Synergy :
Looking Beyond GPUs for DNN Scheduling on Multi-Tenant Clusters

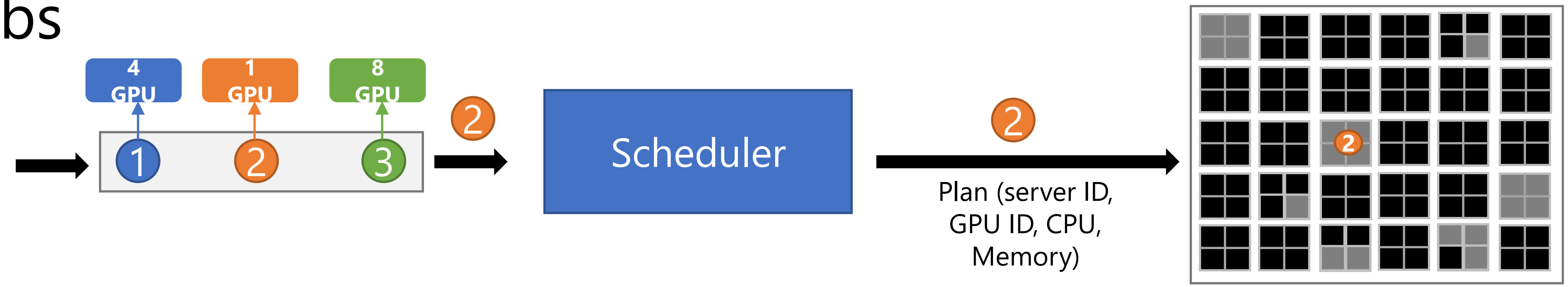
Jayashree Mohan*, Amar Phanishayee*, Janardhan Kulkarni*, Vijay Chidambaram^

*Microsoft Research ^UT Austin

Deep Learning at scale

- Large enterprises train DL jobs on large GPU clusters
  - Multi-tenant : Cluster shared between several users/product groups
  - Variety of training jobs – speech, image, NLP, etc.

- A cluster manager **allocates resources** and **schedules** training jobs

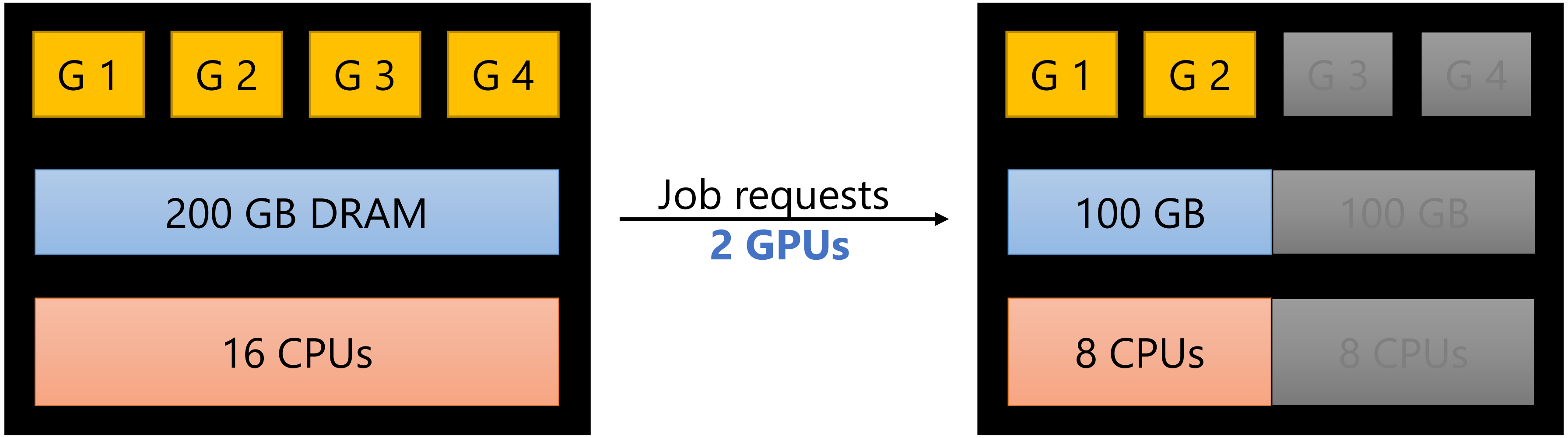


Server

- Scheduling Policy : FIFO, SRTF, LAS, FTF etc
- Cluster metrics : Job completion time (JCT), fairness, makespan, etc

GPU-Proportional Allocation

- Job specifies only GPU demand
- Auxiliary resources (CPU, memory) are allocated proportional to the GPUs requested
- Uses GPU proportional allocation

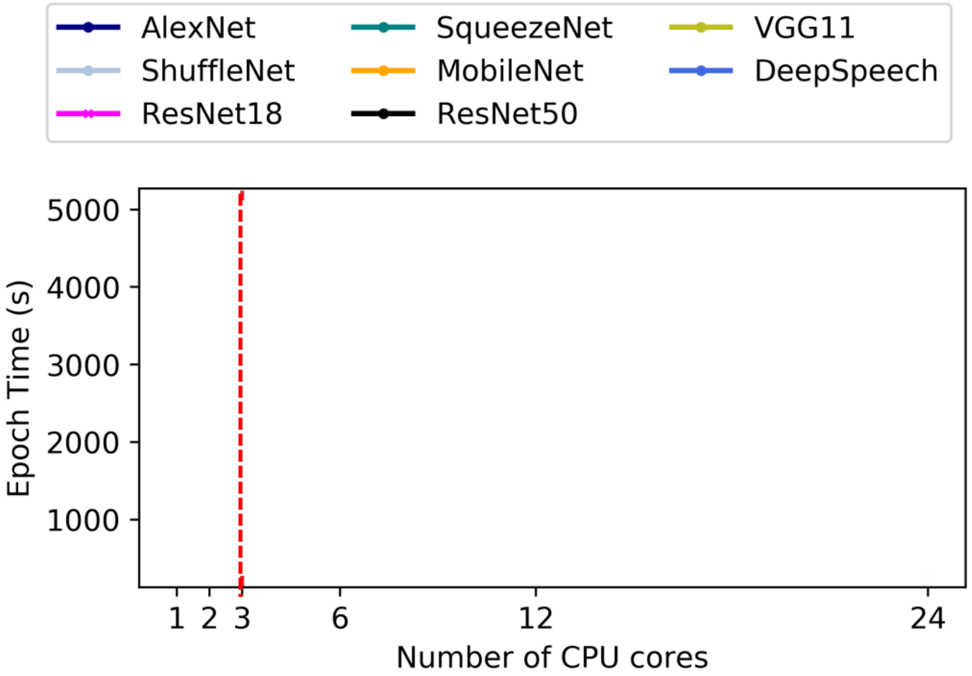


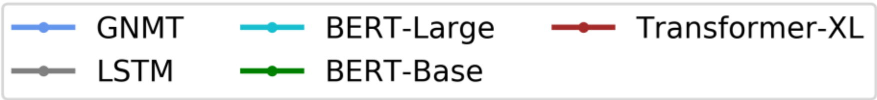
Motivation

- DNNs exhibit **varying levels of sensitivity** to CPU, DRAM allocation

Motivation

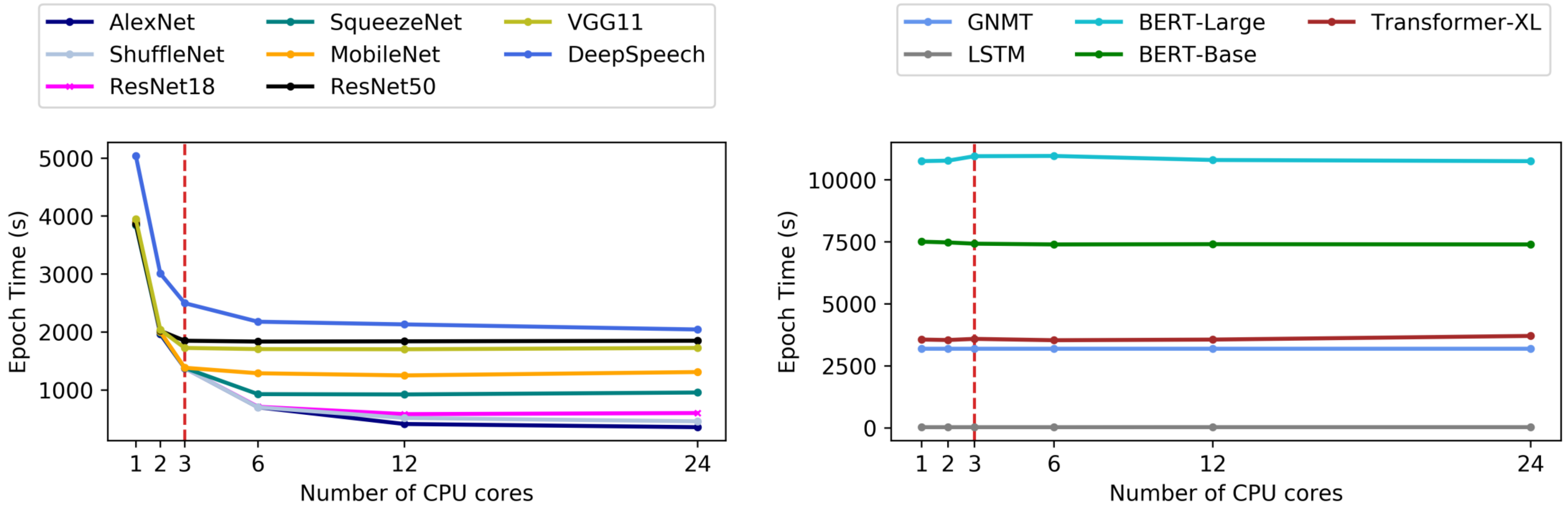
- DNNs exhibit **varying levels of sensitivity** to CPU, DRAM allocation





Motivation

- DNNs exhibit **varying levels of sensitivity** to CPU, DRAM allocation



Exploit the difference in resource requirement across jobs to perform **disproportionate resource allocation**

Challenges

What is the ideal aux. resource requirement for each job ?

Efficiently using resources in DNN scheduling

How to pack these jobs with malleable aux. resource demands on a cluster?

To address these challenges in a scheduling policy agnostic manner, we build Synergy

Synergy

- **Resource-sensitivity** aware scheduler for DNN training jobs
- Identifies each job's best-case CPU and memory requirements using an **optimistic profiling** technique.
- Packs these jobs on to the available servers along multiple resource dimensions using a close-to-optimal **heuristic scheduling mechanism**
- Improves cluster objectives by upto **3.4x** when compared to traditional GPU-proportional scheduling mechanism.

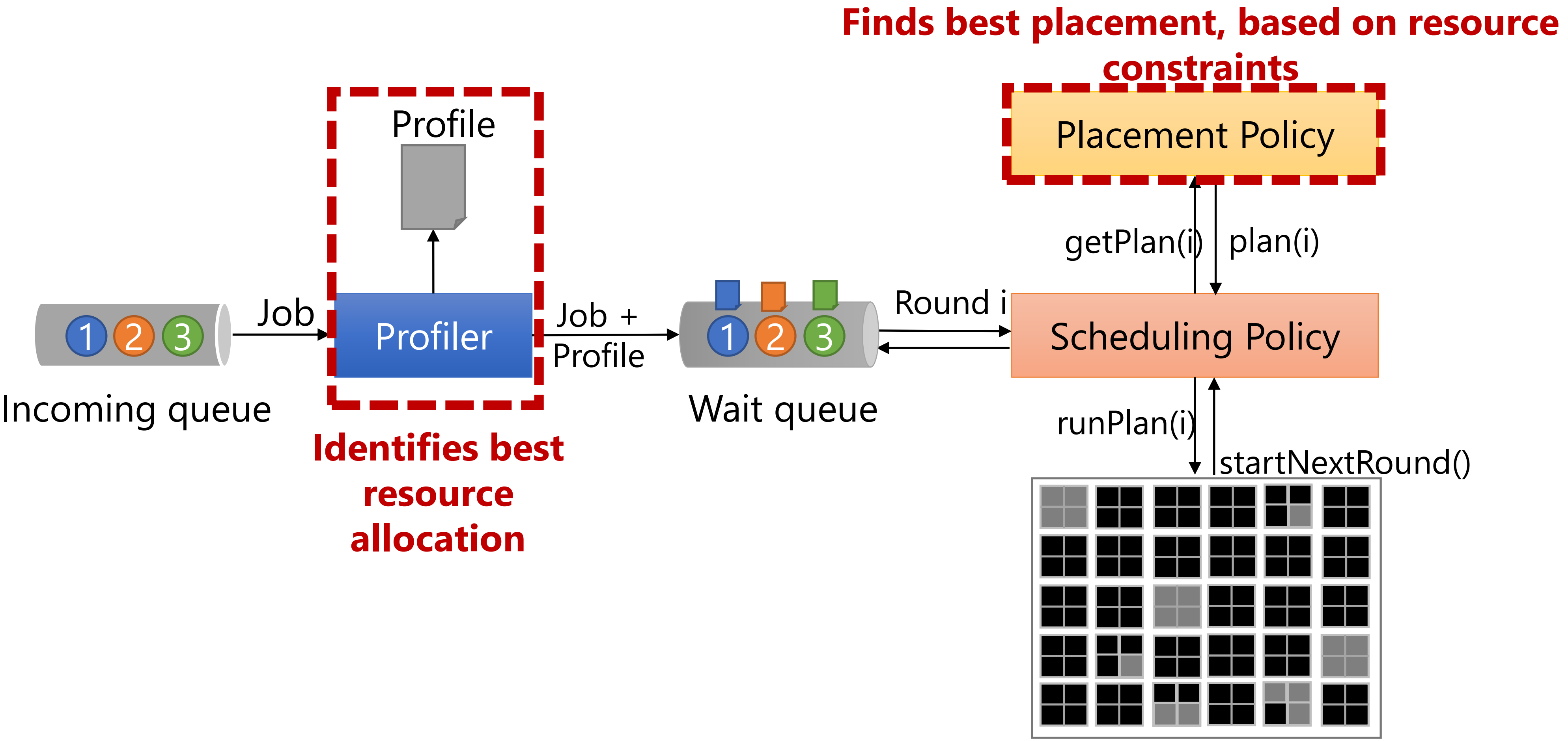
Outline

Synergy

- Motivation
- **Design**
  - Profiling
  - Placement
- Evaluation

Synergy : Design

- Round-based scheduling



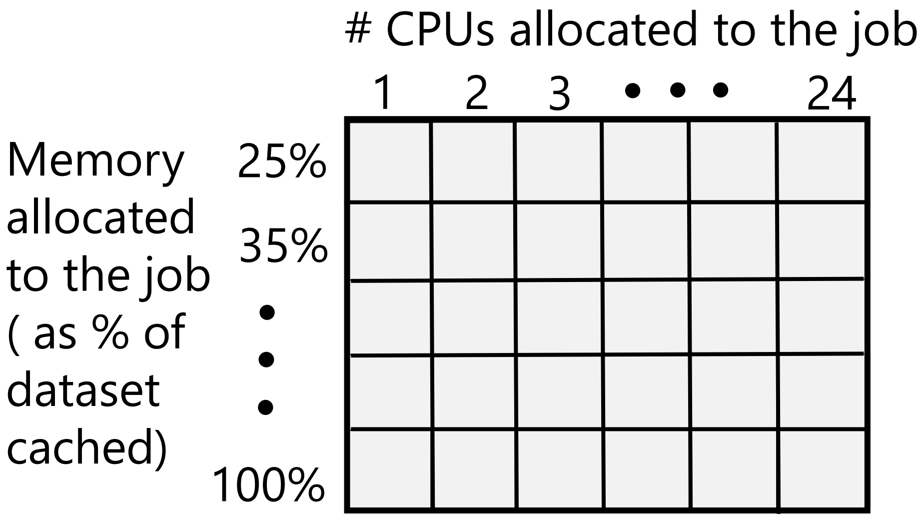
Outline

Synergy

- Motivation
- Design
  - **Profiling**
  - Placement
- Evaluation

Profiling

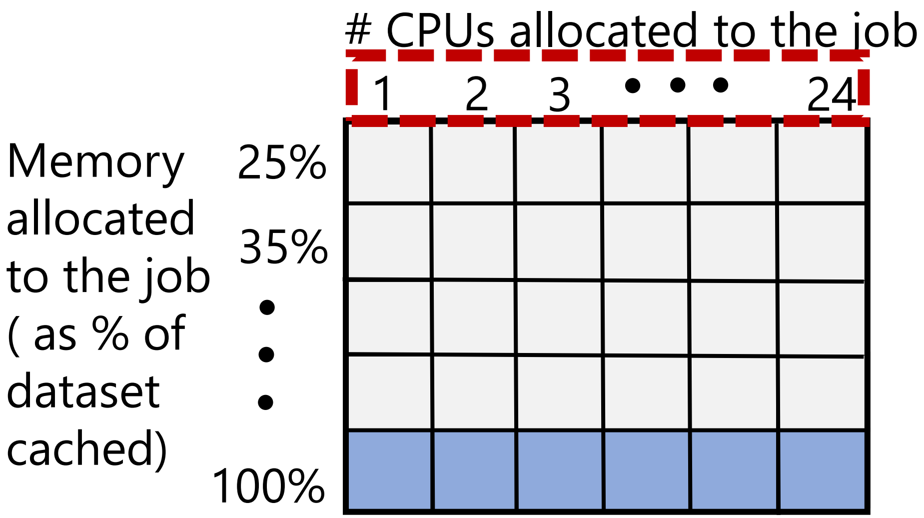
- Runs offline
- Assume there is a dedicated server(identical to the ones in cluster) for profiling that measures the sensitivity of incoming job to CPU, memory and data locality



To fill each point, need to run the training job for a few iterations

Profiling

- Runs offline
- Assume there is a dedicated server(identical to the ones in cluster) for profiling that measures the sensitivity of incoming job to CPU, memory and data locality



Estimated

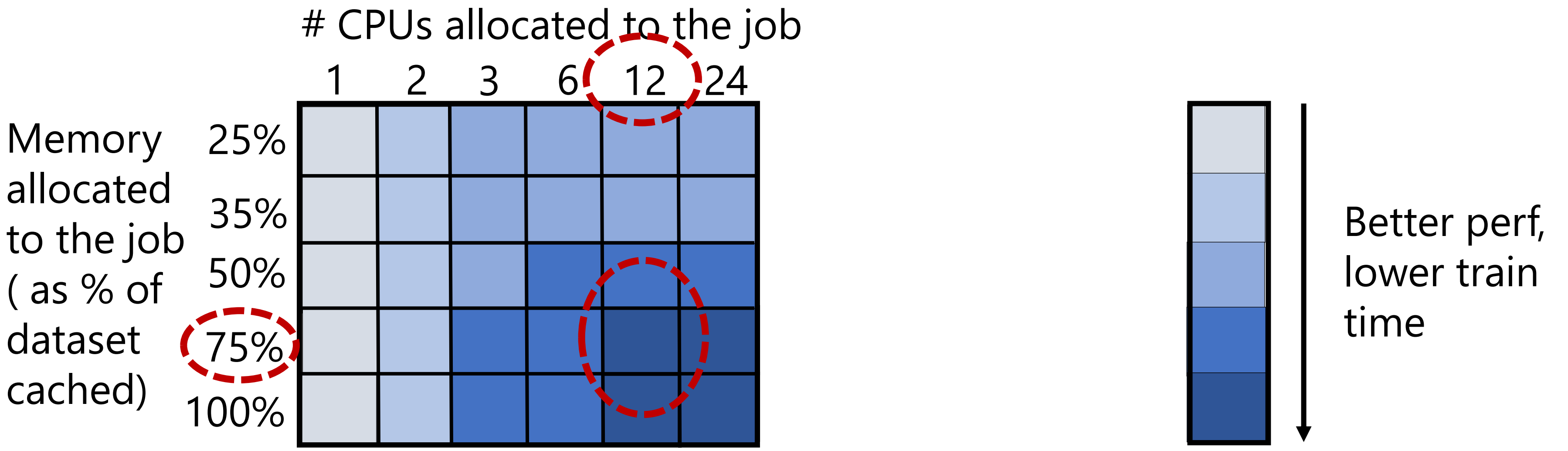
Empirical

- Measure only the CPU sensitivity
- Model memory sensitivity based on
  - cache size
  - memory bandwidth
  - storage bandwidth

- **Use MinIO** [VLDB'21]
- **Predictable per-epoch I/O**

To fill each point, need to run the training job for a few iterations

Profiling



- **CPU, DRAM allocation :** To find ideal resource allocation, find the least (CPU+mem) that reaches max performance

Outline

Synergy

- Motivation
- Design
  - Profiling
  - **Placement**
- Evaluation

Job Placement

- What is the best placement for a set of jobs in the given round?
- Multi-dimensional bin packing – NP hard

Job Placement

- What is the best placement for a set of jobs in the given round?
- Multi-dimensional bin packing – NP hard

Optimal

- Optimal allocation that provides an upper bound on the achievable cluster throughput
- Formulate our problem as a linear program (LP)
- 2 levels of LP :
  1. Idealized setting : All resources present in one (super) machine
     - Given profile matrix, find the allocation that maximizes overall throughput
  2. Construct a feasible allocation across available machines.

- Solving two LPs per scheduling round is a computationally expensive task.
- Final allocation matrix can have fractional GPU allocations

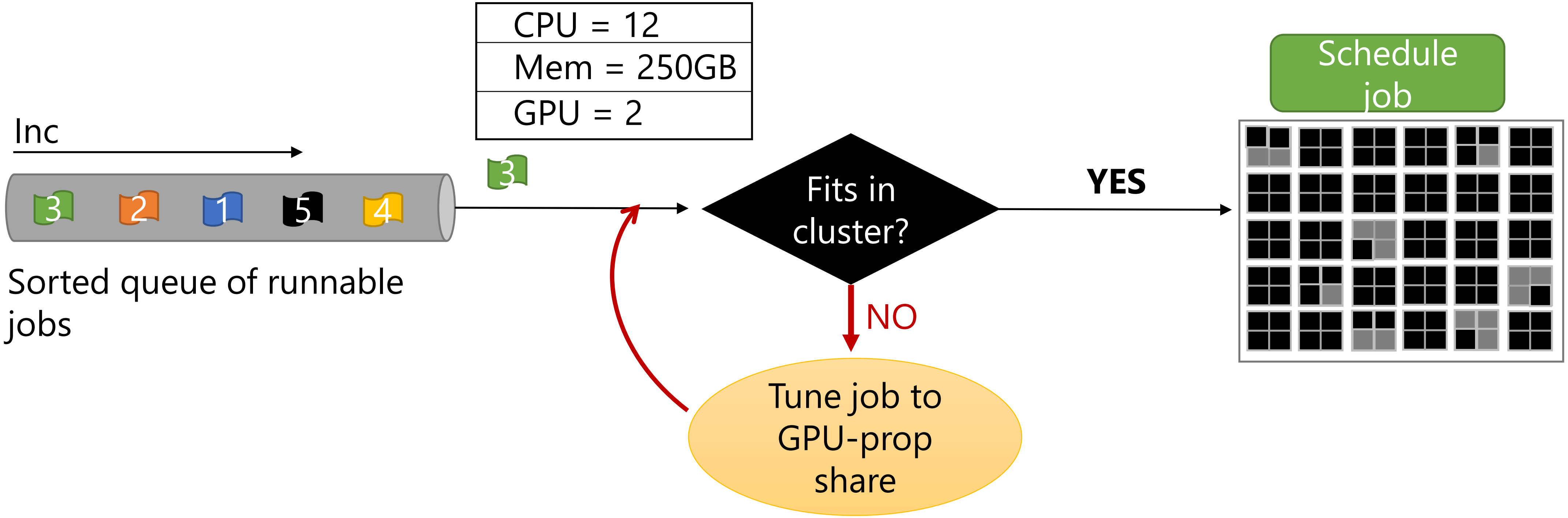
Job Placement

- What is the best placement for a set of jobs in the given round?
- Multi-dimensional bin packing – NP hard

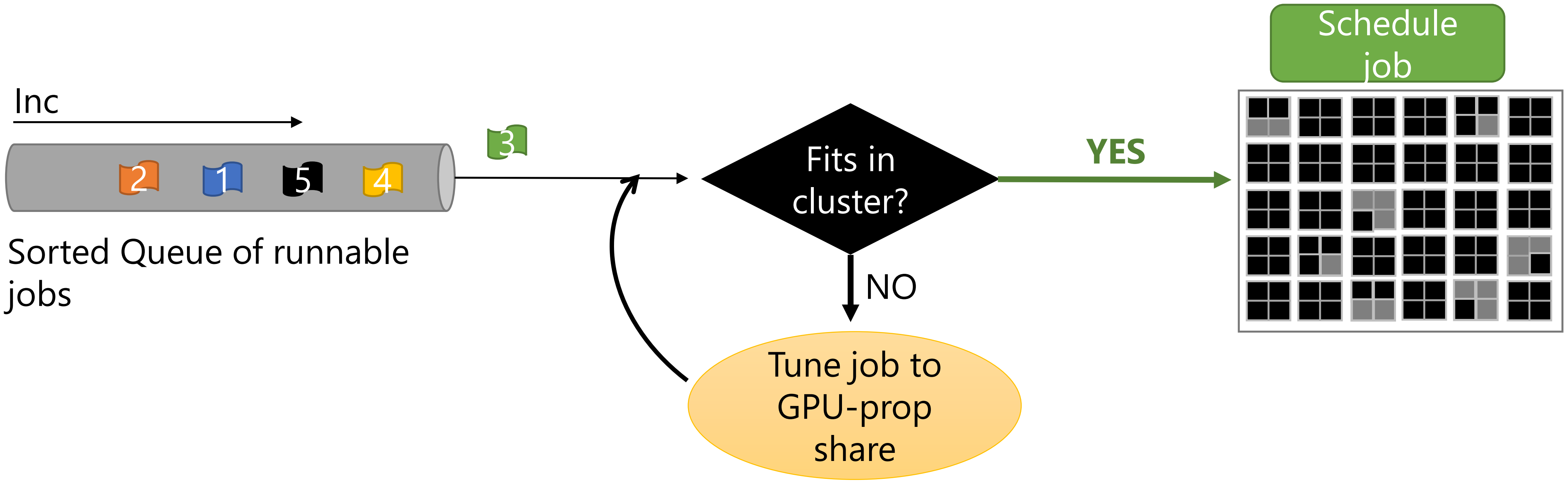
Optimal

Synergy-Tune

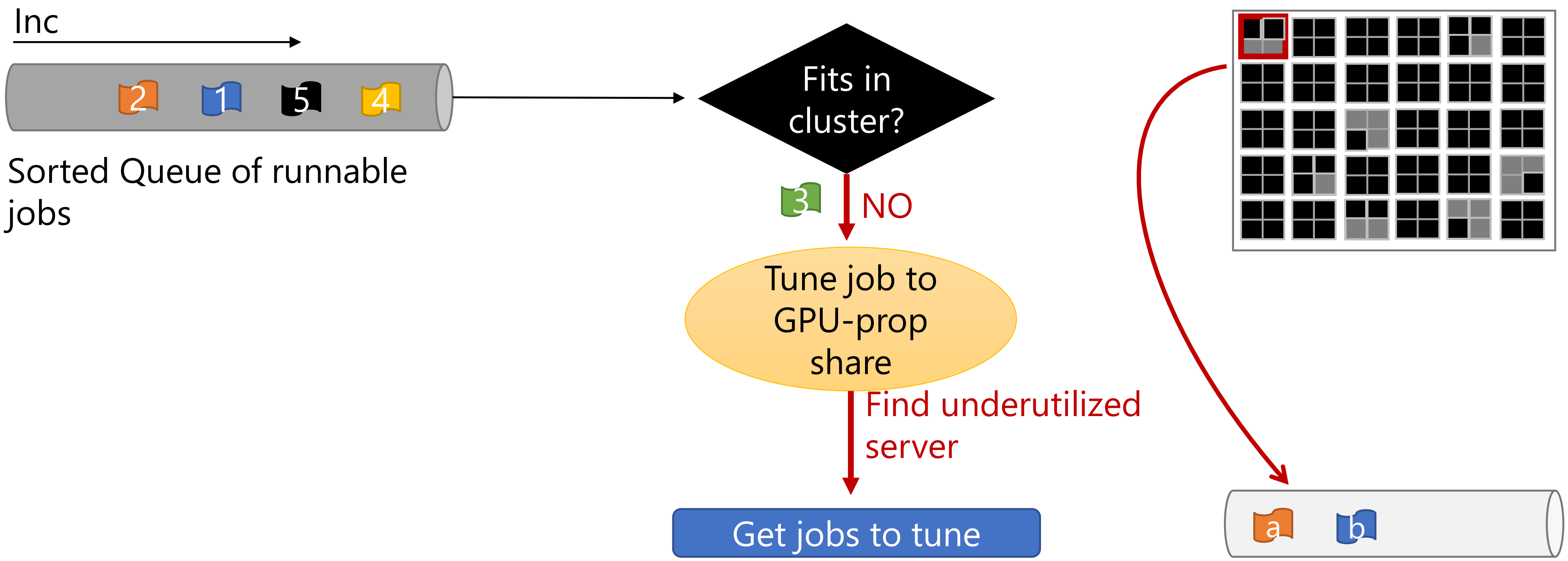
Scheduling with multiple resource demands



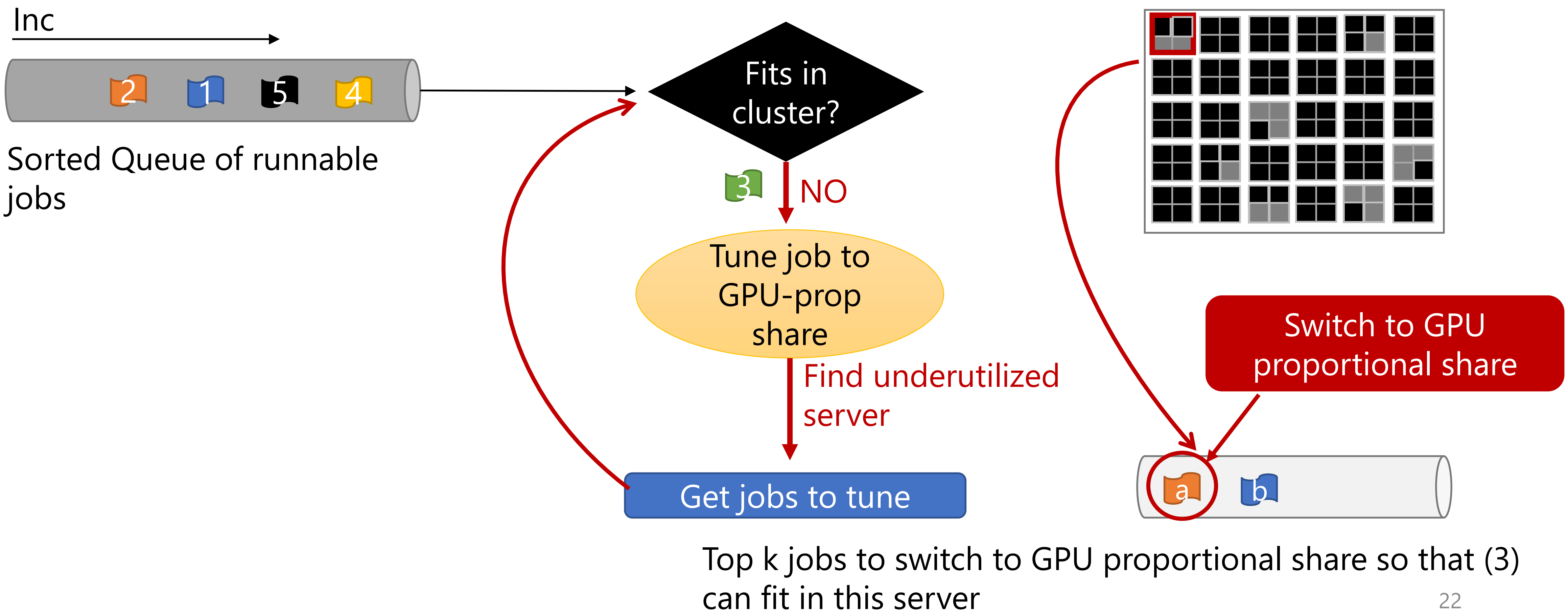
Scheduling with multiple resource demands



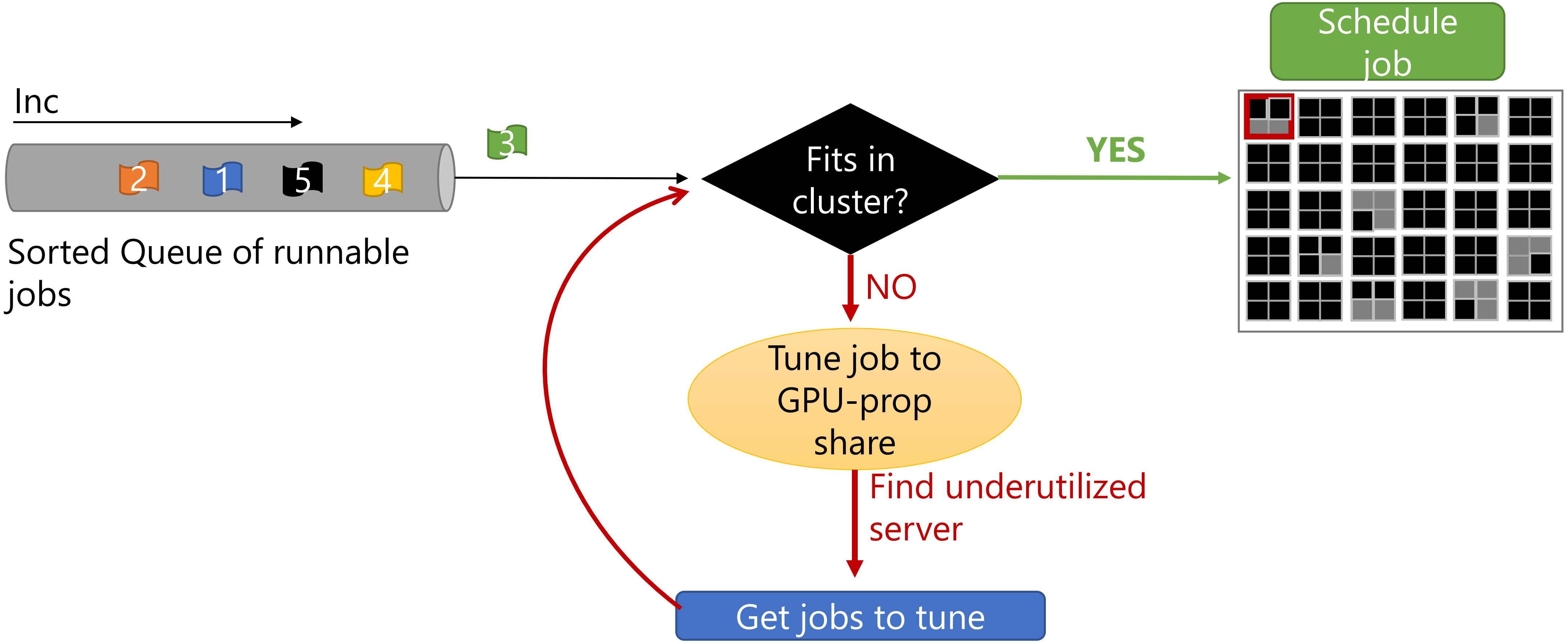
Scheduling with multiple resource demands



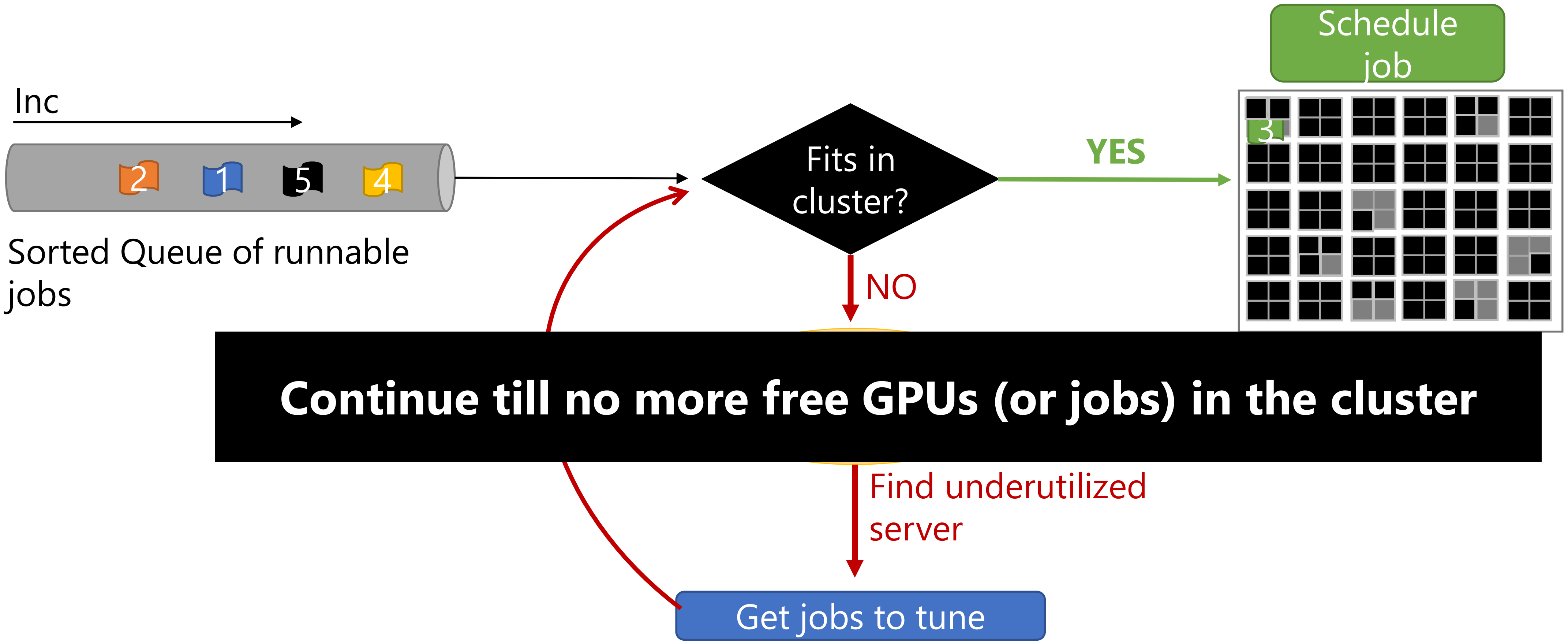
Scheduling with multiple resource demands



Scheduling with multiple resource demands



Scheduling with multiple resource demands



Outline

Synergy

- Motivation
- Design
  - Profiling
  - Placement
- **Evaluation**

Evaluation

- Experiments are performed on servers from an internal GPU cluster at Microsoft
  - Each server has eight V100 GPUs (32GiB), 24 CPU cores, 500 GB DRAM, and SSD storage

- Consider 10 different models (CNNs, RNNs, LSTMs) aross different tasks

Image	Language	Audio
AlexNet, ResNet18, ShuffleNet,MobileNet, ResNet50	GNMT, LSTM, Transformer-XL	DeepSpeech, M5

Microbenchmarks

Trace-driven simulations

Physical cluster deployment

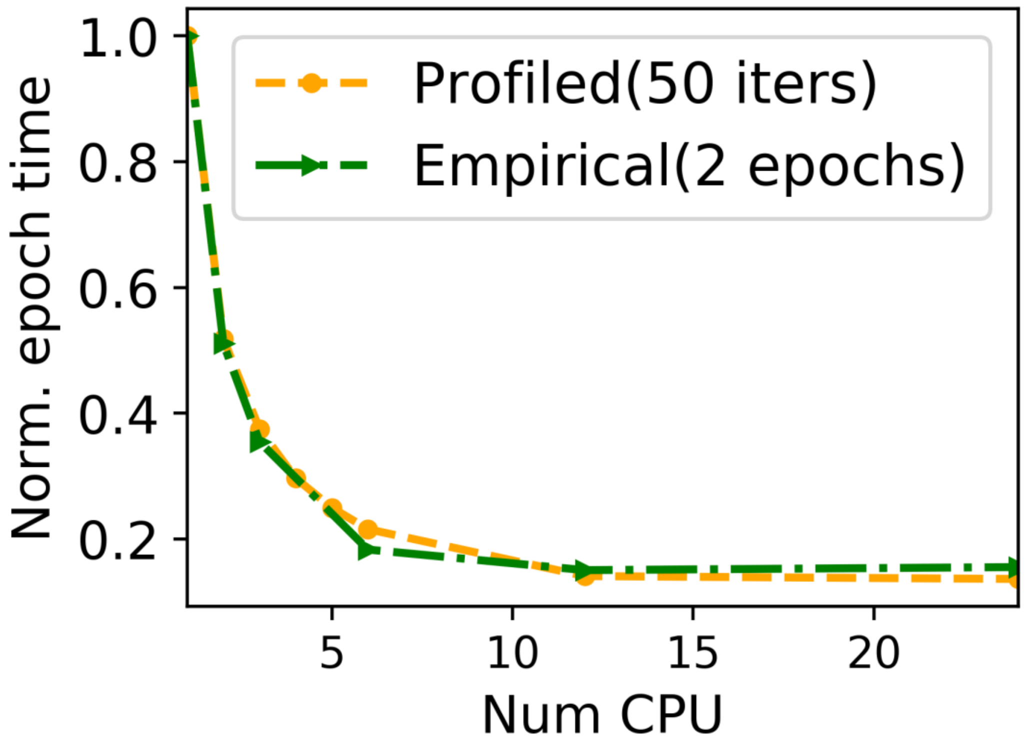
Evaluation Questions

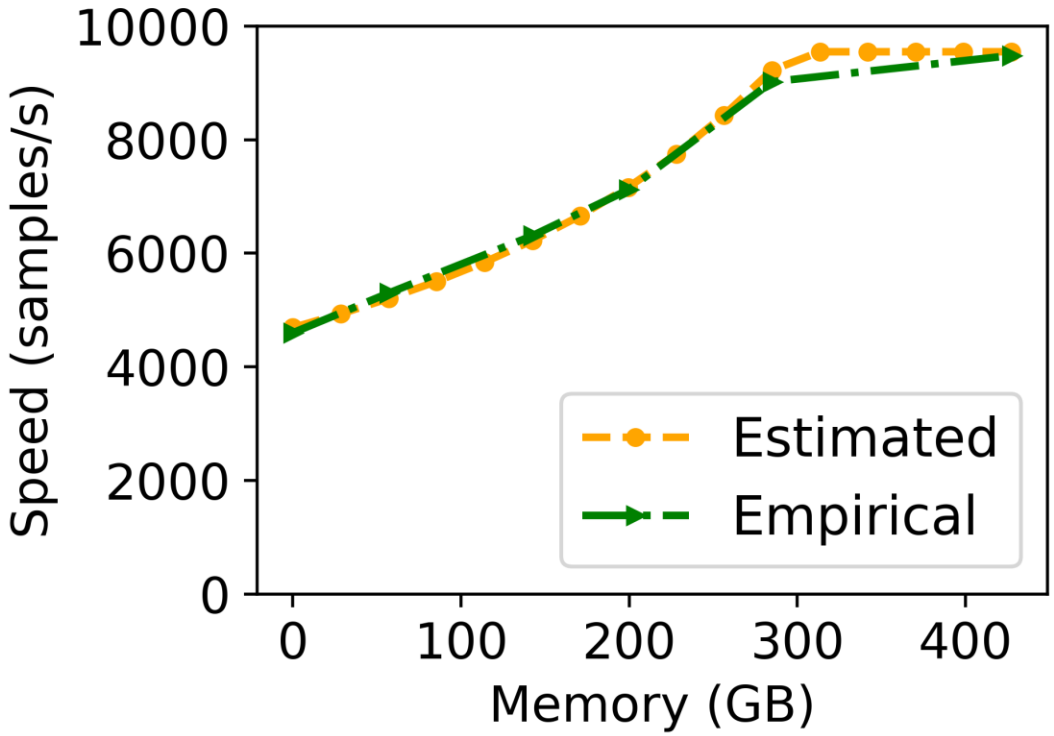
- Can Synergy’s optimistic profiling replicate real trends in job throughput?
- Can Synergy’s scheduling mechanism improve overall cluster metrics like Makespan and average JCT?
- How does Synergy’s heuristic tuning mechanism compare to optimal solution?
- How does Synergy react to varying workload compositions?
- How well does Synergy’s scheduling mechanism scale to larger clusters?
- How does Synergy compare to multi-resource big data scheduling policies like DRF?

Evaluation Questions

- Can Synergy's optimistic profiling replicate real trends in job throughput?
- Can Synergy's scheduling mechanism improve overall cluster metrics like Makespan and average JCT?
- How does Synergy's heuristic tuning mechanism compare to optimal solution?
- How does Synergy react to varying workload compositions?
- How well does Synergy's scheduling mechanism scale to larger clusters?
- How does Synergy compare to multi-resource big data scheduling policies like DRF?

1. Optimistic Profiling





- Profiles CPU and memory demand for ResNet18

Optimistic profiling is able to replicate the CPU and memory demand curve within 3% of empirical results

2. Synergy improves cluster objectives

Physical cluster of 32 GPUs
across 4 machines

2. Synergy improves cluster objectives

Policy	Metric	Mechanism	Deploy (hrs)	Simulation (hrs)
FIFO	Makespan	Proportional	16	15.67
		Tune	11.6	11.33
		Opt	-	11.01

Physical cluster of 32 GPUs
across 4 machines

Synergy-Tune improves makespan by 1.4x

2. Synergy improves cluster objectives

Policy	Metric	Mechanism	Deploy (hrs)	Simulation (hrs)
FIFO	Makespan	Proportional	16	15.67
		Tune	11.6	11.33
		Opt	-	11.01
SRTF	Avg JCT	Proportional	4.81	4.52
		Tune	3.21	3.19
		Opt	-	3.06

Synergy-Tune improves makespan by 1.4x, average JCT by 1.5x

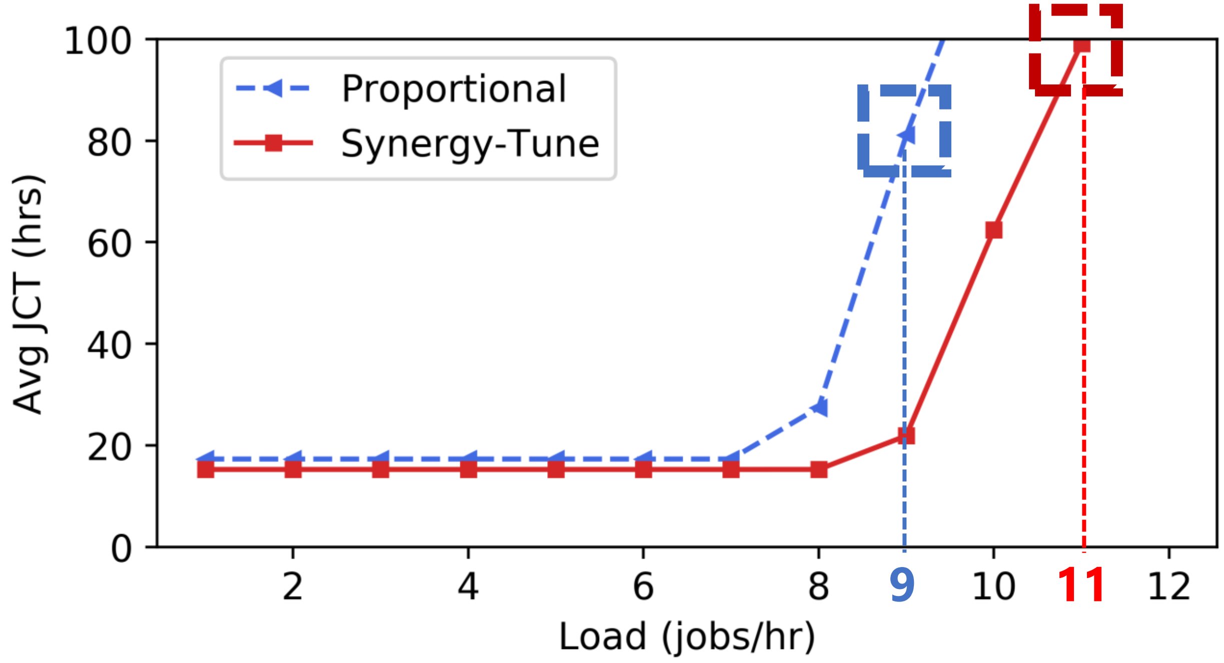
2. Synergy improves cluster objectives

Policy	Metric	Mechanism	Deploy (hrs)	Simulation (hrs)
FIFO	Makespan	Proportional	16	15.67
		Tune	11.6	11.33
		Opt	-	11.01
SRTF	Avg JCT	Proportional	4.81	4.52
		Tune	3.21	3.19
		Opt	-	3.06
	99^{th} p JCT	Proportional	17.32	16.85
		Tune	8.29	8.54
		Opt	-	8.21

Synergy-Tune improves makespan by 1.4x, average JCT by 1.5x and 99th percentile JCT by 2x

3. Synergy enables the cluster to support higher load

Simulated cluster of 128 GPUs



Synergy supports higher load by efficiently utilizing resources to finish jobs faster

Conclusion

- Resource-sensitivity aware scheduler for DNN training
- Exploits heterogeneity in auxiliary resource requirement to perform workload-aware allocation
- Improves cluster-wide performance

https://github.com/msr-fiddle/synergy

Thanks!

Questions?

Contact : jamohan@microsoft.com