# MISO: Exploiting Multi-Instance GPU Capability on Multi-Tenant GPU Clusters

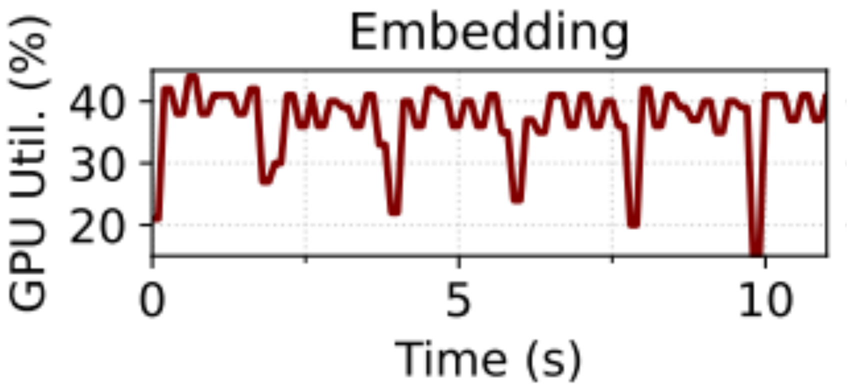
Baolin Li, Tirthak Patel, Siddharth Samsi,
Vijay Gadepally, Devesh Tiwari

Northeastern University

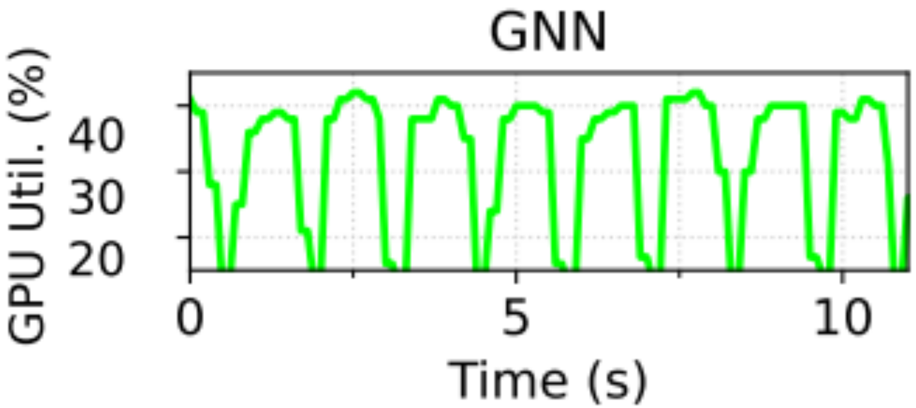
MIT LINCOLN LABORATORY

# GPUs are everywhere in the cloud..

# …but, they are severely underutilized




Word embedding model for topic classification




Graph neural network for quantum chemistry

The state-of-the-art deep learning models utilize less 50% of the GPU resources on modern A100 GPUs and utilization varies significantly over run time

# …but, they are severely underutilized

usenix
THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

**Analysis of Large-Scale Multi-Tenant GPU Clusters for DNN Training Workloads**

Myeongjae Jeon, *UNIST and Microsoft Research;* Shivaram Venkataraman, *University of Wisconsin and Microsoft Research;* Amar Phanishayee and Junjie Qian, *Microsoft Research;* Wencong Xiao, *Beihang University and Microsoft Research;* Fan Yang, *Microsoft Research*

https://www.usenix.org/conference/atc19/presentation/jeon

M Jeon et al., USENIX ATC'19

Up to 50% of the GPU jobs may have less than 25% utilization on multi-tenant clusters

AI-Enabling Workloads on Large-Scale GPU-Accelerated System: Characterization, Opportunities, and Implications

Baolin Li[*], Rohin Arora[*], Siddharth Samsi[†], Tirthak Patel[*], William Arcand[†], David Bestor[†], Chansup Byun[†], Rohan Basu Roy[*], Bill Bergeron[†], John Holodnak[†], Michael Houle[†], Matthew Hubbell[†], Michael Jones[†], Jeremy Kepner[†], Anna Klein[†], Peter Michaleas[†], Joseph McDonald[†], Lauren Milechin[†], Julie Mullen[†], Andrew Prout[†], Benjamin Price[†], Albert Reuther[†], Antonio Rosa[†], Matthew Weiss[†], Charles Yee[†], Daniel Edelman[†], Allan Vanterpool[‡], Anson Cheng[‡], Vijay Gadepally[†], Devesh Tiwari[*]

[*]Northeastern University, [†]MIT Lincoln Laboratory, [‡]US Air Force

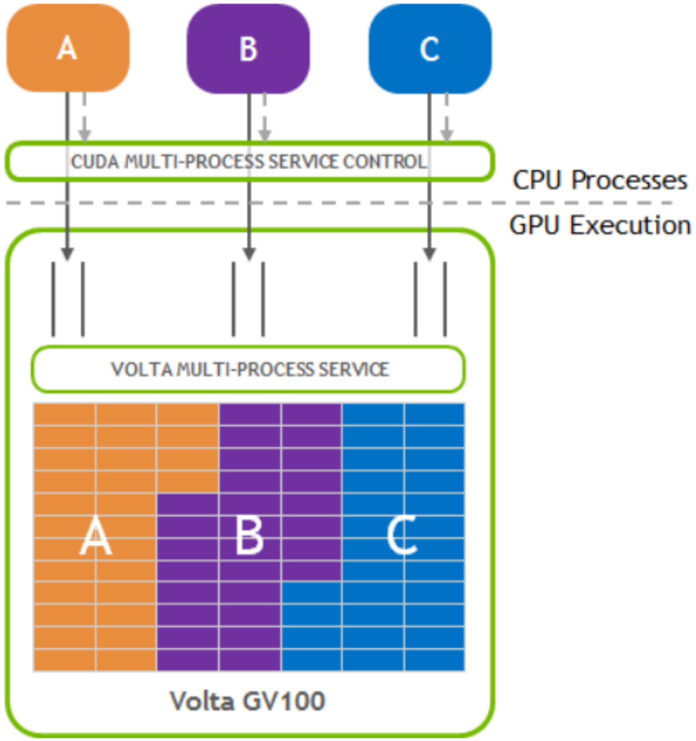
Baolin Li et al., HPCA'22

## What is a potential solution?

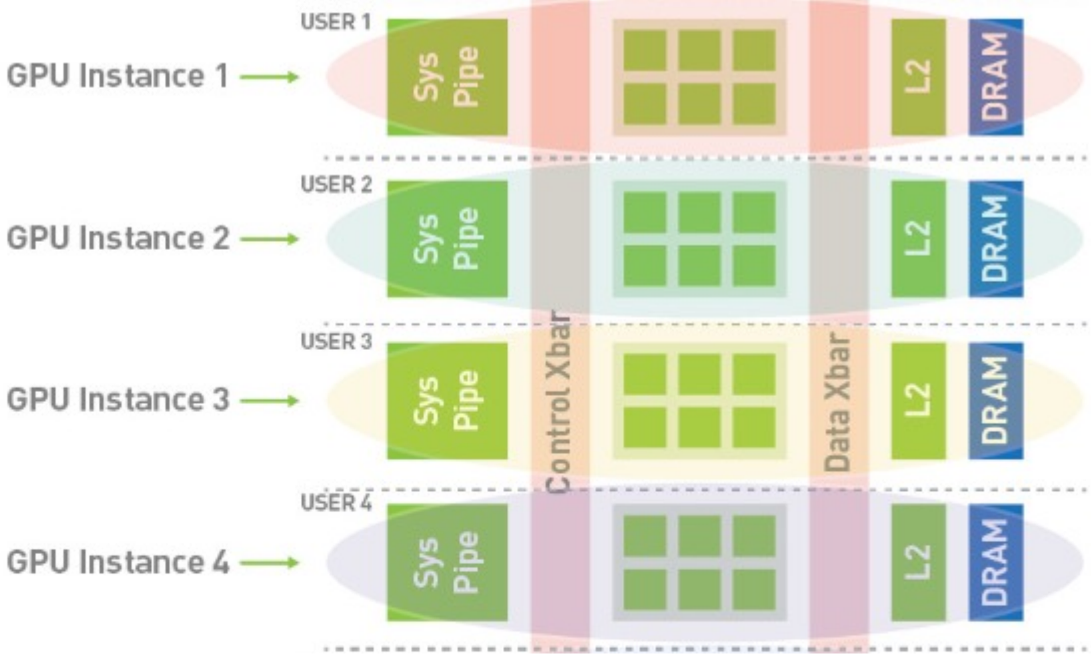
GPU resource sharing allows better utilization

# GPU Resource Sharing Allows Better Utilization

## Multi-Process Service (MPS)

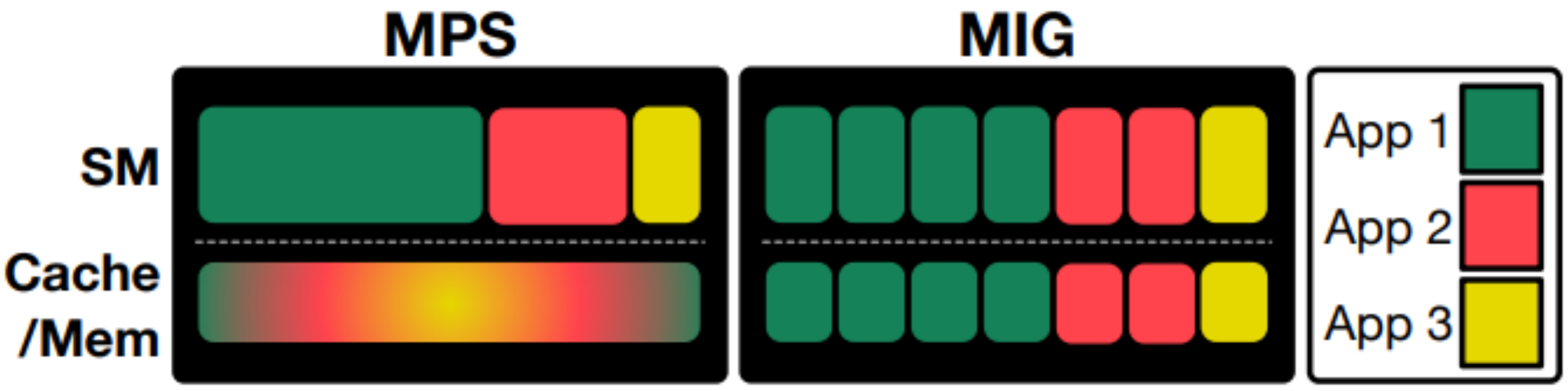



## Multi-Instance GPU (MIG)




[1] https://docs.nvidia.com/deploy/mps/index.html
[2] https://docs.nvidia.com/datacenter/tesla/mig-user-guide/index.html

# MPS and MIG Sharing Mode Trade-Offs




| | | |
|---|---|---|
| Software-based logical partition | Vs | Hardware-based physical partition |
| Flexible | | Limited granularity |
| No perf isolation | | Interference-free |

# Multi-Instance GPU (MIG) on NVIDIA GPUs

## Different MIG slices on an A100 GPU

| Slice | Compute | Memory | Cache | Max Count |
|---|---|---|---|---|
| 7g.40gb | 7 GPC | 40 GB | Full | 1 |
| 4g.20gb | 4 GPC | 20 GB | 4/8 | 1 |
| 3g.20gb | 3 GPC | 20 GB | 4/8 | 2 |
| 2g.10gb | 2 GPC | 10 GB | 2/8 | 3 |
| 1g.5gb | 1 GPC | 5 GB | 1/8 | 7 |

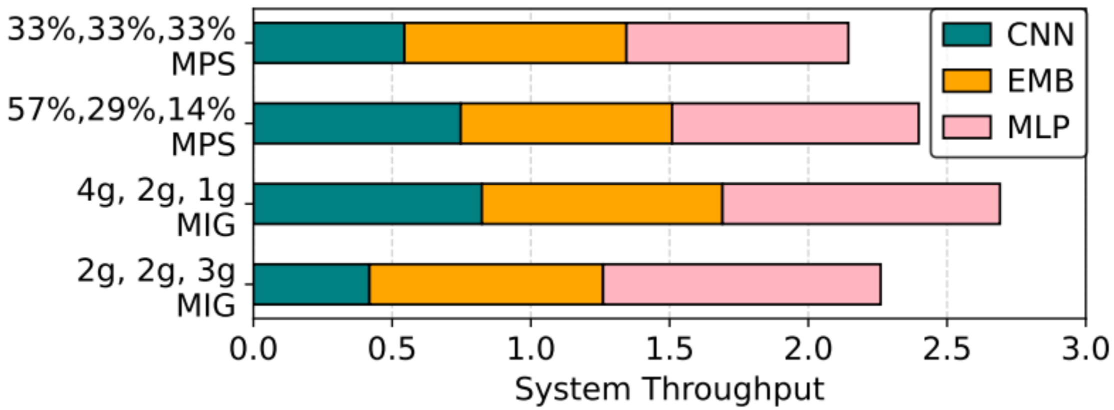
| Config | GPC Slice #0 | GPC Slice #1 | GPC Slice #2 | GPC Slice #3 | GPC Slice #4 | GPC Slice #5 | GPC Slice #6 |
|---|---|---|---|---|---|---|---|
| 1 | 7 | | | | | | |
| 2 | 4 | | | | 2 | | 1 |
| 3 | 4 | | | | 1 | 1 | 1 |
| 4 | 3 | | | 3 | | | |
| 5 | 3 | | | 2 | | 1 | |
| 6 | 3 | | | 1 | 1 | 1 | |
| 7 | 2 | | 2 | | 3 | | |
| 8 | 2 | | 1 | 1 | 3 | | |
| 9 | 1 | 1 | 2 | | 3 | | |
| 10 | 1 | 1 | 1 | 1 | 3 | | |
| 11 | 2 | | 2 | | 2 | | 1 |
| 12 | 2 | | 1 | 1 | 2 | | 1 |
| 13 | 1 | 1 | 2 | | 2 | | 1 |
| 14 | 2 | | 1 | 1 | 1 | 1 | 1 |
| 15 | 1 | 1 | 2 | | 1 | 1 | 1 |
| 16 | 1 | 1 | 1 | 1 | 2 | | 1 |
| 17 | 1 | 1 | 1 | 1 | 1 | 2 | |
| 18 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

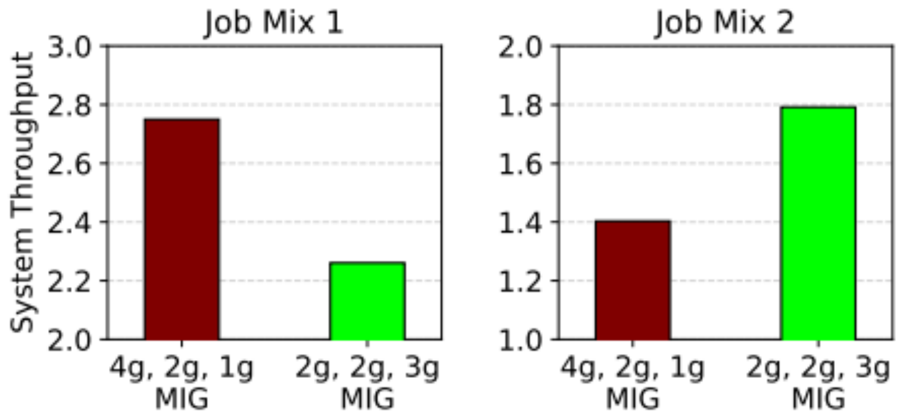
[5] https://docs.nvidia.com/datacenter/tesla/mig-user-guide/index.html

# Challenges in GPU Resource Partitioning

## Brief experimental insights and motivation

# Observation 1. Compared to MPS, MIG-based partitioning is more promising, but challenging



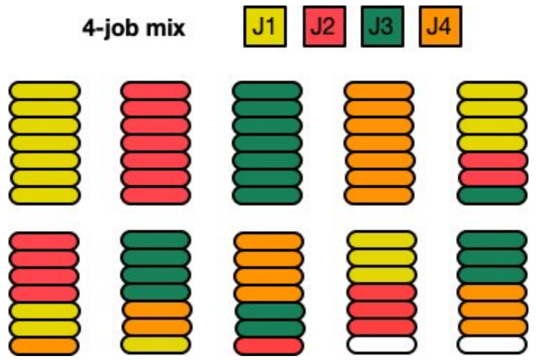




MIG's interference-free partitioning provides an opportunity for higher performance than MPS's interference-prone partitioning

Optimal GPU resource partitioning using MIG slices varies significantly across job mixes

# Observation 11. Determining effective MIG-based partitions incurs higher overhead

| Slice | Compute | Memory | Cache | Max Count |
|---|---|---|---|---|
| 7g.40gb | 7 GPC | 40 GB | Full | 1 |
| 4g.20gb | 4 GPC | 20 GB | 4/8 | 1 |
| 3g.20gb | 3 GPC | 20 GB | 4/8 | 2 |
| 2g.10gb | 2 GPC | 10 GB | 2/8 | 3 |
| 1g.5gb | 1 GPC | 5 GB | 1/8 | 7 |




A job-mix four jobs requires exploring multiple MIG configurations

Determining the optimal MIG partition configuration for a job-mix, requires knowing individual job's speedup on all different MIG slices.

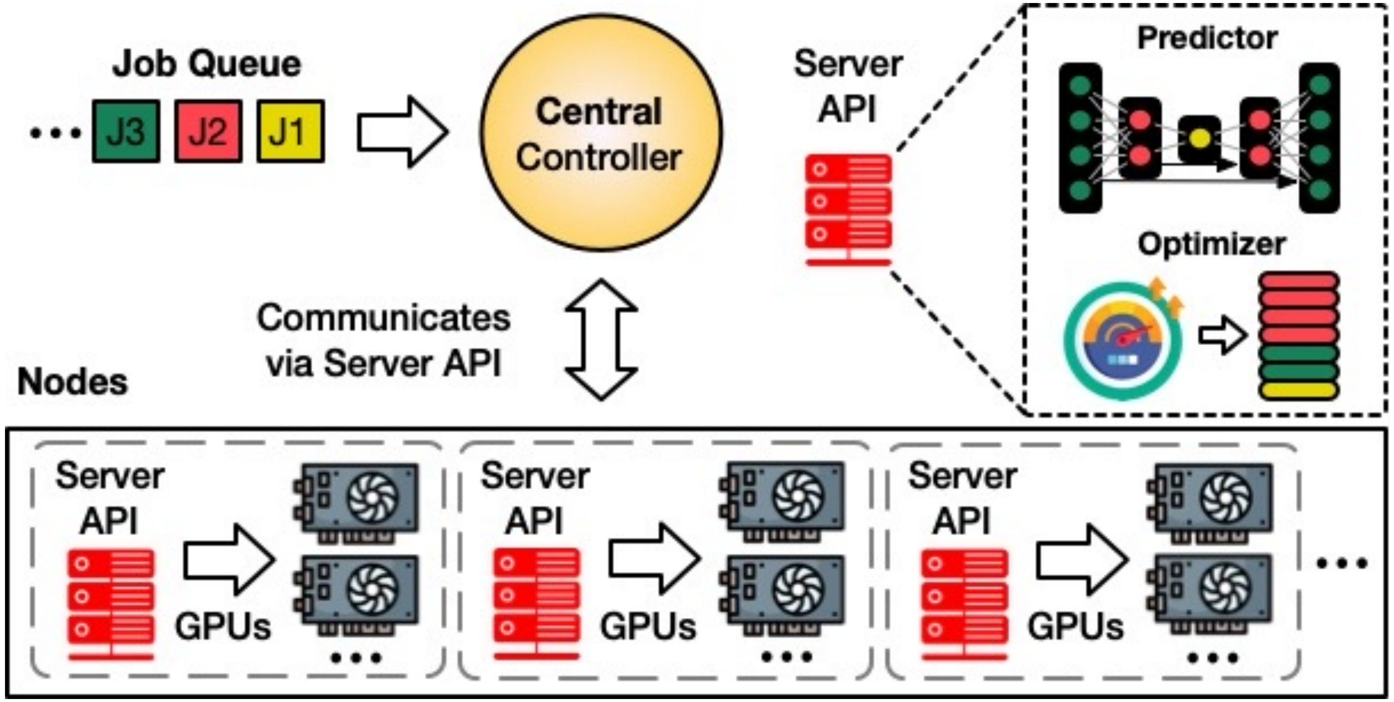
But profiling the performance speedup for all jobs on every MIG slice in the MIG mode causes prohibitive checkpoint-restart overhead, unlike the MPS-mode.

# MISO: Key Idea

MISO leverages the flexible but interference-prone MPS-based partitions to find the optimal MIG-based (interference-free) partitions to achieve higher performance for multi-tenant GPUs

MISO leverages best of the both the worlds (MPS and MIG): MPS for profiling and performance estimation, MIG for interference-free resource partitioning.

# Overview of the MISO Design




MISO uses lightweight MPS-mode run to quickly estimate jobs' performance on different MIG configurations using a machine learning model, and then partition the GPU resources intelligently.

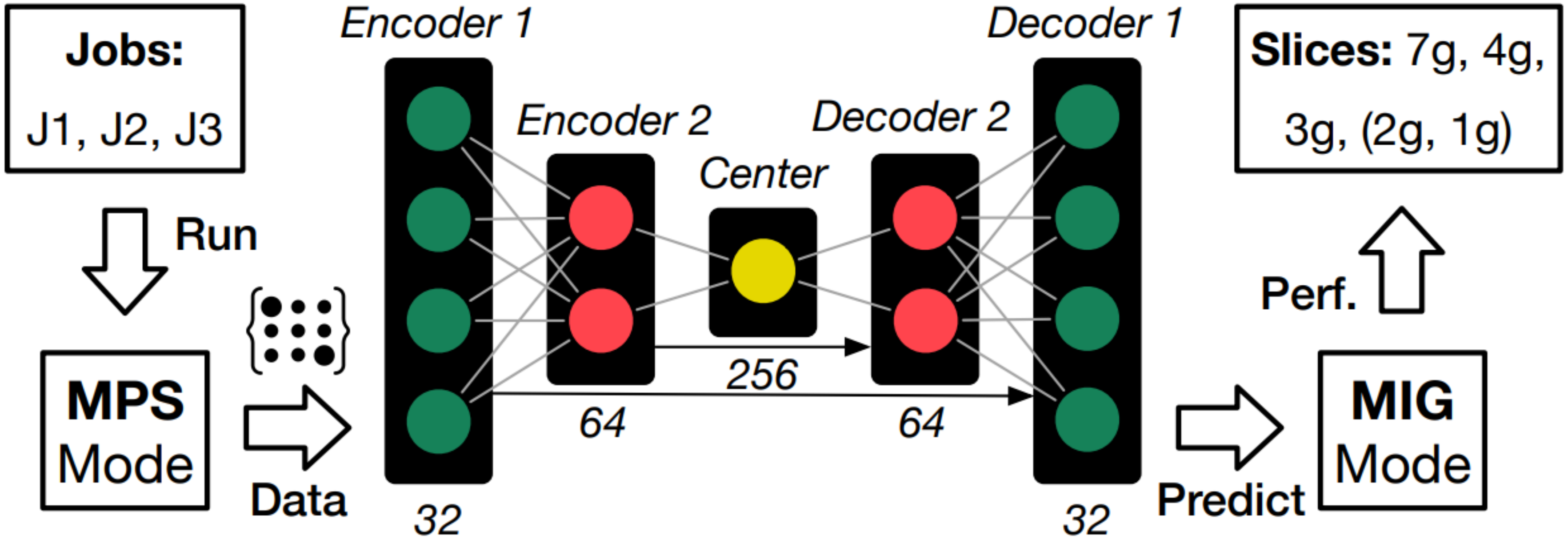
# MISO's Job MIG Performance Estimator Using MPS mode

Observation: Under MPS-mode, one can adjust GPU sharing levels for concurrent jobs in a job mix without frequently switching jobs in and out of the GPU.

MISO uses this flexibility to estimate performance on different MIG slices.

$$\begin{array}{cc} \text{MPS Level} & \begin{array}{cccc}\text{Job1} & \text{Job2} & \cdots & \text{Job7}\end{array} \\ \begin{array}{c}100\% \\ 50\% \\ 14\%\end{array} & \left\{\begin{array}{cccc} A_{11} & A_{12} & \cdots & A_{17} \\ A_{21} & A_{22} & \cdots & A_{27} \\ A_{31} & A_{32} & \cdots & A_{37} \end{array}\right\} \end{array} \xrightarrow{\text{Translate}} \begin{array}{cc} \text{MIG Slice} & \begin{array}{cccc}\text{Job1} & \text{Job2} & \cdots & \text{Job7}\end{array} \\ \begin{array}{c}7g \\ 4g \\ 3g\end{array} & \left\{\begin{array}{cccc} B_{11} & B_{12} & \cdots & B_{17} \\ B_{21} & B_{22} & \cdots & B_{27} \\ B_{31} & B_{32} & \cdots & B_{37} \end{array}\right\} \end{array}$$

# MISO's Job MIG Performance Estimator




Train a U-Net variant to translate the MPS performance into MIG performance

The 2g and 1g MIG slices can be extrapolated from 7g, 4g, 3g MIG performance

# MISO's MIG Partition Optimizer


**Algorithm 1:** MISO's partition optimizer.

$best\_obj \leftarrow 0$ `// Maximum objective so far`
$best\_config \leftarrow None$ `// Best partition so far`
$P_{valid} \leftarrow$ list of $P_{mig}$ partitions whose length equals $m$
**foreach** $\vec{x}$ *in* $P_{valid}$ **do**
  $obj\_func \leftarrow \sum_{i=1}^{m} f_i(x_i)$
  **if** $obj\_func > best\_obj$ **then**
    $best\_obj \leftarrow obj\_func$
    $best\_config \leftarrow \vec{x}$
  **end**
**end**
**return** $best\_config$

MISO quickly finds the optimal MIG partition without heuristics

Focuses on optimizing each GPU locally

- Avoids overhead from the global NP problem
- Avoids extra job checkpointing between GPU nodes

# MISO: Evaluation and Insights

# Experimental Methodology

## Setup

- 4-node system
- 2 AMD EPYC 7542 CPUs each node
- 2 NVIDIA A100 GPUs each node

for $m$ jobs $J_1$ to $J_m$, suppose job $J_i$ 's execution speed on an A100 GPU without co-location is $p_i$ , and its current execution speed is $q_i$

## Metrics

- Average Job Completion Time (JCT)
- Makespan
- System Throughput

$$\text{System Throughput (STP)} = \sum_{i=1}^{m} \frac{q_i}{p_i}$$

## Workloads

- Helios Trace [6] (SC'21)
- Poisson distributed arrival
- Deep learning workloads including BERT, GNN, CycleGAN.

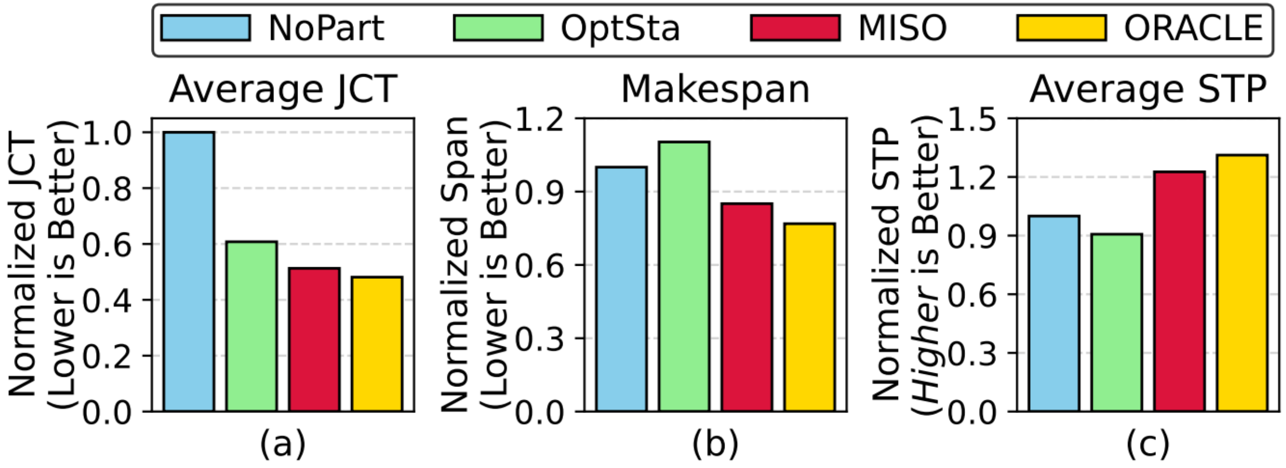
[3] Hu, Q., Sun, P., Yan, S., Wen, Y. and Zhang, T., 2021, November. Characterization and prediction of deep learning workloads in large-scale gpu datacenters. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (pp. 1-15).

## Schemes

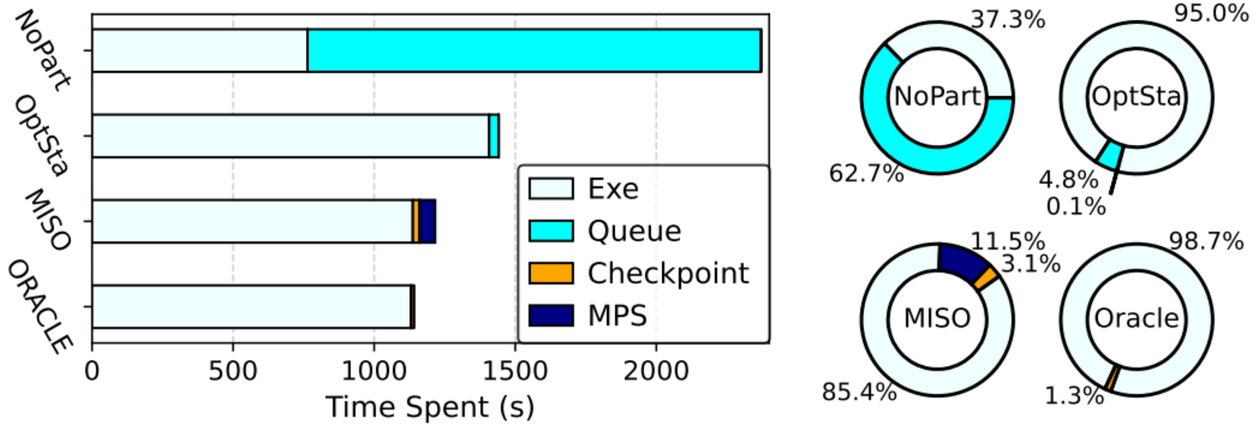
- NoPart: no partition
- OptSta: optimal static MIG partitioning
- ORACLE: knows MIG speedup for every job

# MISO offers significant improvements

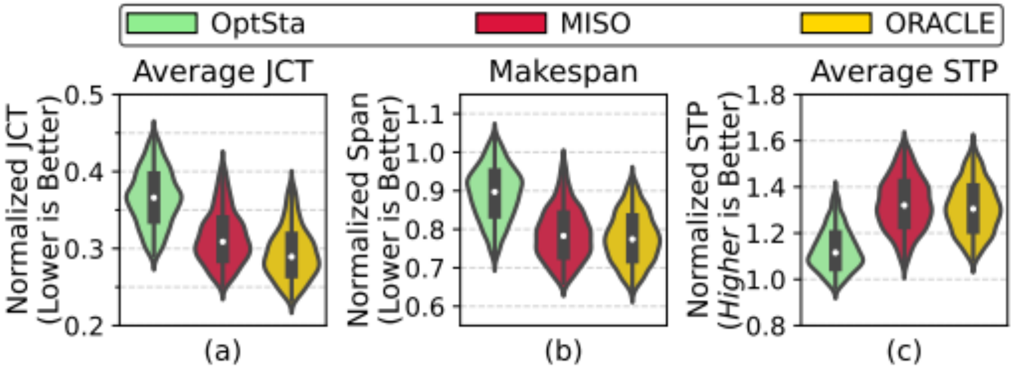
**Over 30% improvement in job completion time, makespan and system throughput**



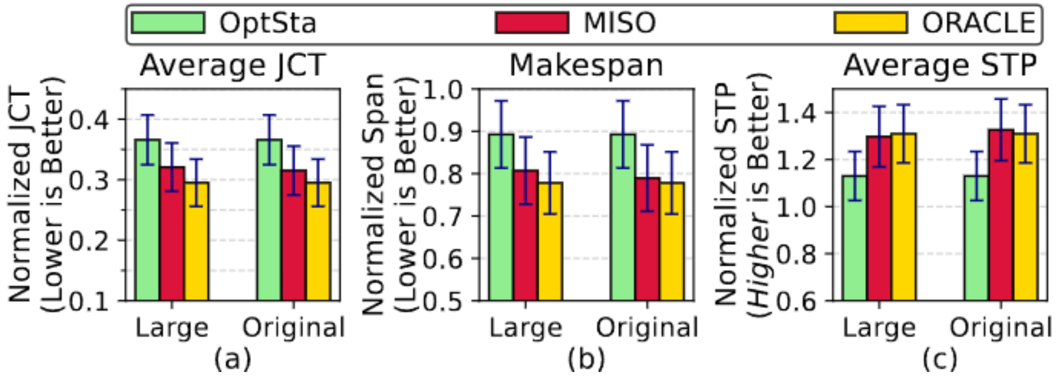

**Where does MISO performance improvements come from?**



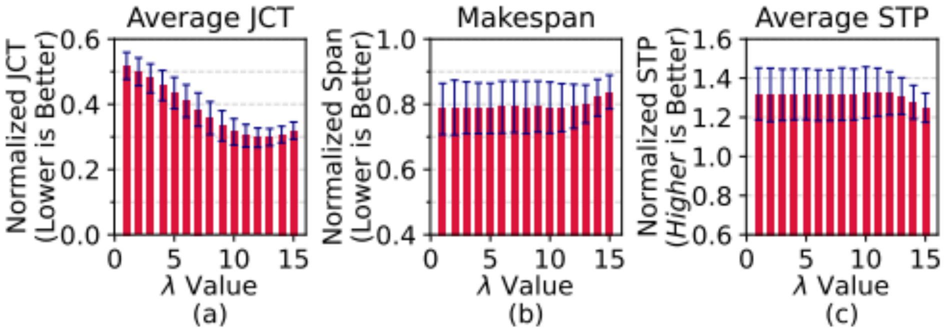
# MISO outperforms across different scenarios




Robust to different initial conditions



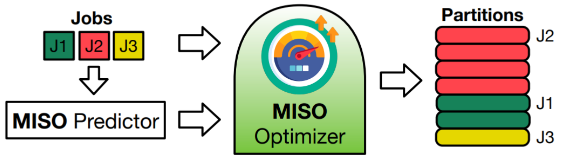

Robust to model prediction errors




Robust to different job-arrival rates

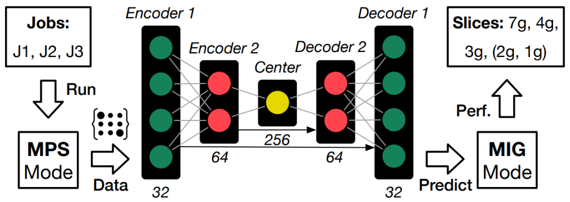
# MISO Summary of Key Contributions

MISO is the first method for GPU resource partitioning on a MIG-enabled multi-tenant GPU cluster.




MISO combines the best of both worlds (MPS and MIG).

MISO uses the lightweight MPS profiling to quickly estimate the optimal MIG partition without the excessive overhead to profile each job's MIG slice performance.




MISO provides significant improvement over unpartitioned GPU cluster and close to oracle-partitioned GPU cluster.

**Contact**

Baolin Li

li.baol@northeastern.edu

This research was supported by NSF Award 1910601, NSF Award 2124897, and Northeastern University. This research was also sponsored by the United States Air Force Research Laboratory and the United States Air Force Artificial Intelligence Accelerator and was accomplished under Cooperative Agreement Number FA8750- 19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.