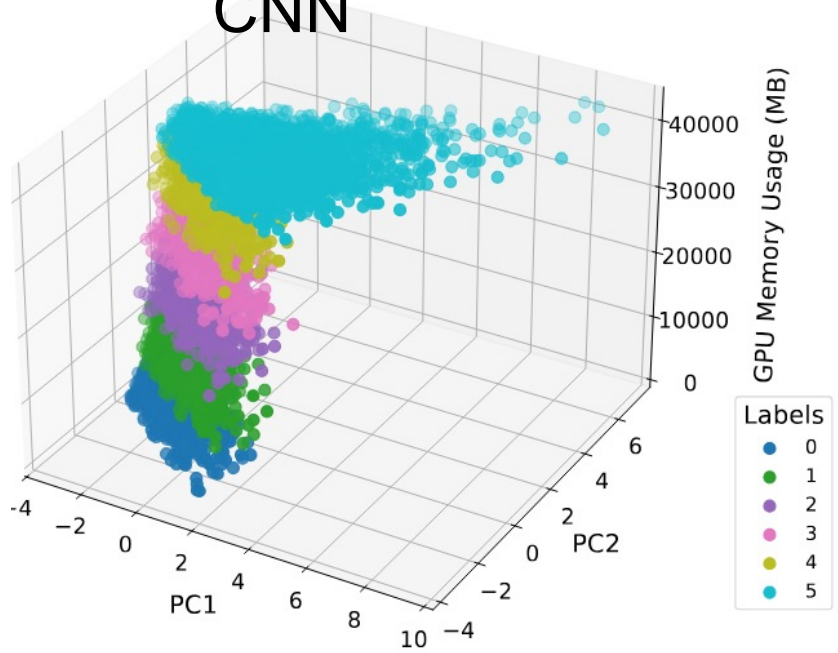
- Discretize data into same-GB classes
  - e.g., 0-1GB (1), 1GB-2GB (2), …

- Looked into the data through PCA and t-SNE
  - The classes are observable!

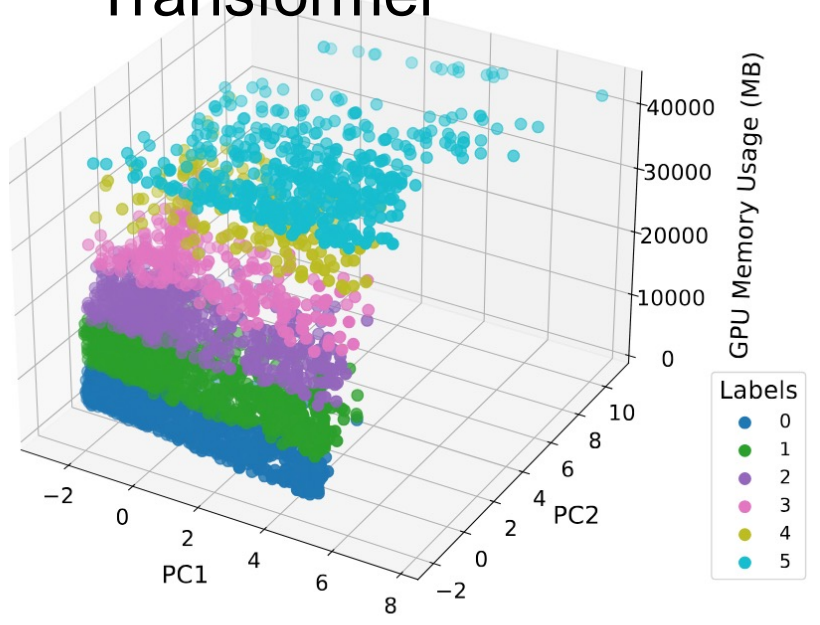
MLP



CNN

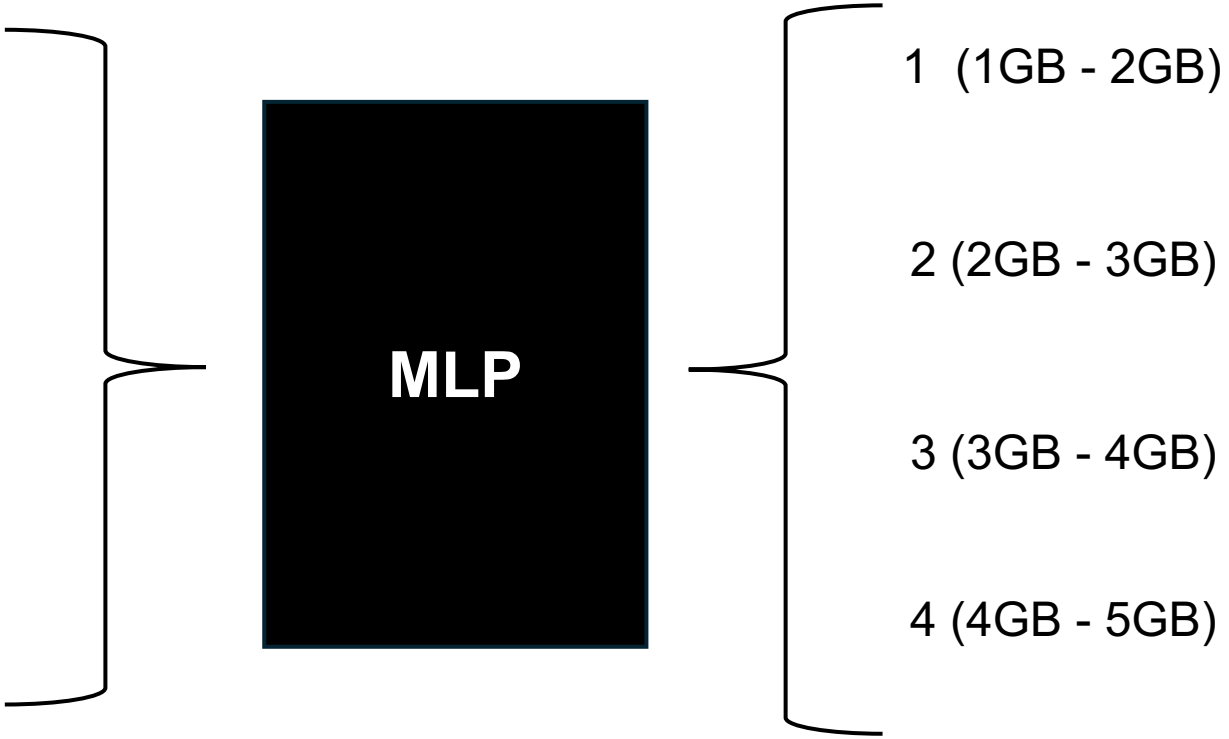


Transformer



Memory Estimator - MLP-based

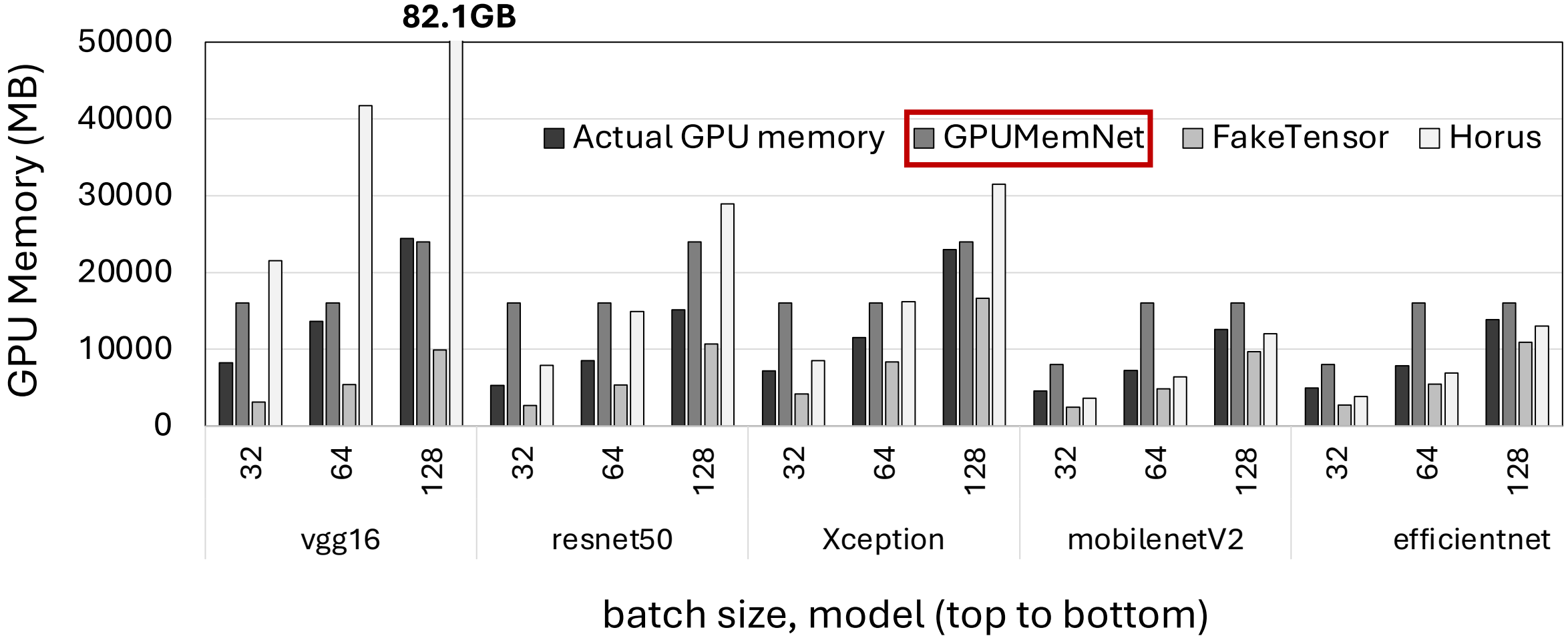
- #linear layers
- #batch normalization layers
- #dropout layers
- batch size
- #parameters
- #activations
- activation encoding cos/sin



Results

Dataset	Estimator	Class Range Size	#Classes	Accuracy
MLP	MLP	1GB	5	0.95
	MLP	2GB	4	0.97
	Transformer	1GB	5	0.97
	Transformer	2GB	4	0.98
CNN	MLP	8GB	6	0.82
	Transformer	8GB	6	0.81
Transformer	MLP	8GB	6	0.87
	Transformer	8GB	6	0.85

Unseen real-world models



GPUMemNet closest to actual GPU memory!
Almost never underestimates, preventing OOMs.

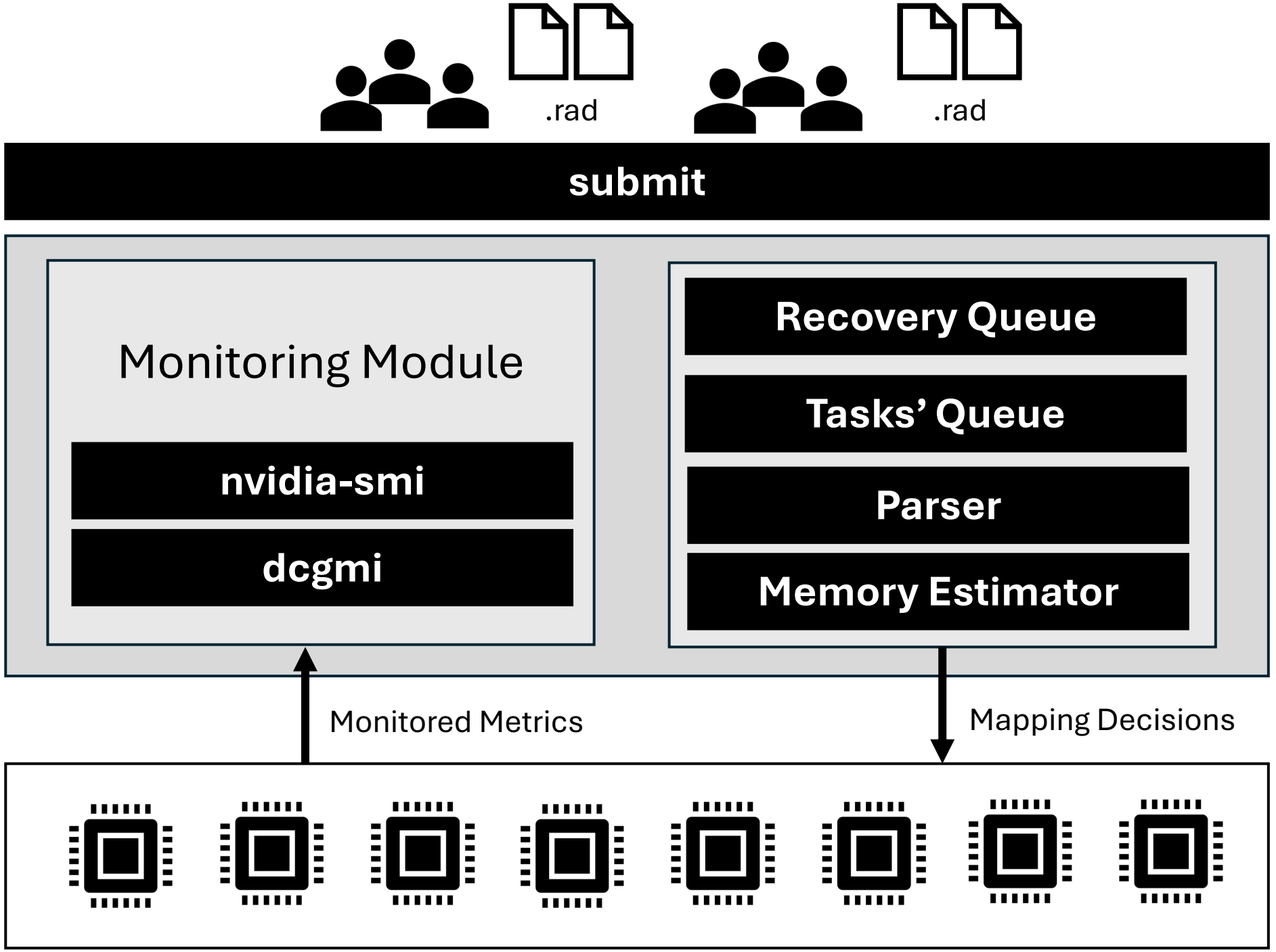
Summary

- GPUs are underutilized.
- Collocation can be an opportunity.
- GPU memory estimation is needed for more reliable collocation
- GPUMemNet
  - Dataset
  - Tools for extending the dataset

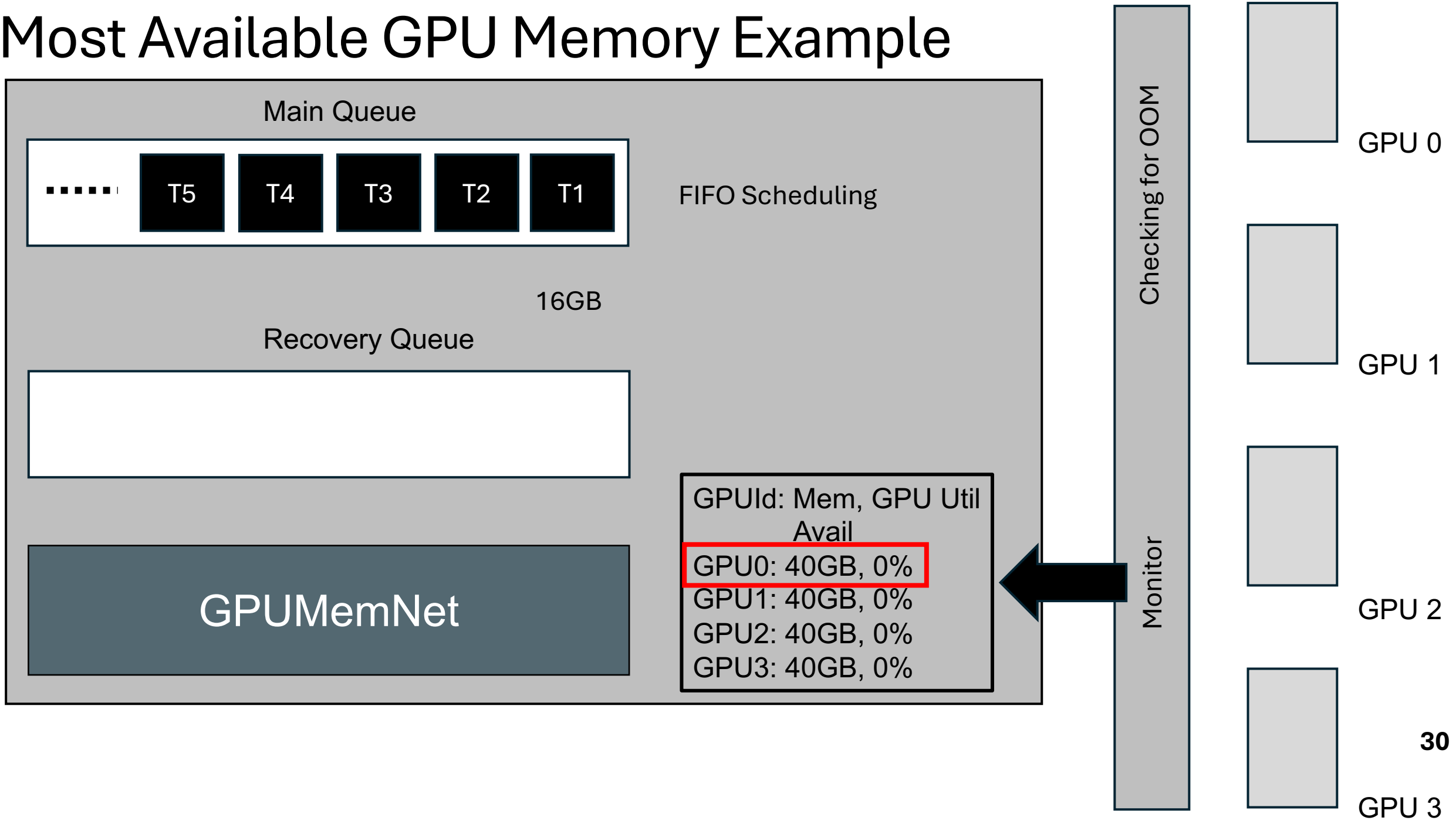
4

Resource-Aware Data systems-Resource Manager (RAD-RM)

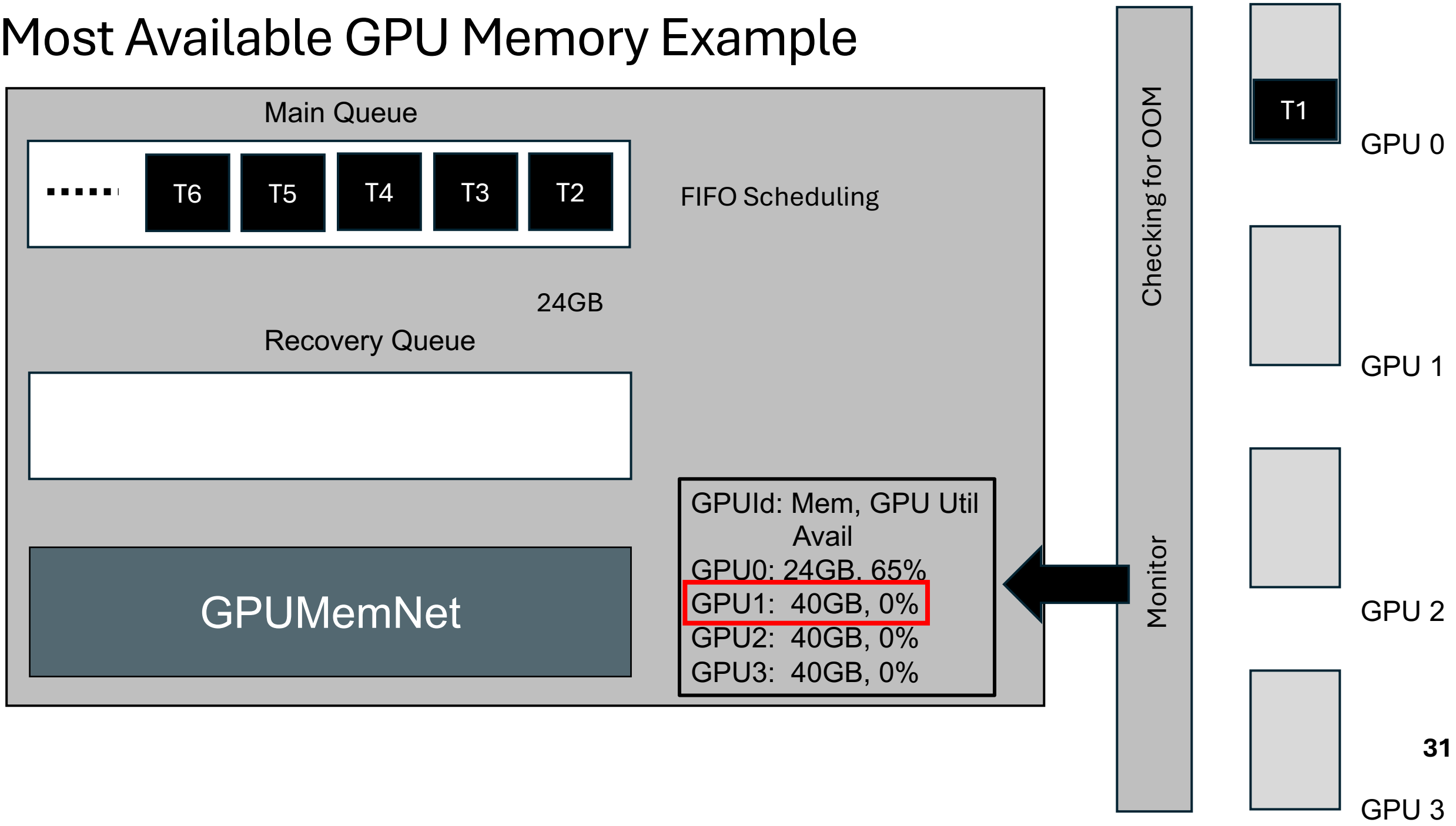
CARMA


.rad
.rad
submit
Monitoring Module
nvidia-smi
dcgmi
Recovery Queue
Tasks' Queue
Parser
Memory Estimator
Monitored Metrics
Mapping Decisions

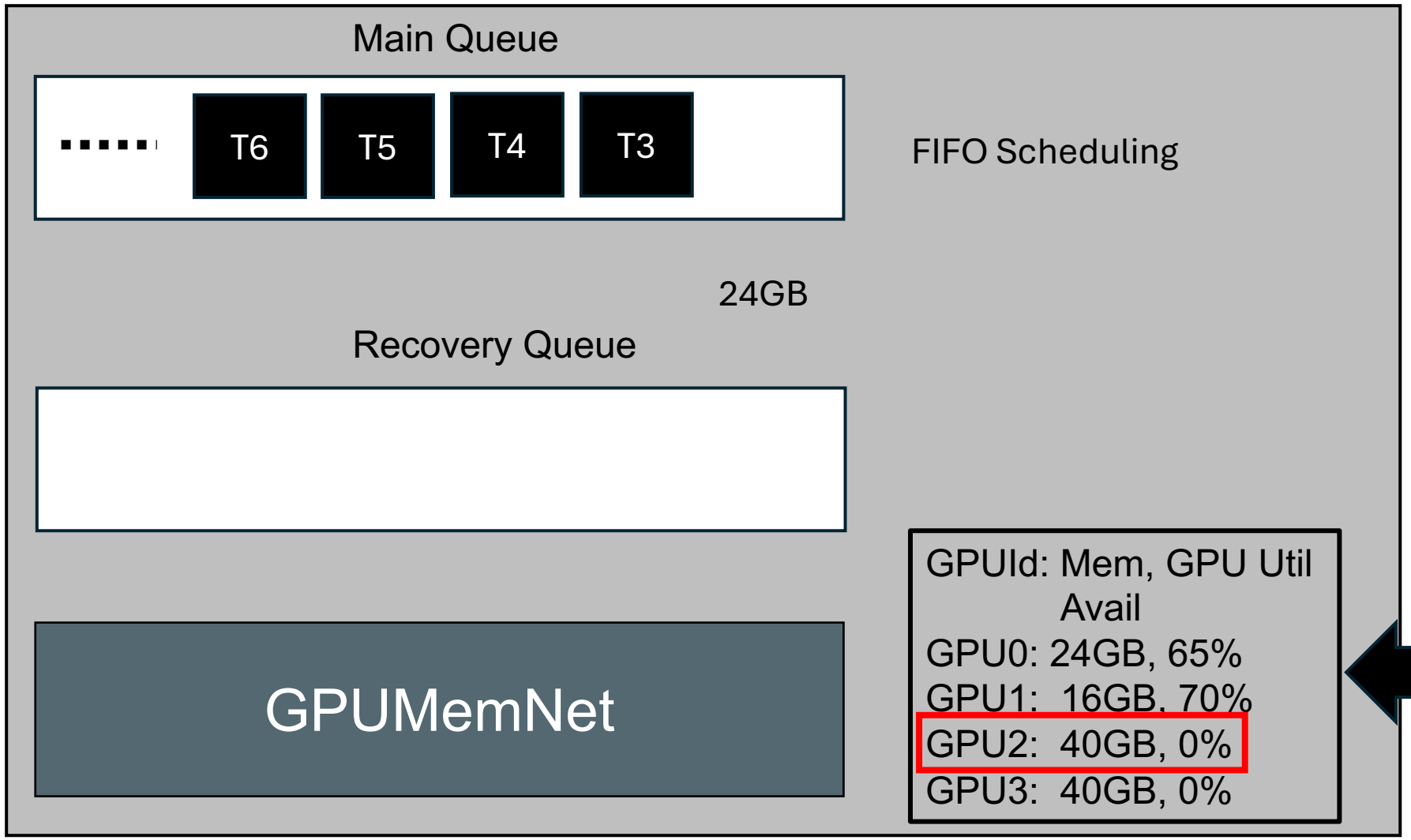
Most Available GPU Memory Example

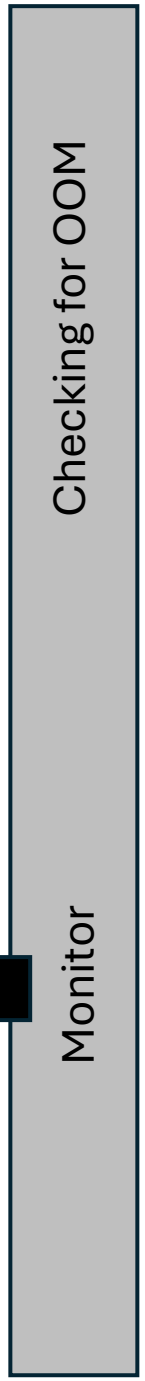


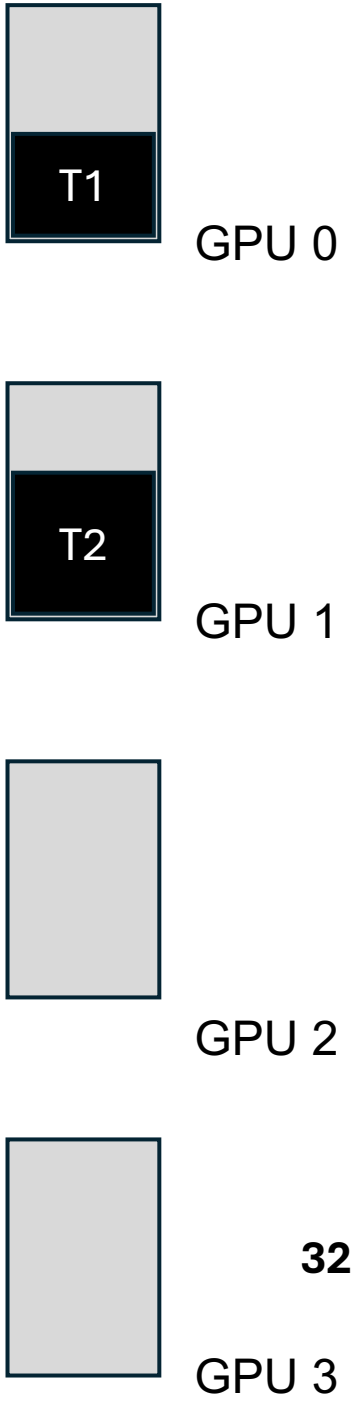
Most Available GPU Memory Example



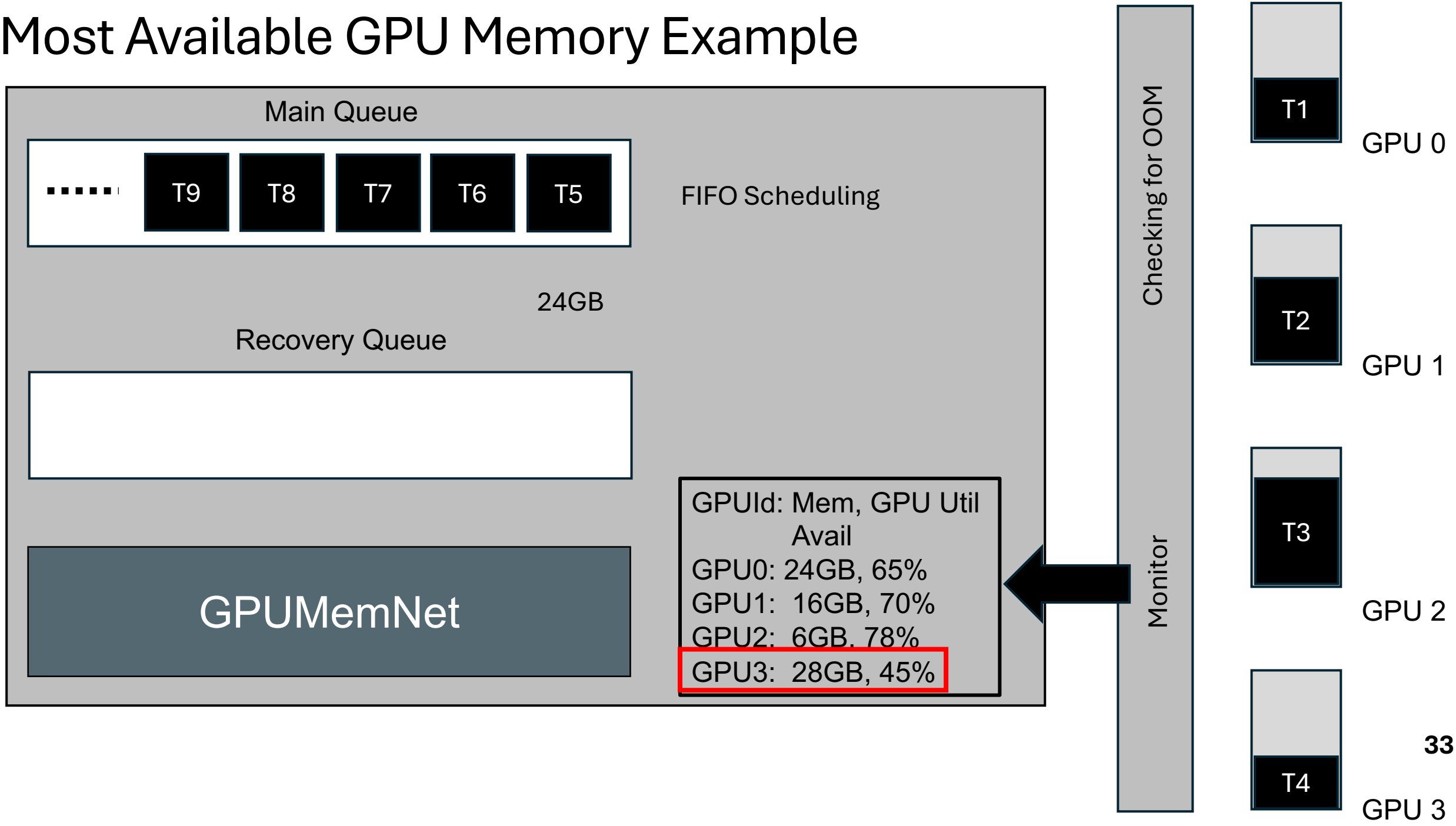
Most Available GPU Memory Example



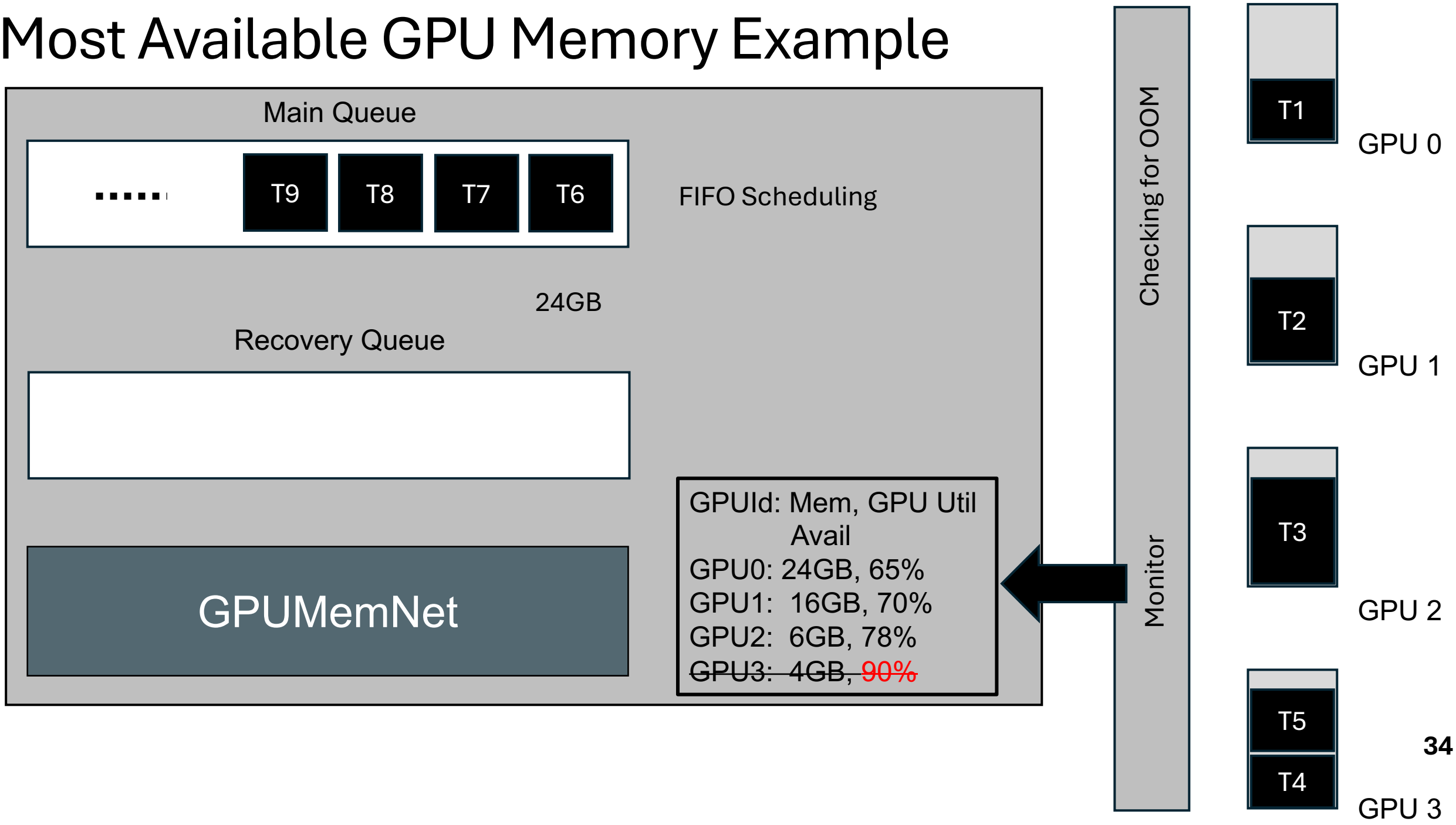




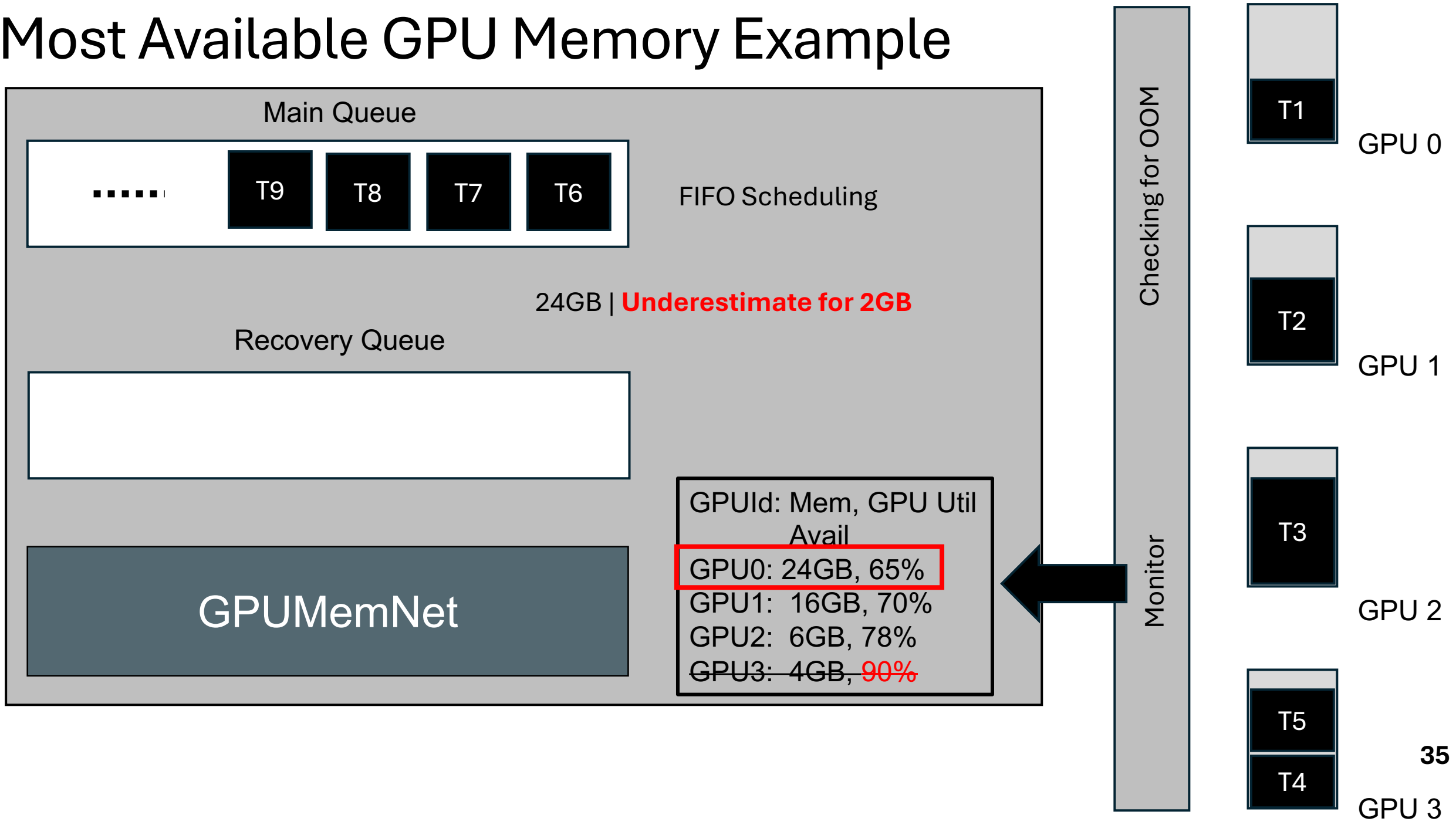
Most Available GPU Memory Example



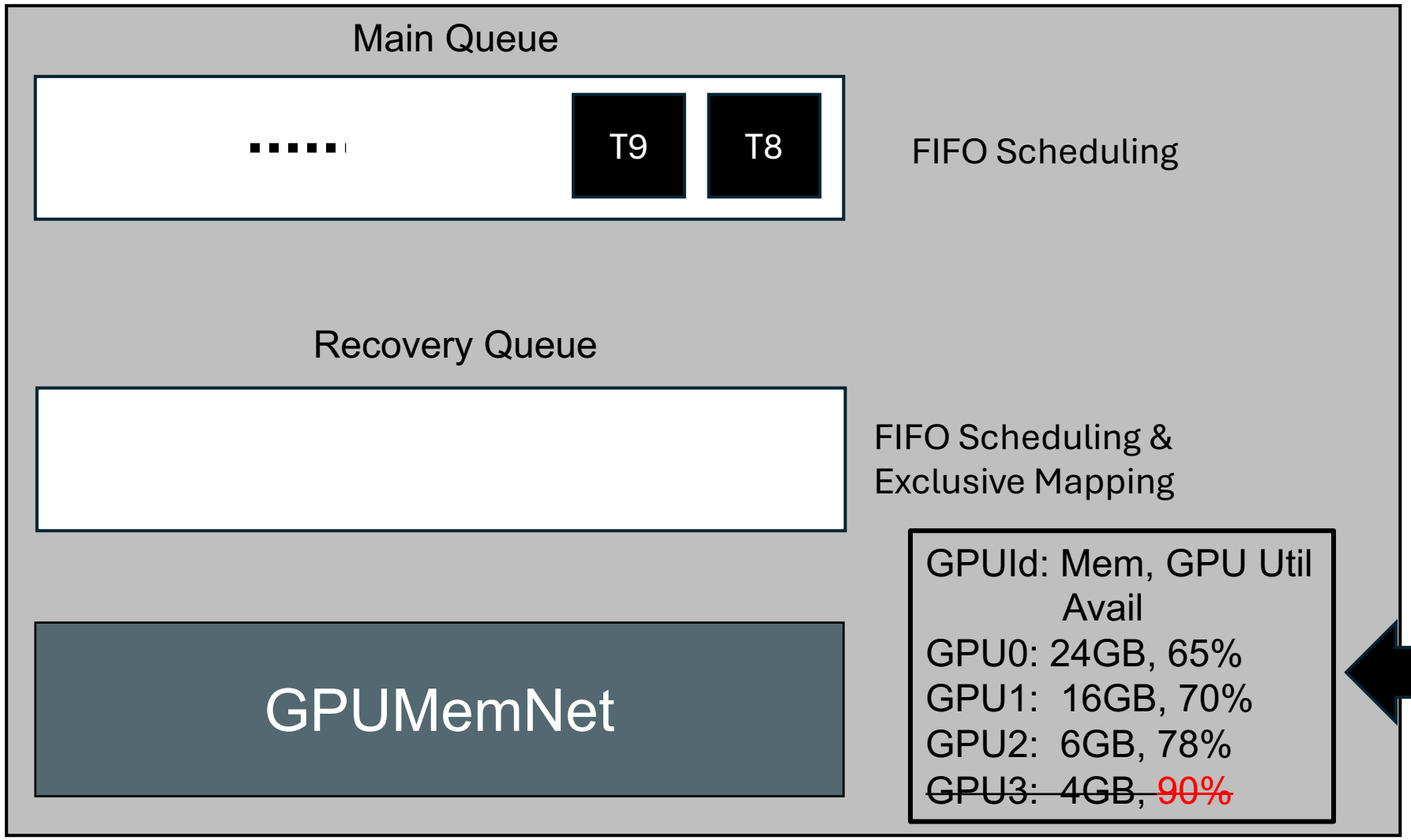
Most Available GPU Memory Example

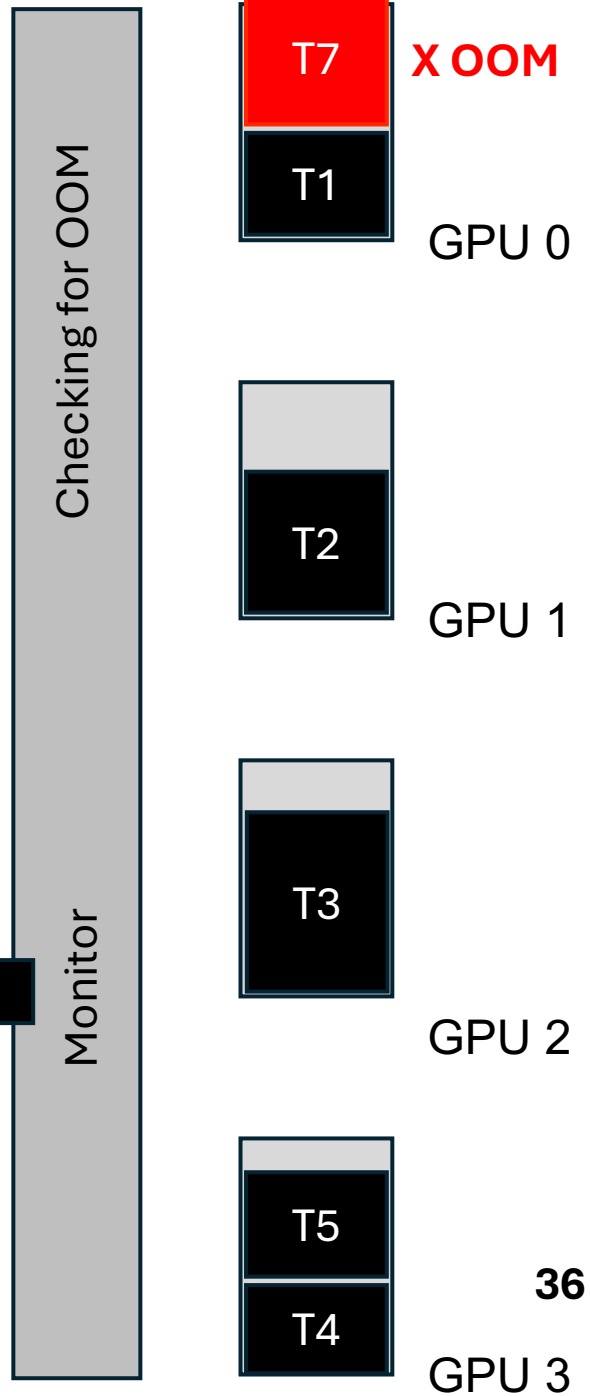


Most Available GPU Memory Example



Most Available GPU Memory Example





Evaluation Setup

- NVIDIA DGX Station A100
  - 4x 40GB A100 GPUs
  - 1x AMD EPYC 7742

- Mapped Training Tasks
  - A subset of Microsoft **Philly Trace** ✤ submission times

  - 90-tasks trace of light, medium and heavy tasks (65% light, 27% medium, 8% large) ♣
  - 60-task trace of medium and heavy tasks (83% medium and 17% large) ♣

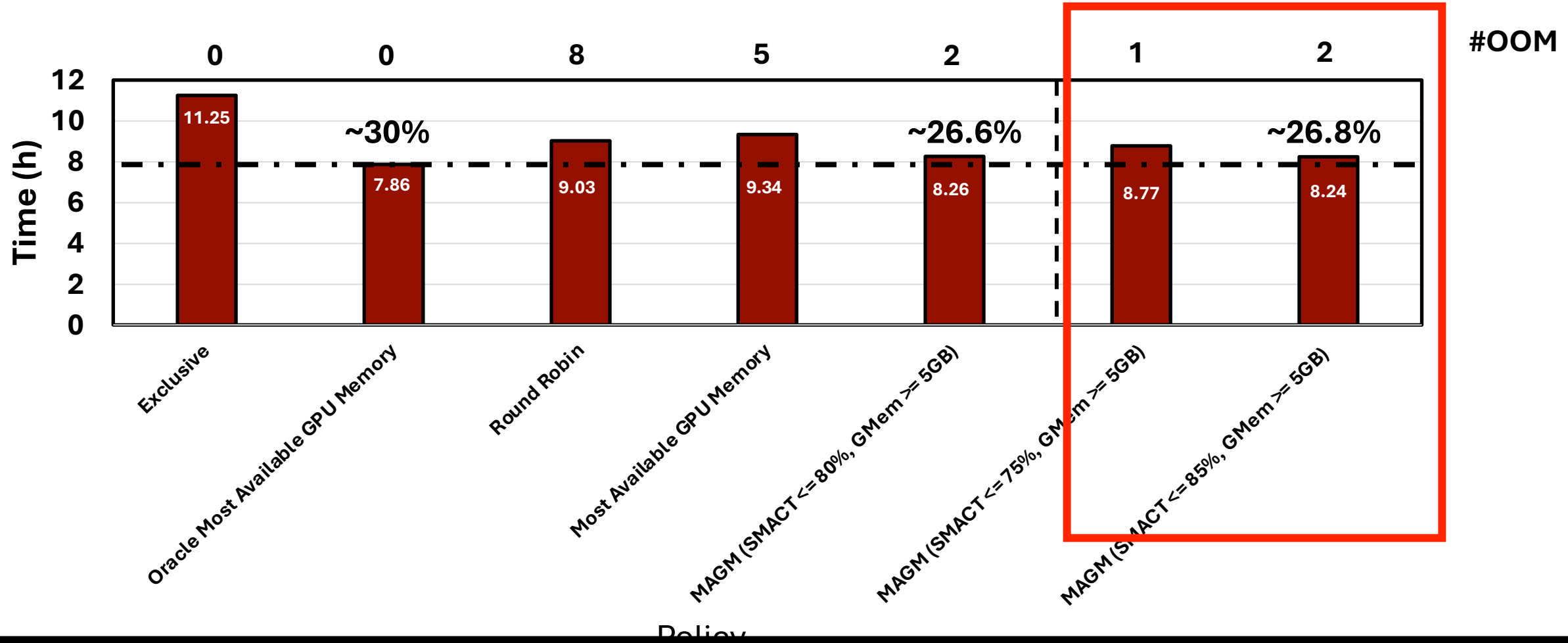
✤ Jeon, Myeongjae, et al. "**Analysis of Large-Scale Multi-Tenant GPU clusters for DNN training workloads**," *USENIX **ATC** 19*.

♣ Ye, Zhisheng, et al. "**Astraea: A fair deep learning scheduler for multi-tenant gpu clusters**," ***ITPDS*** 21.

A glimpse of 90-task trace!

```
python src/submit.py --task workloads/efficientNet3.rad
sleep 441
python src/submit.py --task workloads/resnet18_cifar100_20e_3.rad
sleep 52
python src/submit.py --task workloads/resnet18_cifar100_50e_1.rad
sleep 51
python src/submit.py --task workloads/resnet34_cifar100_20e_2.rad
sleep 365
python src/submit.py --task workloads/resnet34_cifar100_20e_1.rad
sleep 18
python src/submit.py --task workloads/resnet.rad
sleep 166
python src/submit.py --task workloads/resnet18_cifar100_20e_3.rad
sleep 439
python src/submit.py --task workloads/resnet18_cifar100_20e_1.rad
sleep 83
```

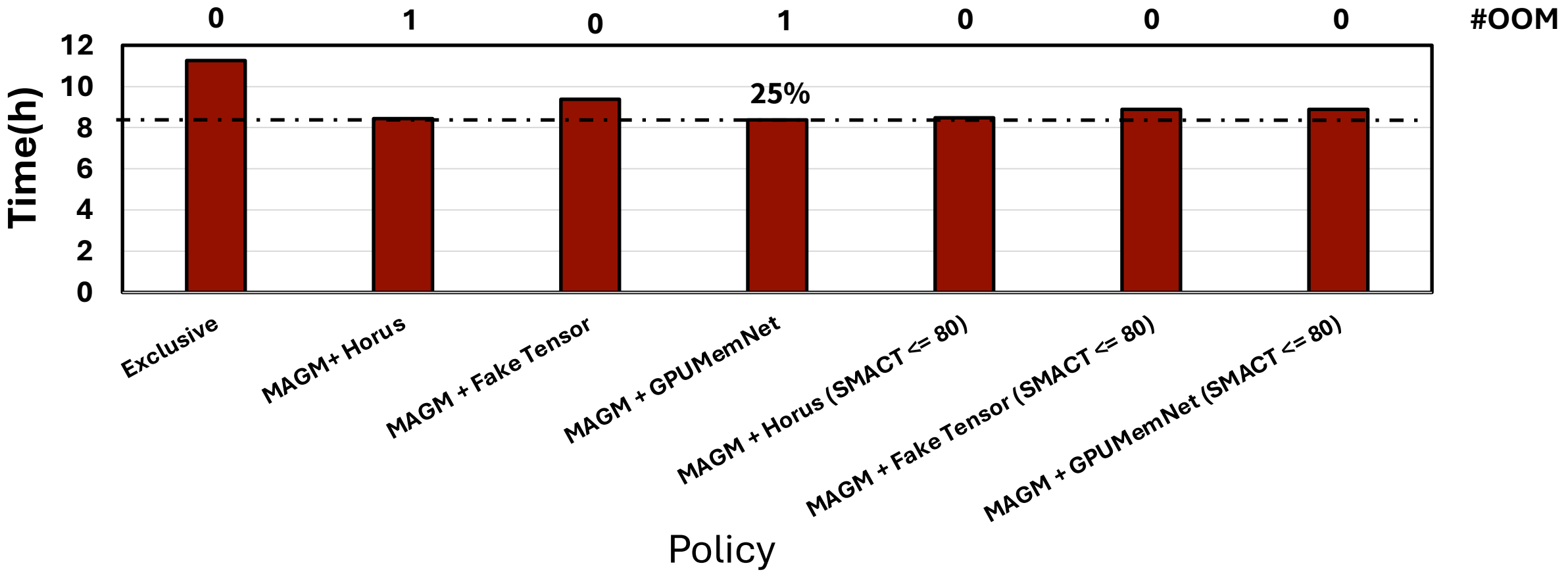
90-task trace relying on the recovery method scenarios



Collocation with least resource interference is beneficial!

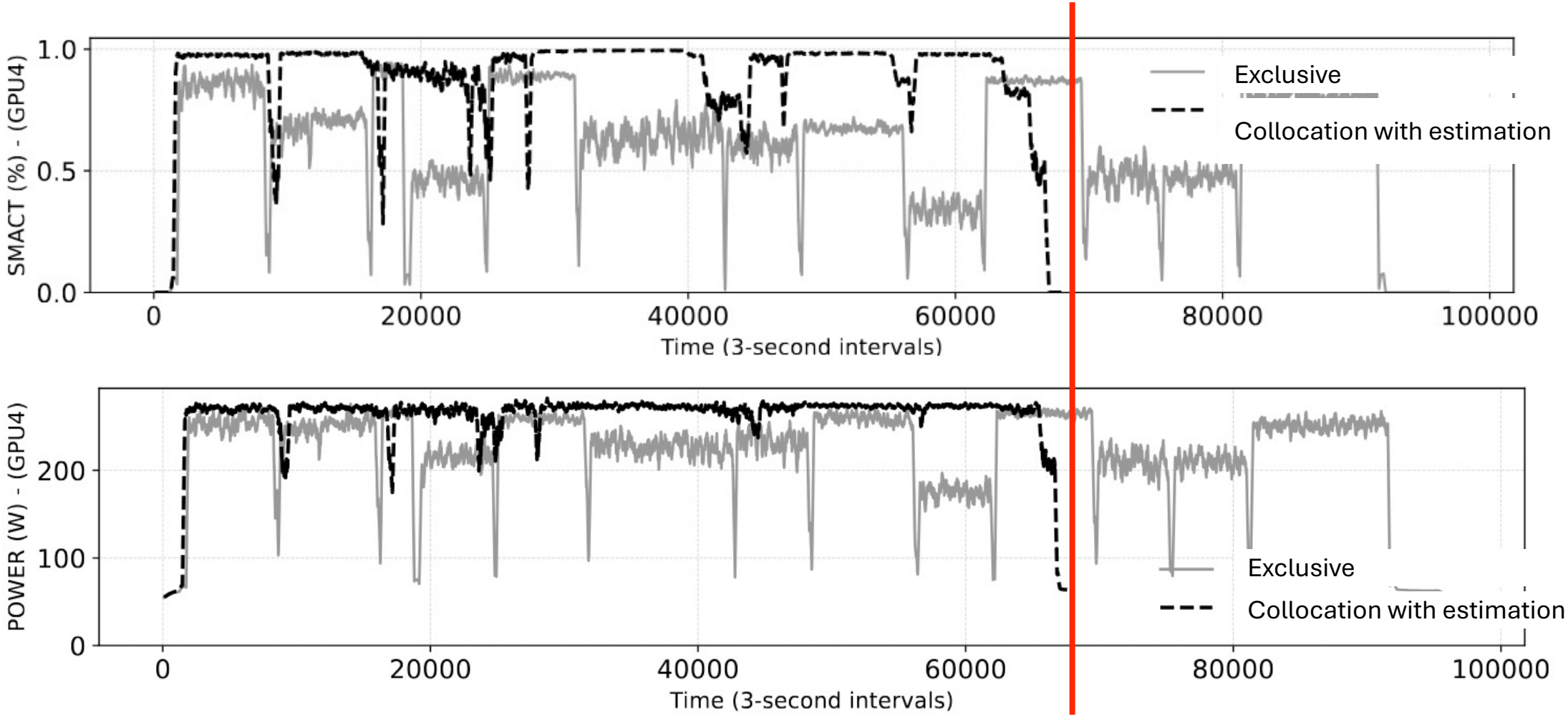
Preconditions coupled with collocation policies affect the number of OOM crashes.

90-task trace with integrated GMem Estimators



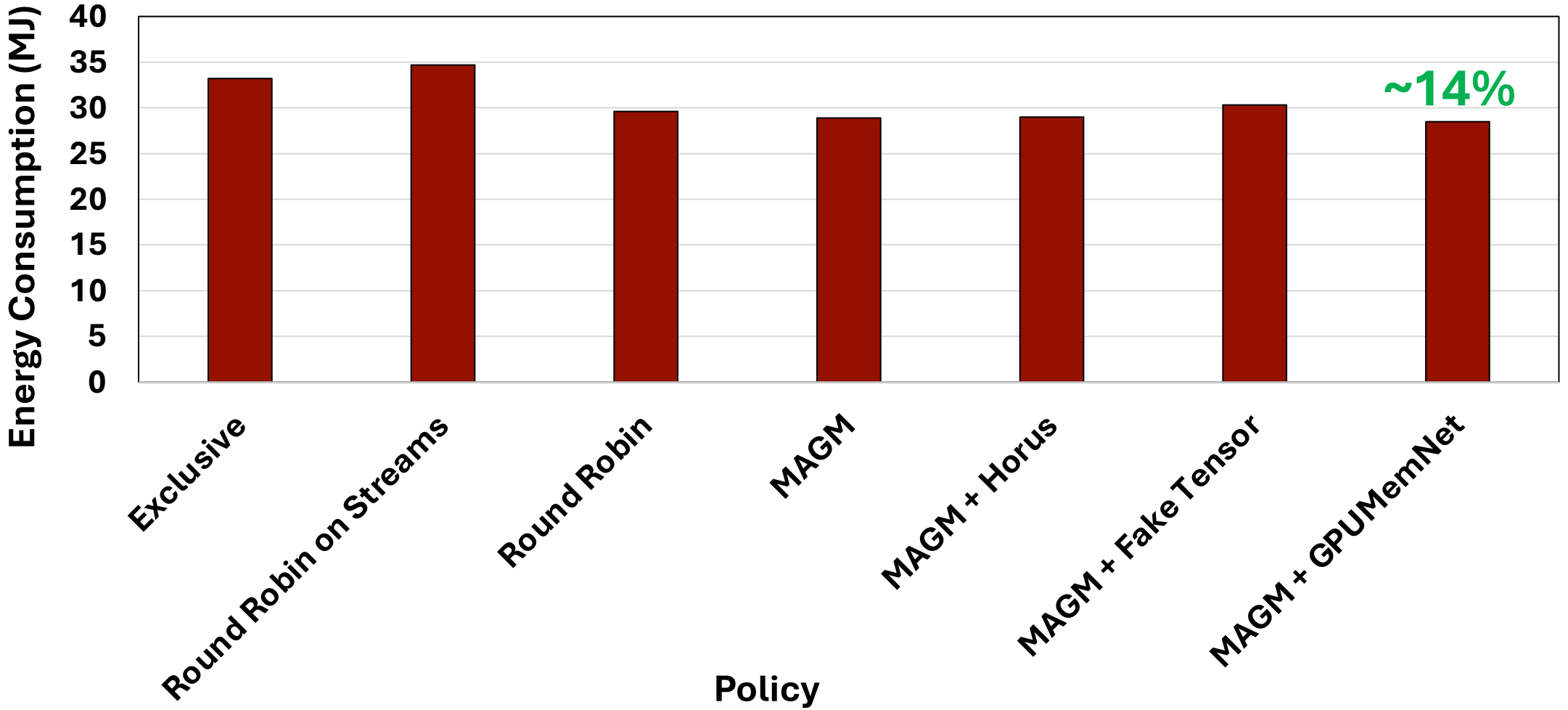
Having even inaccurate estimations help preventing OOMs!

GPU Utilization & Power over time



Collocation-aware resource management improves GPU Utilization.

Energy Consumption

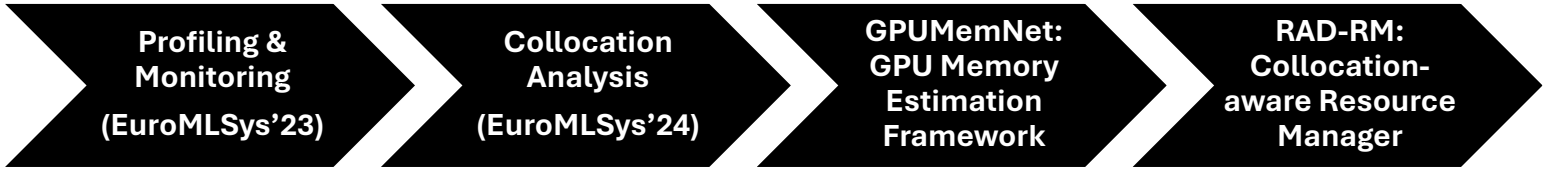


Collocation-aware resource management improves energy efficiency.

Summary

- Collocation policies and preconditions affect the number of OOMs.
- Least interference promises higher performance via throughput gain.
- The sequence of coming tasks also affects the number of OOMs.
- Collocation-aware resource managers improve GPU utilization and energy efficiency.

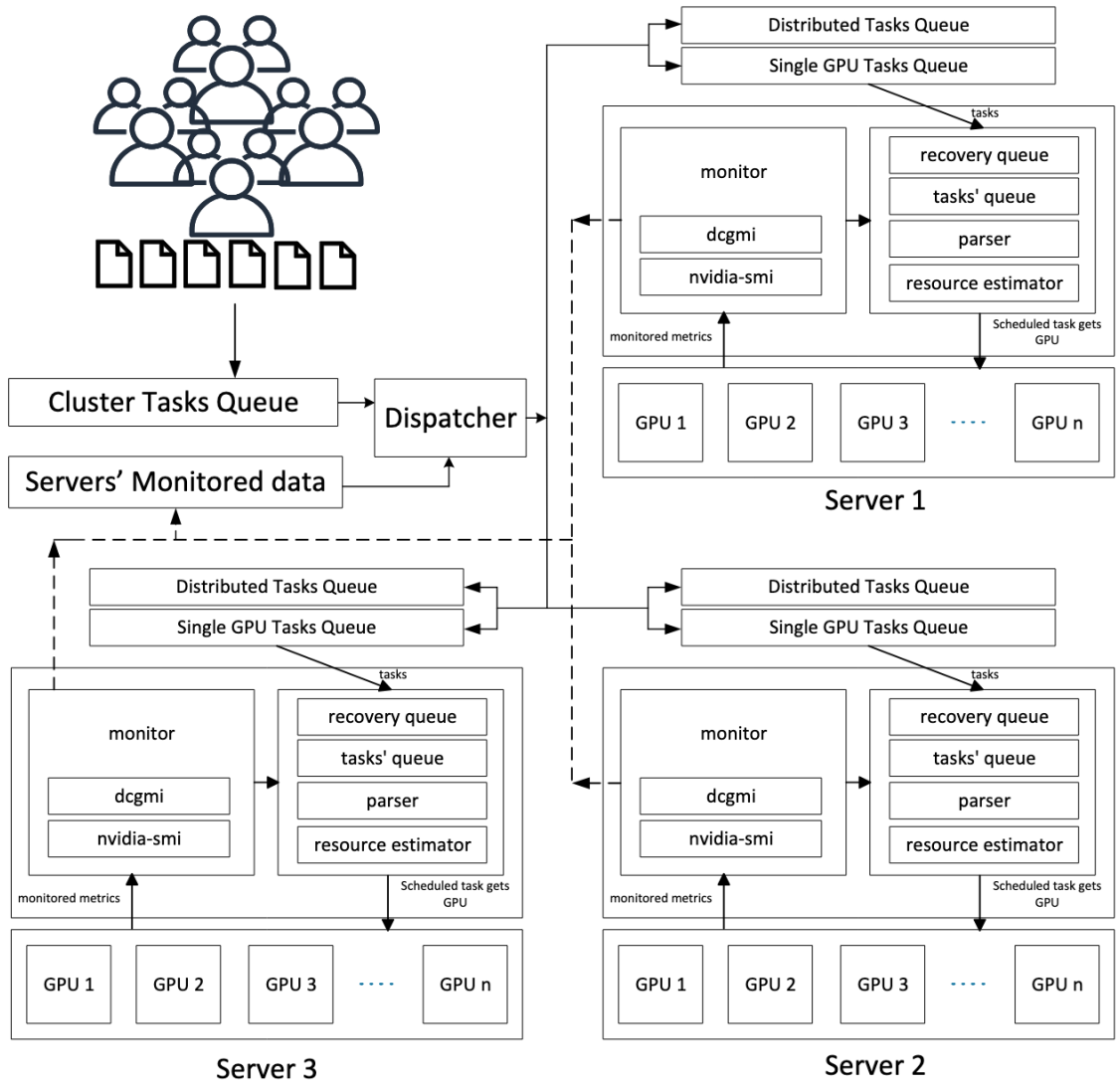
Conclusion



- Fine-grained metrics (SMACT for utilization) from monitoring tools
- Collocation is beneficial when applied carefully.
  - If aggregate SMACT <= ~80%
  - OOM crashes and performance degradation due to resource interference
- GPU Memory estimation is necessary for a more reliable collocation.
- Collocation-aware resource management improves GPU utilization and energy efficiency.

Future Directions

- GPU Utilization Estimation
  - and Integrating in the system
- Looser Recovery Method
- Fairness and Checkpointing
- Extending to cluster scale



Thanks ☺