MLaaS in the Wild

Workload Analysis and Scheduling in Large-Scale Heterogeneous GPU Clusters

Qizhen Weng[†*], Wencong Xiao[*], Yinghao Yu[*†], Wei Wang[†], Cheng Wang[*], Jian He[*], Yong Li[*], Liping Zhang[*], Wei Lin[*], Yu Ding[*]

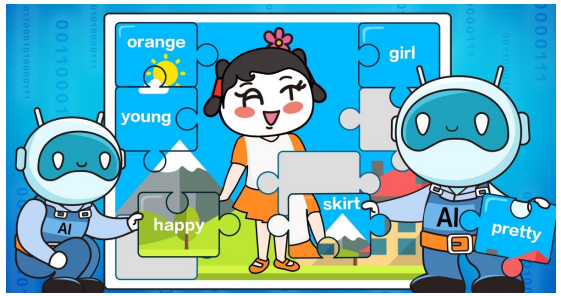
[†]Hong Kong University of Science and Technology [*]Alibaba Group

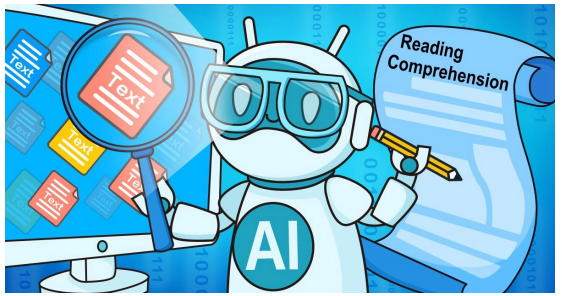
香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Alibaba Group
阿里巴巴集团

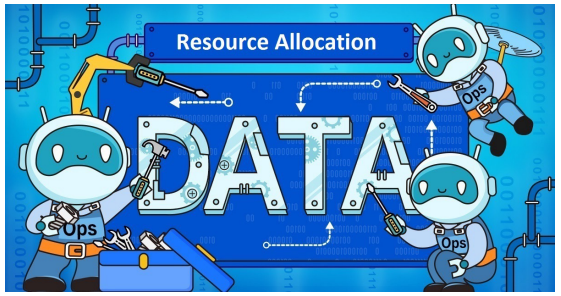
Production ML Workloads

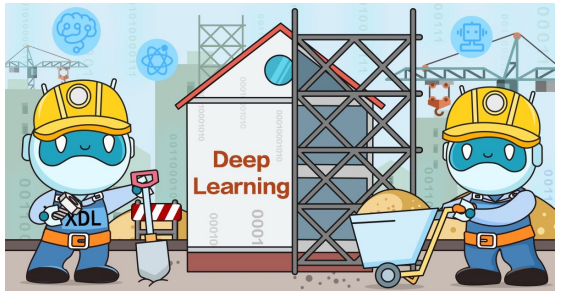
- Machine Learning (ML) is ubiquitous: CV, NLP, Recommend …





- ML jobs can scale to hundreds of GPUs, requesting TBs of data.

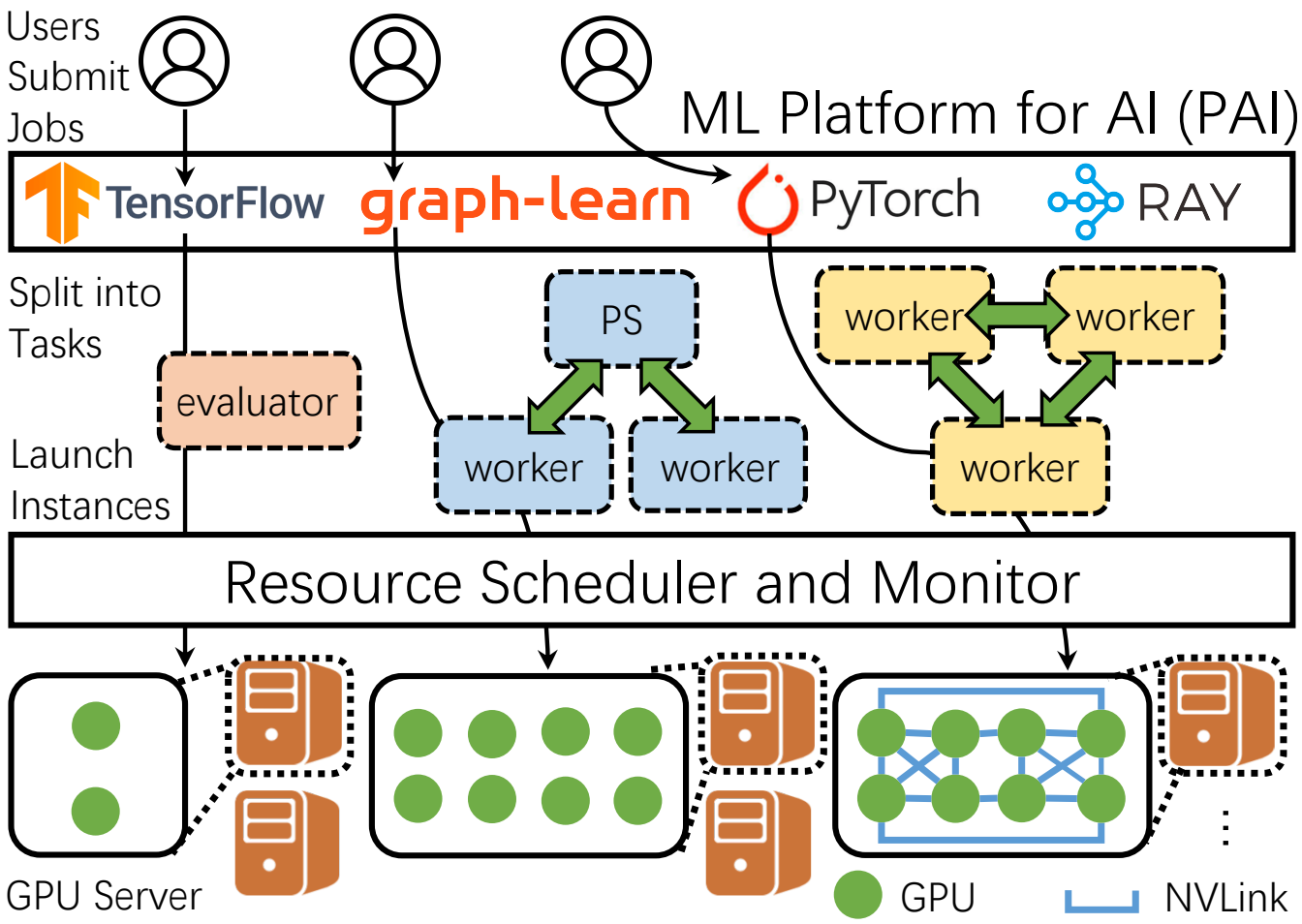




Images credit to: https://alibabatech.medium.com/

PAI: a Cloud-based MLaaS Platform

- All-in-one platform for training & inference
- Support a variety of ML frameworks
- Allocate containerized instances to hetero. nodes



Trace Basics

Traces are released at: https://github.com/alibaba/clusterdata/tree/master/cluster-trace-gpu-v2020

Trace Overview

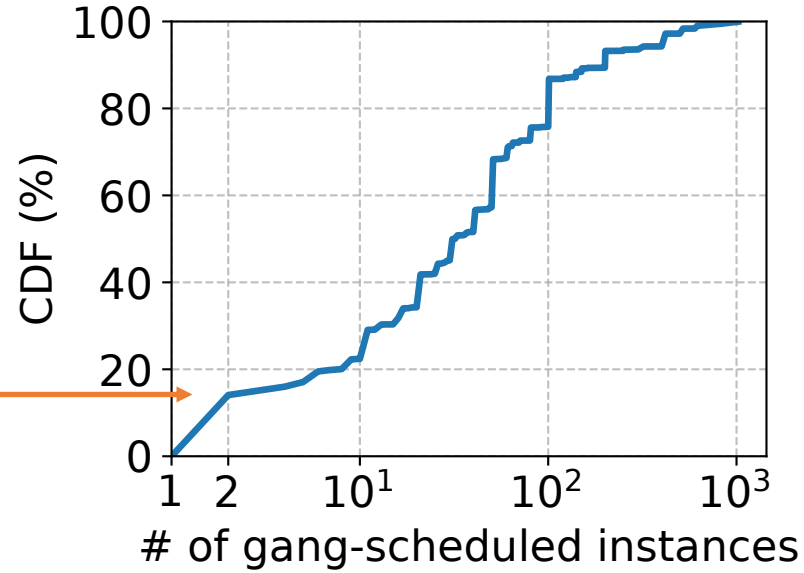
- Collected in Jul. and Aug. 2020 on 1800+ nodes, 6700+ GPUs
- Over 1300 users submitted 1.2M tasks with 7.5M instances
  - Allow GPU sharing & type specifying
  - 85% Instances are gang-scheduled

† are equipped with NVLink

System	#CPUs	Mem (GiB)	#GPUs	GPU type	#Nodes
PAI	64	512	2	P100	798
	96	512	2	T4	497
	96	512	8	Misc.	280
	96	384	8	V100M32†	135
	96	512/384	8	V100†	104
	96	512	0	N/A	83

A sample job

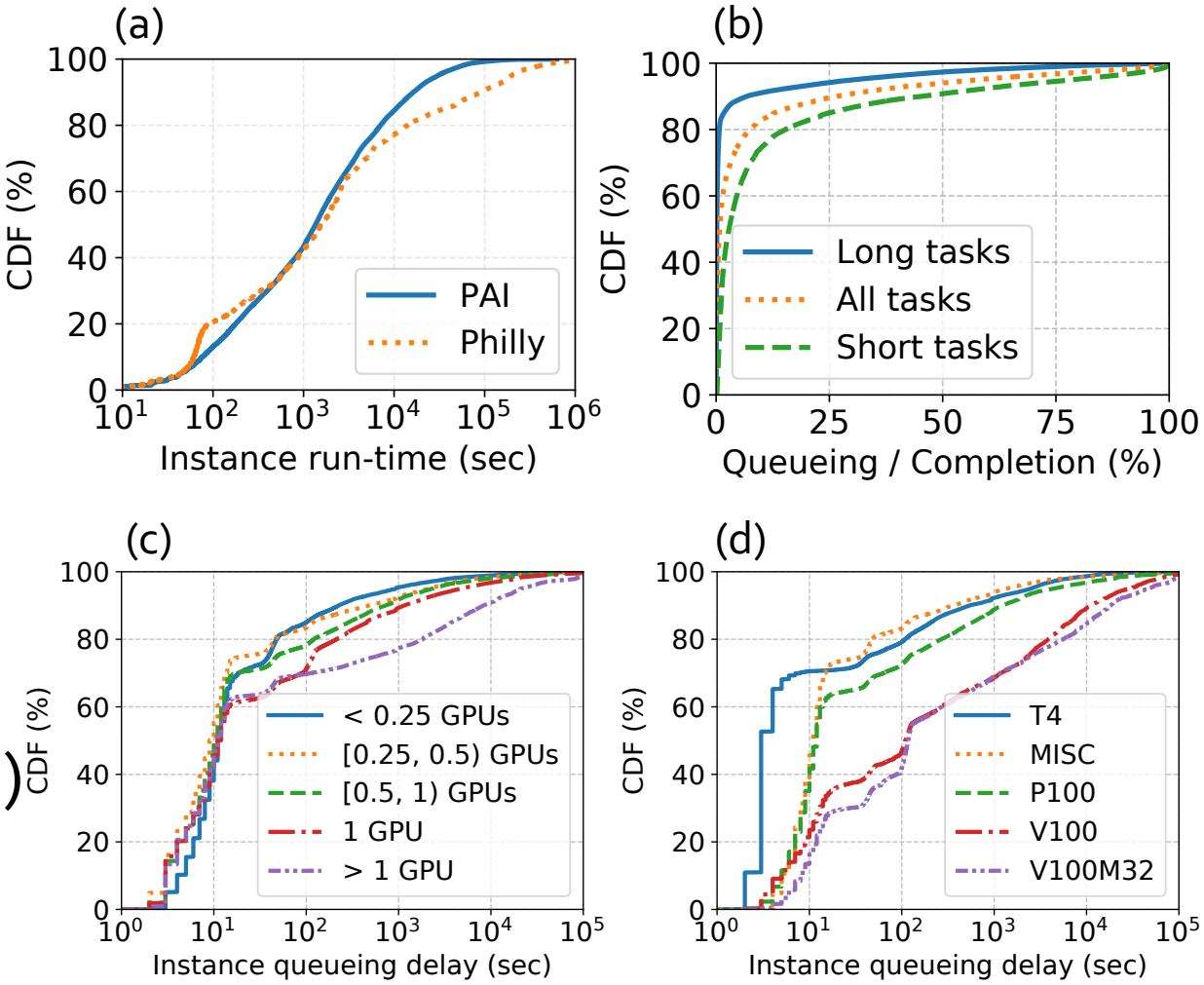
task_name	num_inst	plan_cpu	plan_mem	plan_gpu	gpu
worker	20	800	30000	100	T4
ps	25	1500	40000	N/A	N/A



Traces are released at: https://github.com/alibaba/clusterdata/tree/master/cluster-trace-gpu-v2020

Run-time and Queueing delays

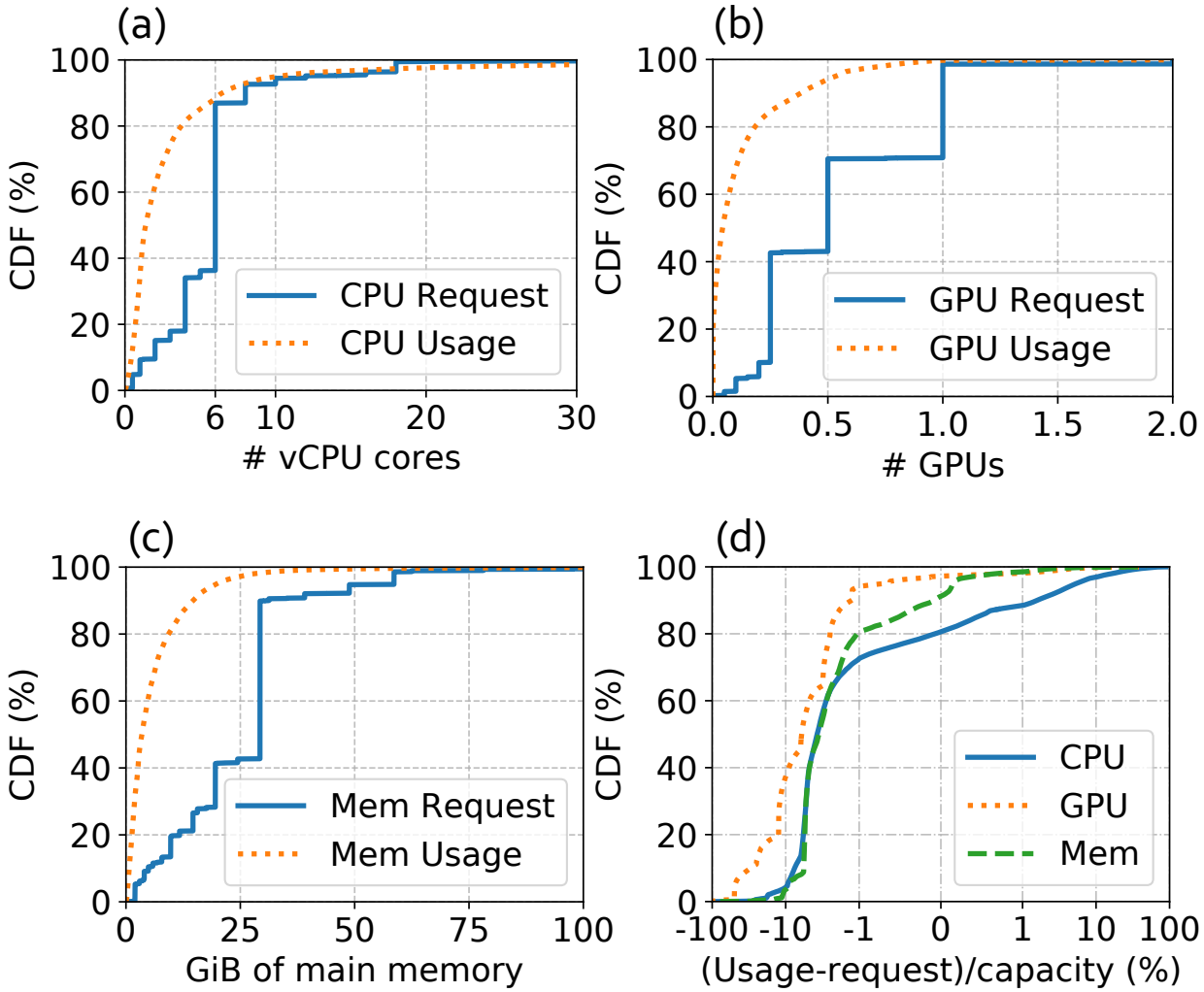
- Instance run-time ranges from seconds to hours (c.f. Philly[1]) (a)
- Short tasks wait for a larger proportion of their runtime (b)
- Tasks requesting full and/or high-end GPUs wait longer (c)(d)



[1] Jeon, Myeongjae et al. "Analysis of Large-Scale Multi-Tenant GPU Clusters for DNN Training Workloads." ATC '19 https://github.com/msr-fiddle/philly-traces

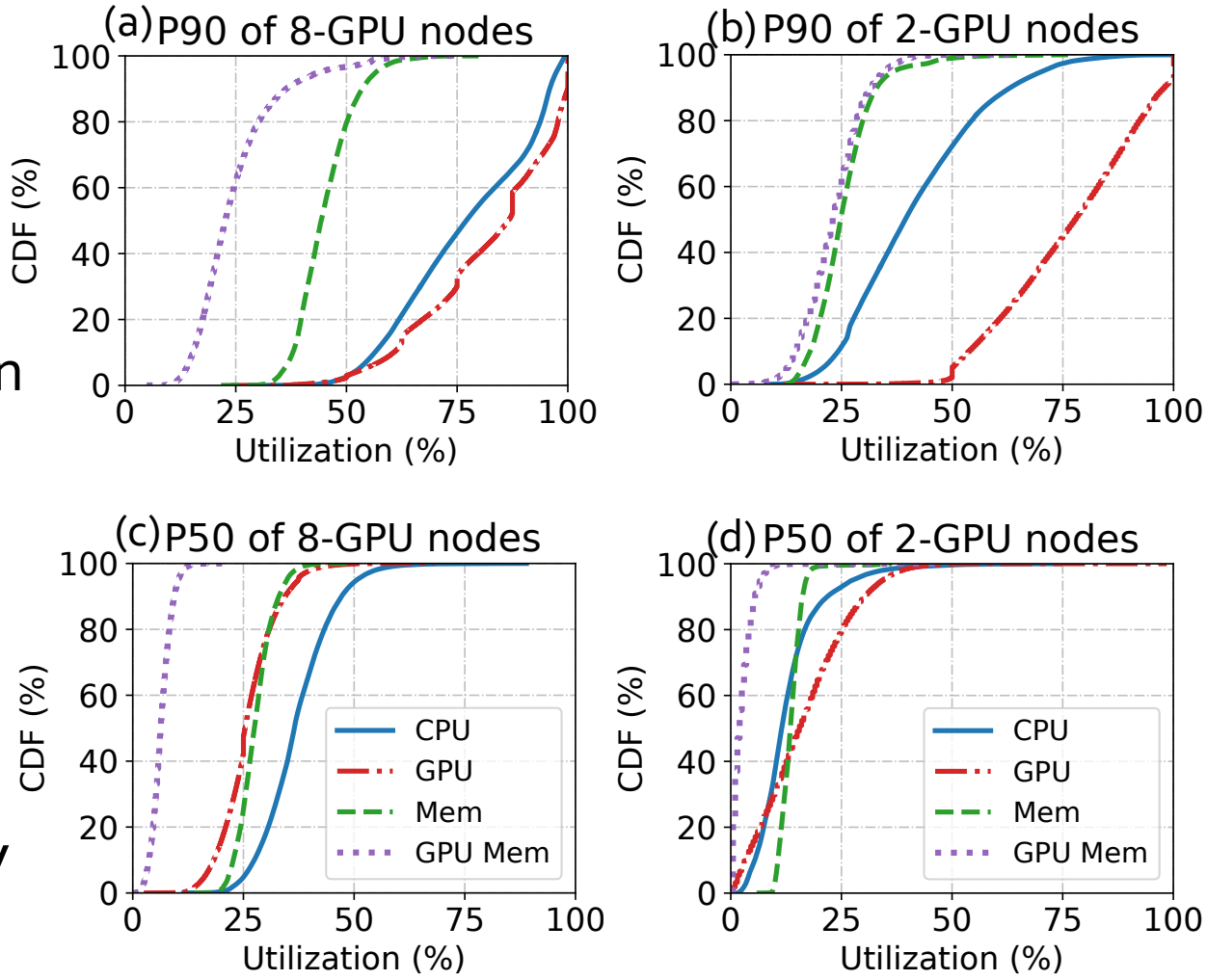
Resource Requests & Usage

- Heavy-tailed distribution of requests: ~20% are "elephants" (solid lines in (a) (b) (c))
- Uneven resource usage: High on CPUs but low on GPUs (dotted lines in (a) (b) (c))
- Usage / Request: 19% instances overuse CPUs (solid lines in (d)); only 3% instances overuse GPUs (dotted lines in (d)).



Machine Resource Utilization

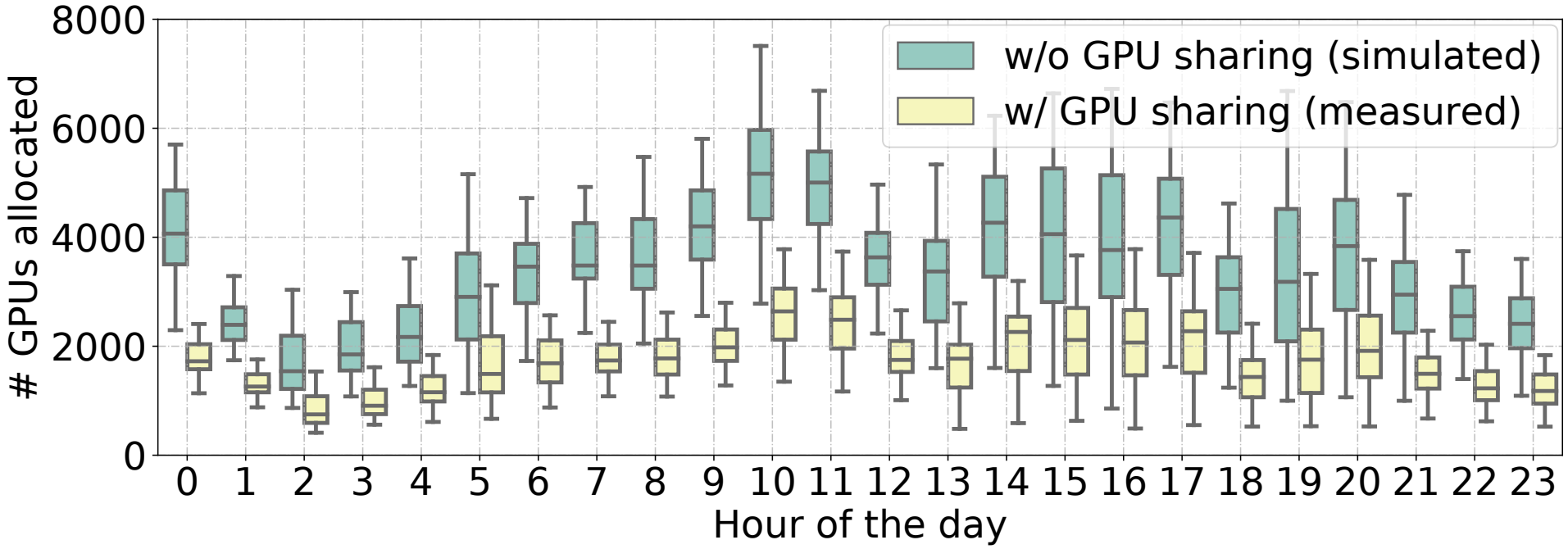
- 8-GPU machines (a) (c)
  - High GPU, high CPU utilization
- 2-GPU machines (b) (d)
  - High GPU, medium CPU utilization
- Gap between P90 and P50 (median) util. is large for GPUs
- Main memory and GPU memory are sufficient.



Opportunities we found

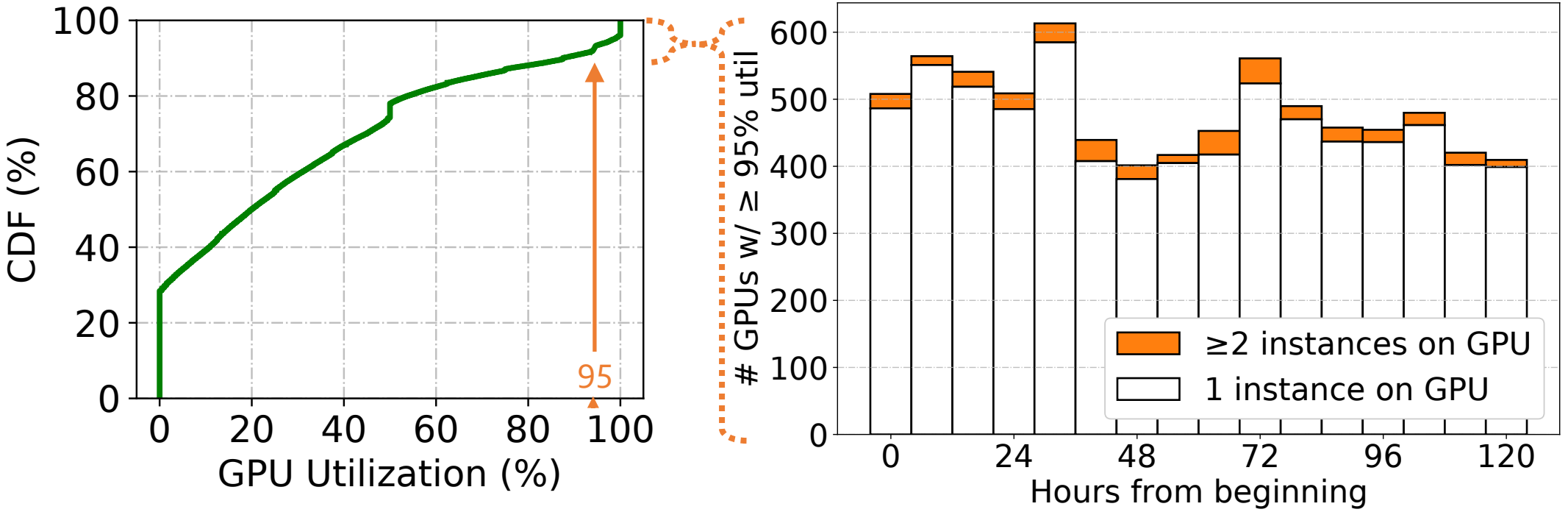
GPU Sharing

- Space-multiplex: GPU memory. Time-multiplex: GPU compute units.
- Sharing saves 50% GPUs on avg. and 2,500 GPUs in peak hrs.



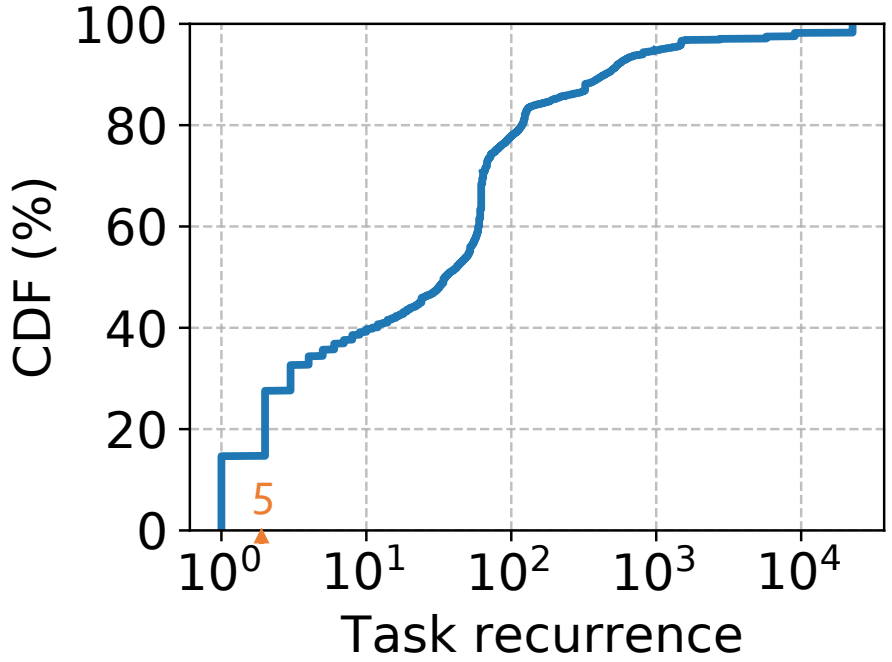
GPU Sharing

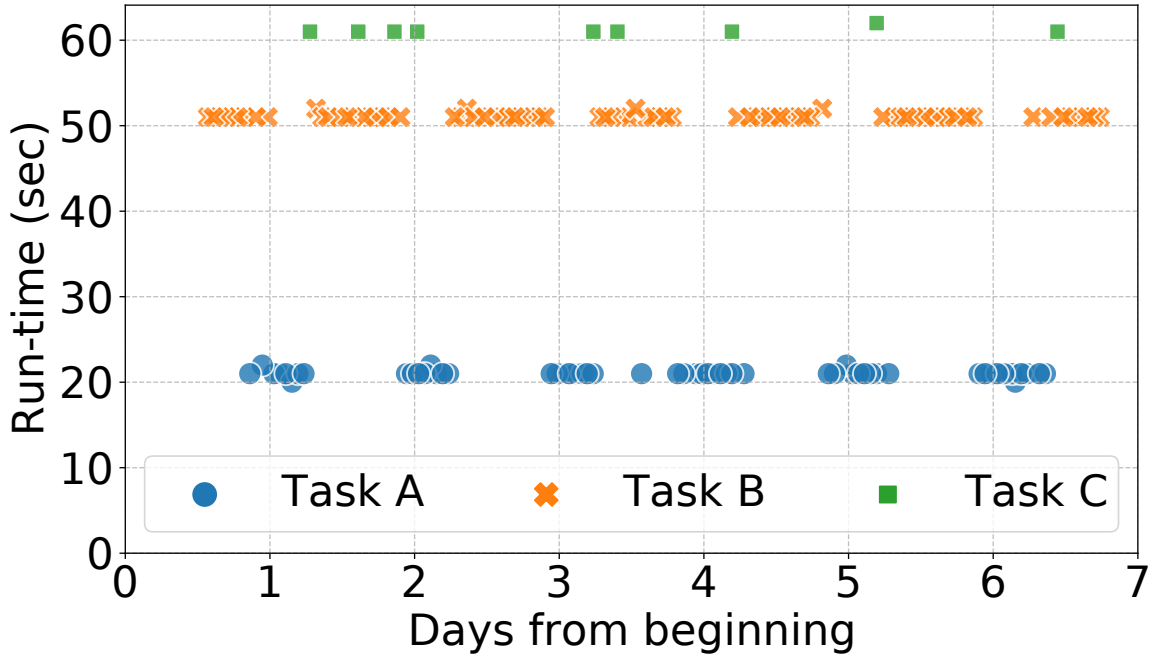
- Current GPU sharing does not cause severe GPU contention
  - On ≥ 95% utilized GPUs, only ~4% of them have ≥ 2 workers run simultaneously



Duration Predict for Recurring Tasks

- Recurring tasks are prevalent (65% tasks repeat ≥ 5 times)
  - Criteria of repetition: Group tag—hash value of jobs' customized input (e.g., entry scripts, command-line params, data sources and sinks)

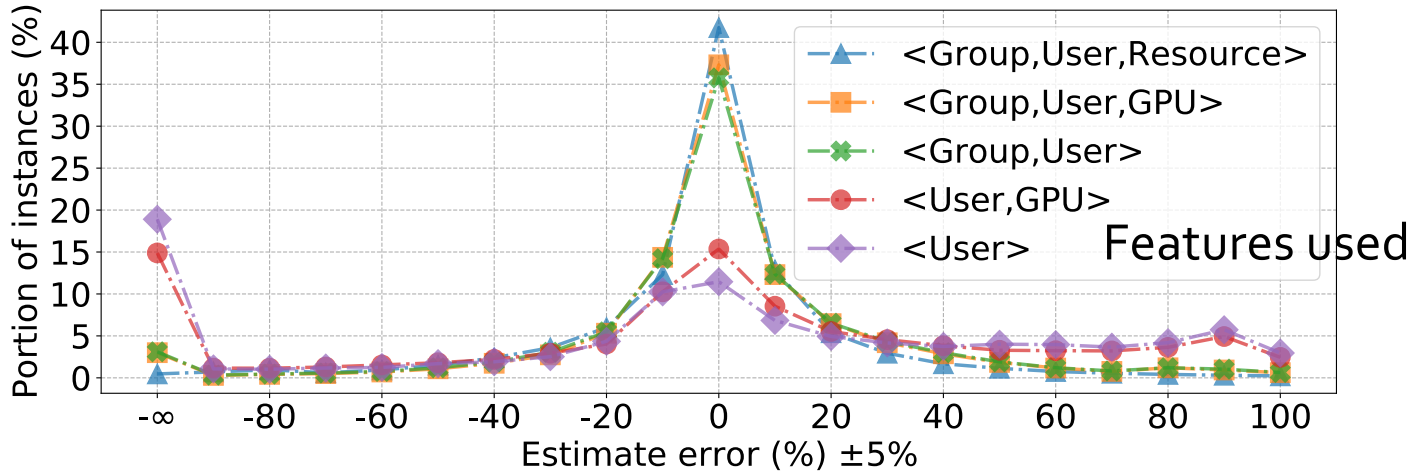


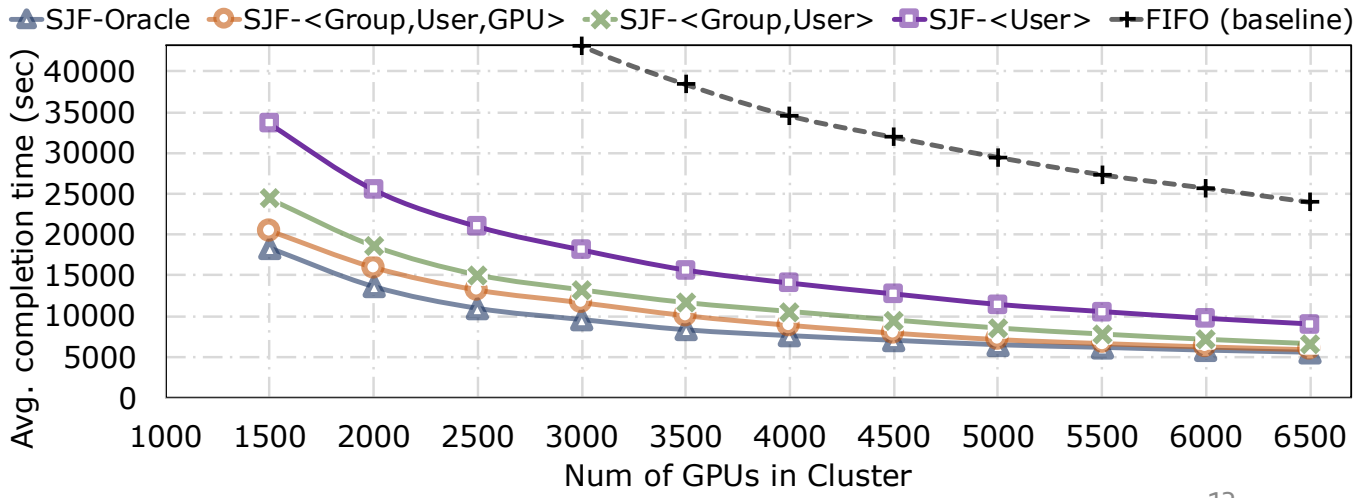


Sample inference tasks

Duration Predict for Recurring Tasks

- Predictor: Regression tree
  - Predict 78% instances duration within ±25% error
- Scheduling: SJF vs. FIFO
  - Speedup by 63%-77% with different predictors



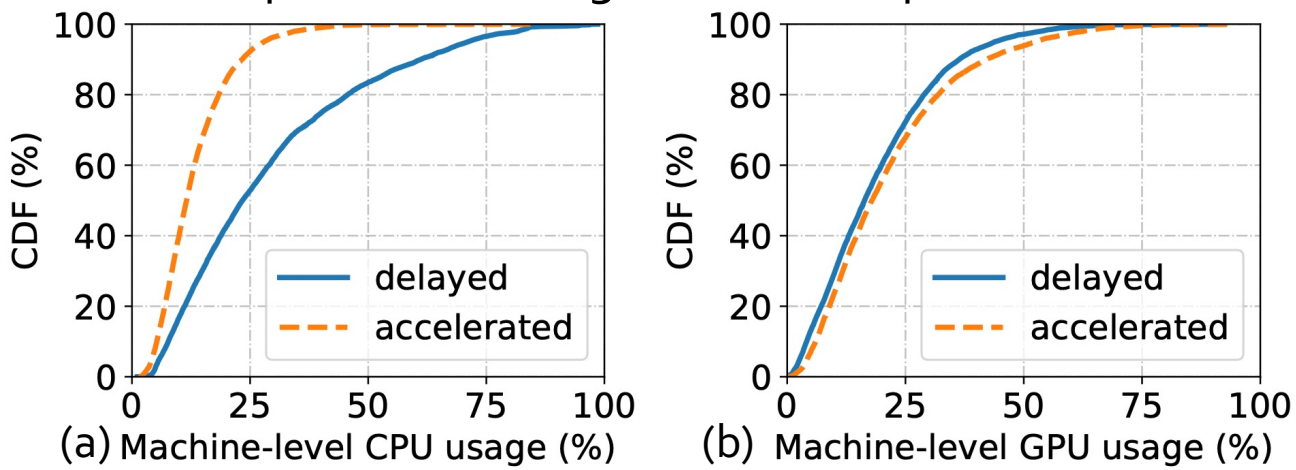


Challenges we met

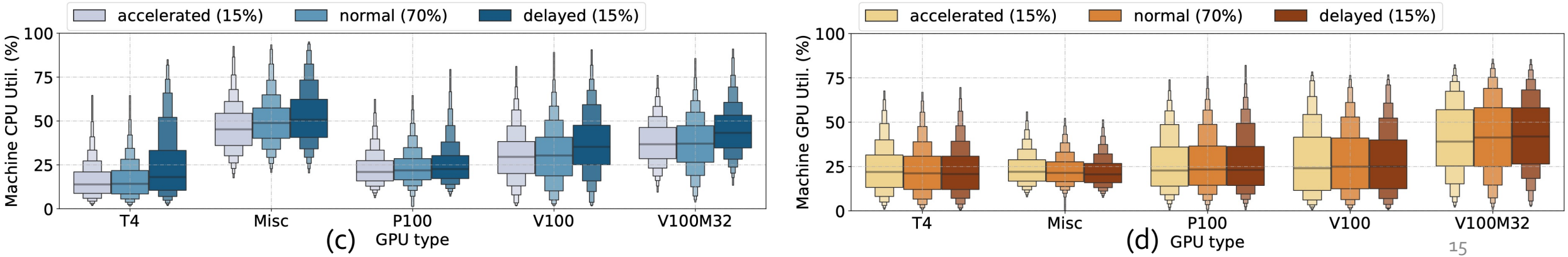
CPU can be the bottleneck

- Task delays are more likely to be observed under high CPU util. (a) (c), rather than high GPU util. (b) (d)

A sample Click-Through Rate (CTR) prediction task

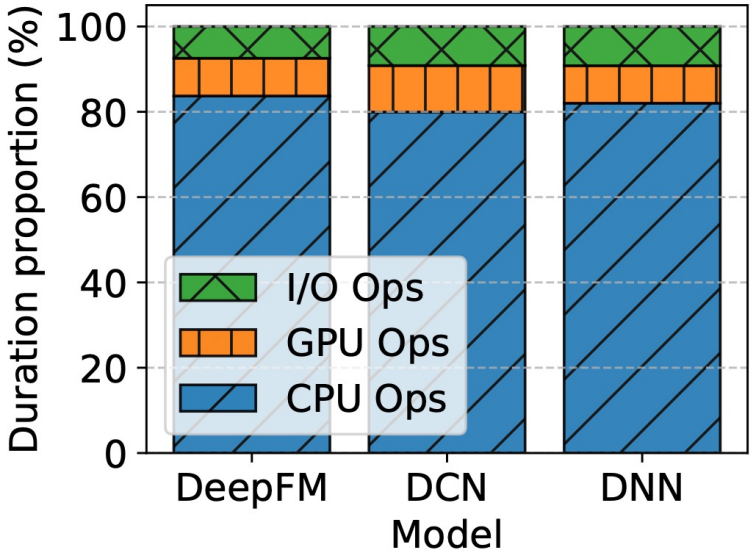


General tasks on various GPU nodes

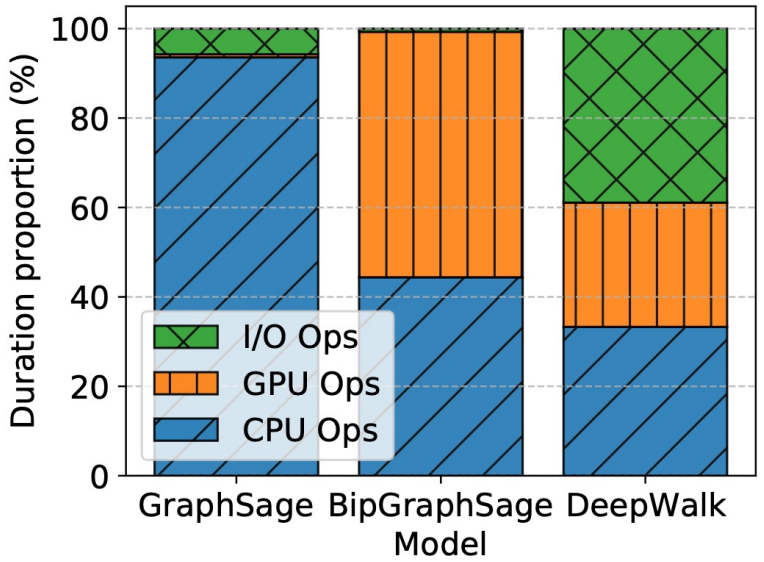


Majority of High-CPU-Low-GPU Tasks

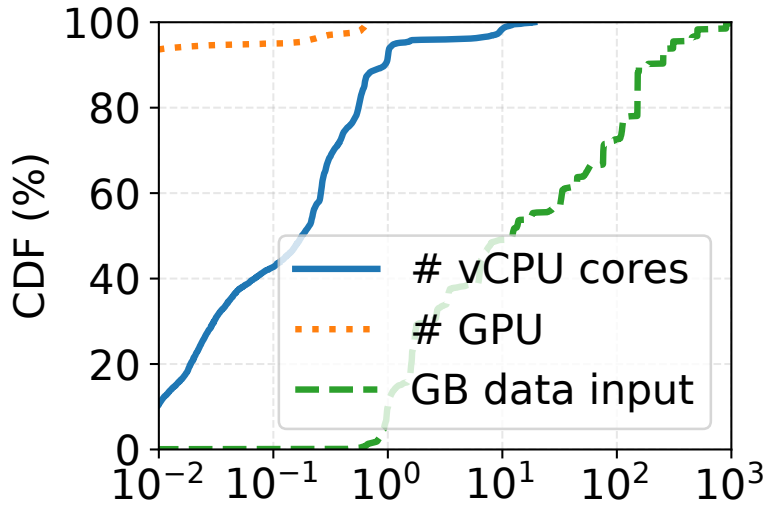
- CTR tasks could spend ~80% time on CPU (fetch and pre-process)
- GNNs spend 30-90% time on Edge Iteration, Neighbor Sampling, …
- RL launches massive CPU-intensive instances to run simulations



(a) Recommending: CTR models



(b) Graph Neural Networks (GNNs)

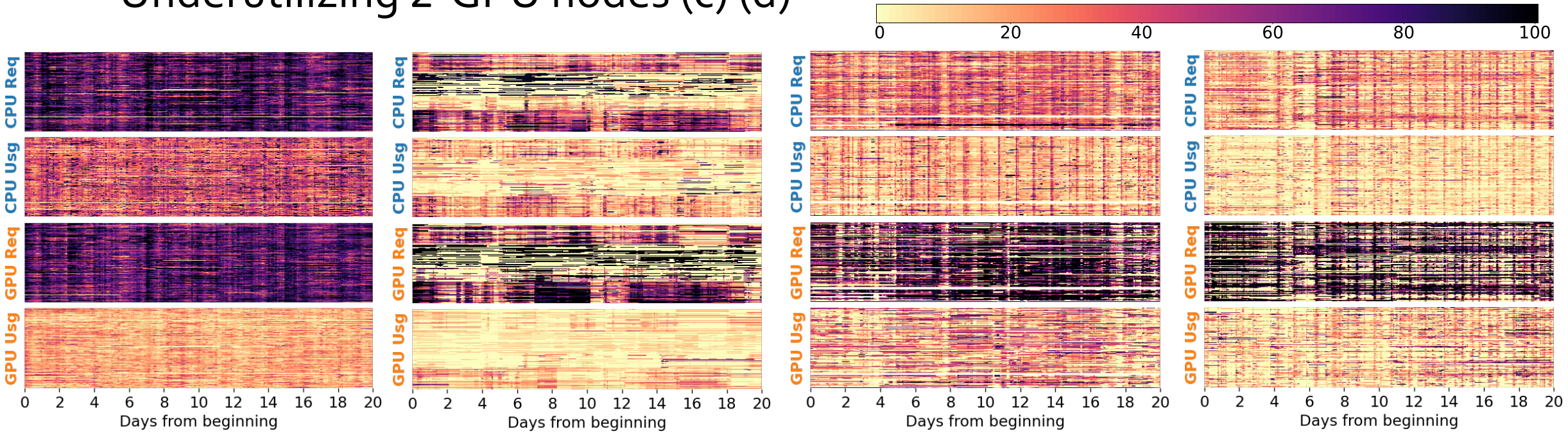


(c) Reinforcement Learning (RL)

Imbalanced Scheduling

- Overcrowding weak 8-GPU nodes (a)
- Packing on high-end V100 nodes (b)
- Underutilizing 2-GPU nodes (c) (d)

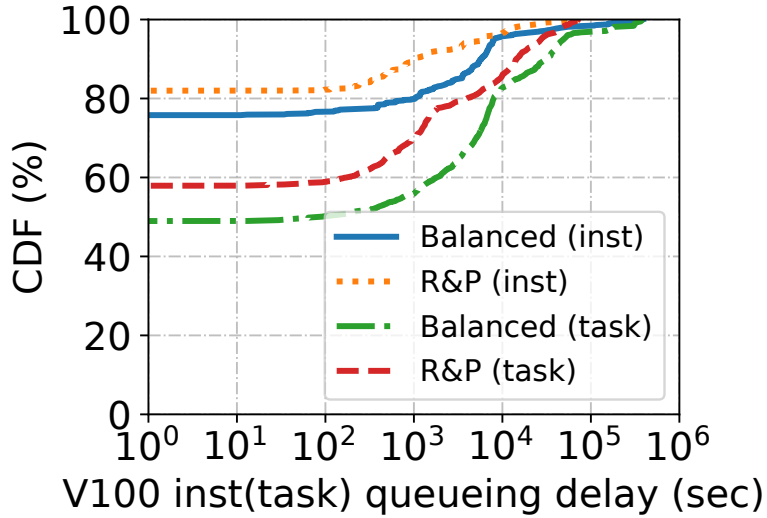
vCPU cores per GPU	All nodes	8-GPU nodes	2-GPU nodes
Machine specs	23.2	12.0	38.1
Instance requests	21.4	22.8	18.1



(a) 8-Misc-GPU Nodes (b) 8-V100-GPU Nodes (c) 2-P100-GPU Nodes (d) 2-T4-GPU Nodes

The Deployed Scheduling Policy

- Reserving-and-packing
  - Reserves high-end GPUs for high-GPU tasks
  - Packs other workloads to less-advanced GPUs
- Load-balancing
  - Assigns instances to nodes with low allocation rate



- Our scheduler prioritizes reserving-and-packing over load-balancing
  - Avoids extremely long latency for multi-GPU training jobs.
  - Simulation: V100 instances wait 68% shorter time in avg. with R&P than Balanced.

Takeaways

- A majority of tasks have gang-scheduled instances, mostly small in GPU
- GPU sharing and duration prediction of recurring tasks can help
- CPU could be the bottleneck, esp. under imbalanced scheduling

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Alibaba Group 阿里巴巴集团

Traces are released at: https://github.com/alibaba/clusterdata/tree/master/cluster-trace-gpu-v2020