Network Contention-Aware Cluster Scheduling with Reinforcement Learning

Junyeol Ryu
junyeol@aces.snu.ac.kr

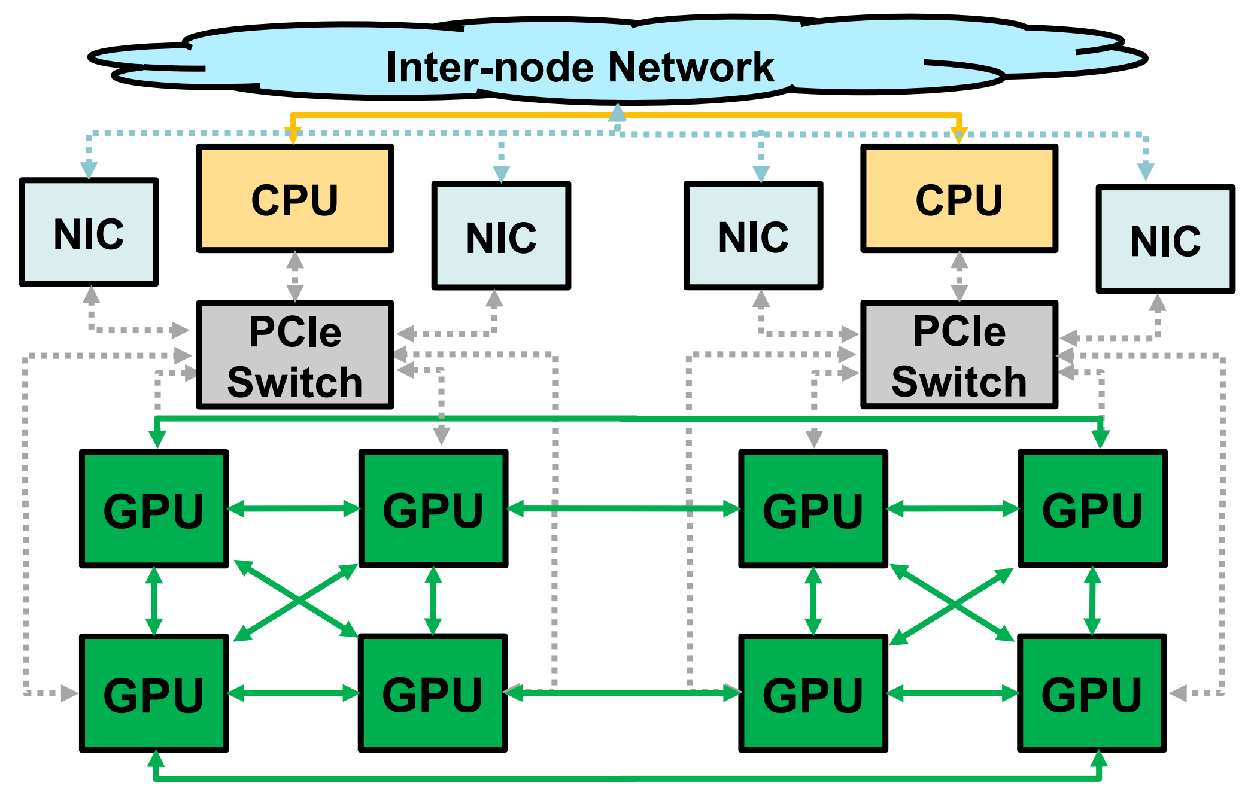
Jeongyoon Eo
jeongyoon.eo@snu.ac.kr

Distributed deep learning

Deep learning (DL)
Training DNN to analyze & derive knowledge from data

Parallel training
Large DNN is partitioned to multiple GPUs

GPU cluster
Shared cluster has multiple concurrent distributed DL jobs



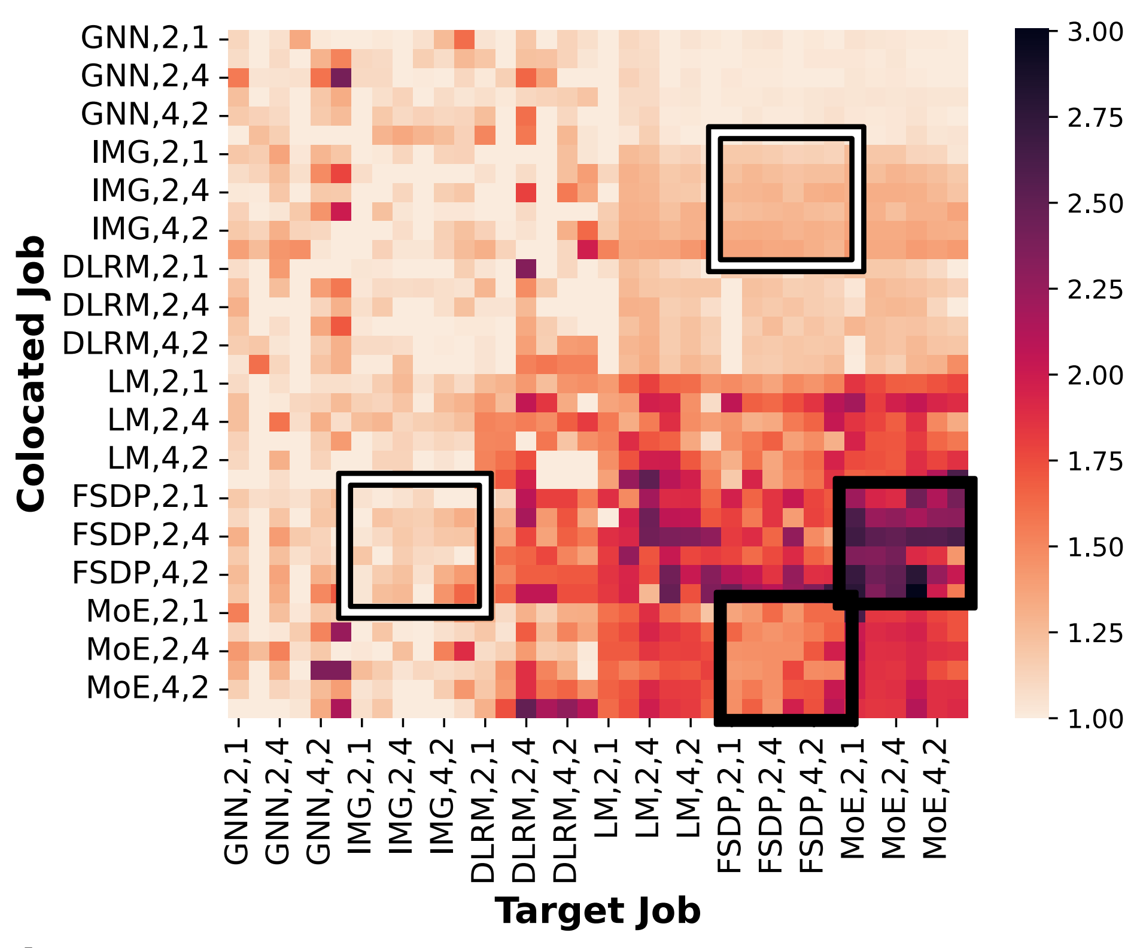
Network contention

Although GPUs are dedicated to jobs,
contention occurs from **shared network links** (PCIe, InfiniBand, etc.,)

Unfavored scheduling
Naïve co-locating FSDP and MoE suffers 49.1% and 66.7% throughput (darkest spots in the black boxes)

Contention-aware scheduling
By changing how they are co-located, above example can be improved up to 21.6% and 19.5%
(lightest spots in the black boxes)

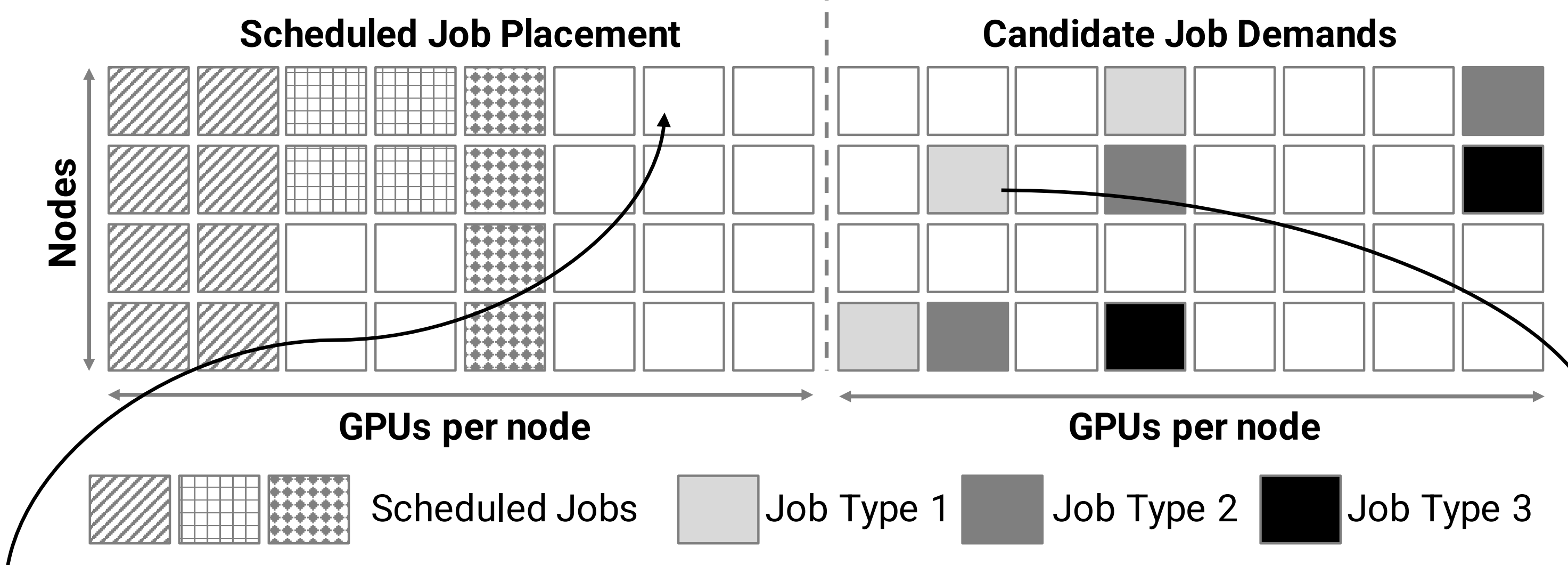
Also, colocation of certain models do not suffer network contention
(e.g., IMG and FSDP; white boxes)



Takeaway
Communication characteristics of model & Placement & Co-located job
= Exhibit diverse *contention sensitivity* ($\frac{throughput_{ideal}}{throughput_{contention}}$)

Scheduling by (1) *continuously* capturing contention sensitivity of jobs and (2) *adaptively* reducing expected contention will effectively alleviate cluster-wide network contention!

RL formulation



▨ Scheduled Jobs ⬜ Job Type 1 ▩ Job Type 2 ⬛ Job Type 3

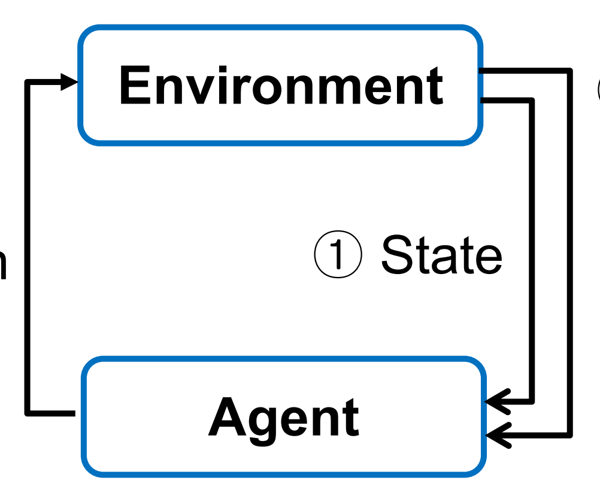
State
Fixed state design allows state to be applied as input to neural network-based RL algorithm

$slot_{i,j}$: Job can distribute its total $j * 2^i$ demands across 2^i nodes with j GPUs each

Action
Reducing action space is critical for faster convergence!
{ Job 1: [Node1, Node2], … }

Training algorithm
Agent is trained with a neural network-based RL algorithm using job traces on a simulated GPU cluster environment

Iterate rounds until all jobs complete …



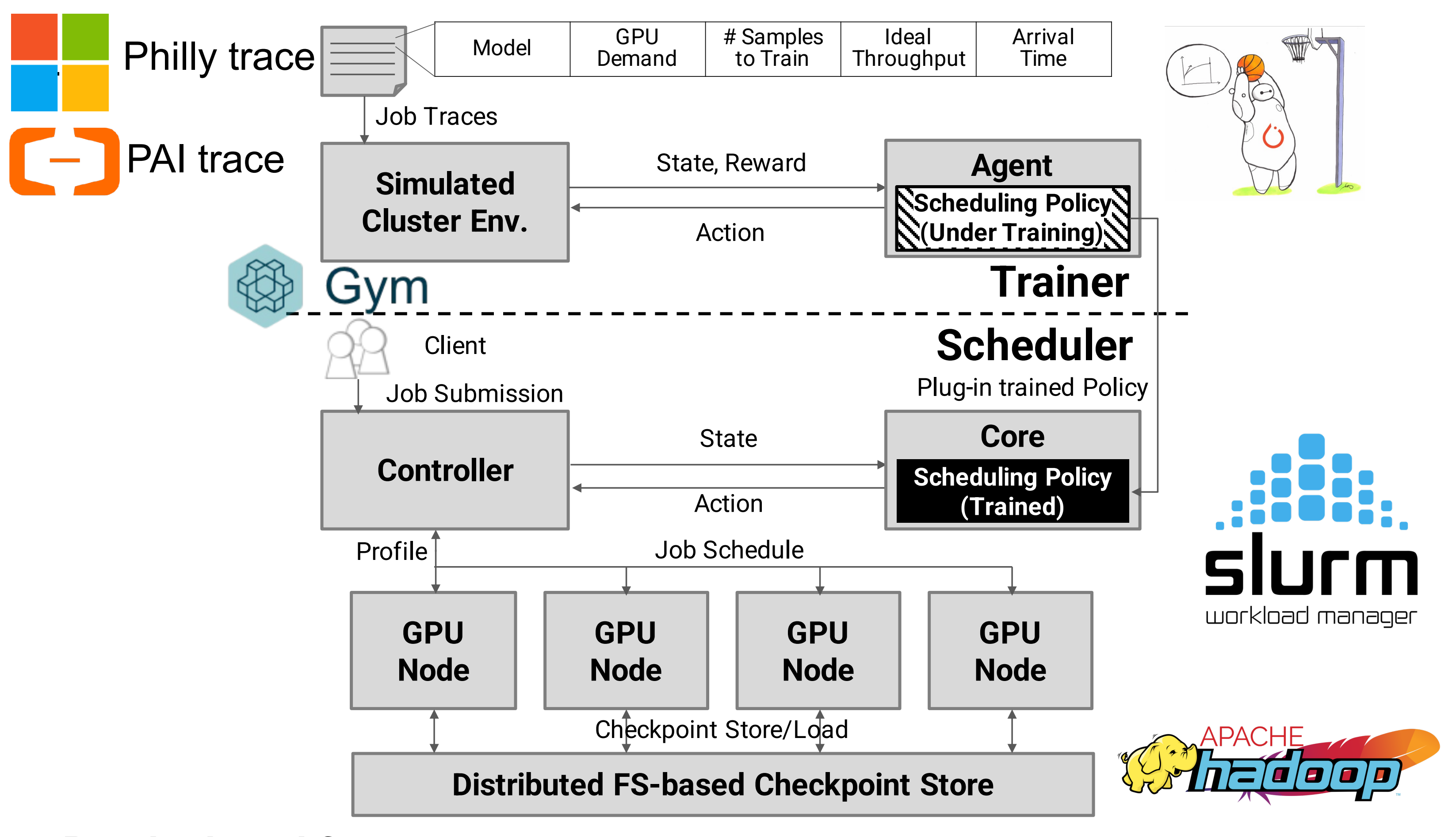
③ **Reward** = $-W_1 * CS + W_2 * Util$
- Penalize increase in cluster-wide average contention sensitivity of scheduled jobs
- Incentivize increase in cluster-wide average GPU utilization

Training an episode of 256 jobs completes in less than a minute

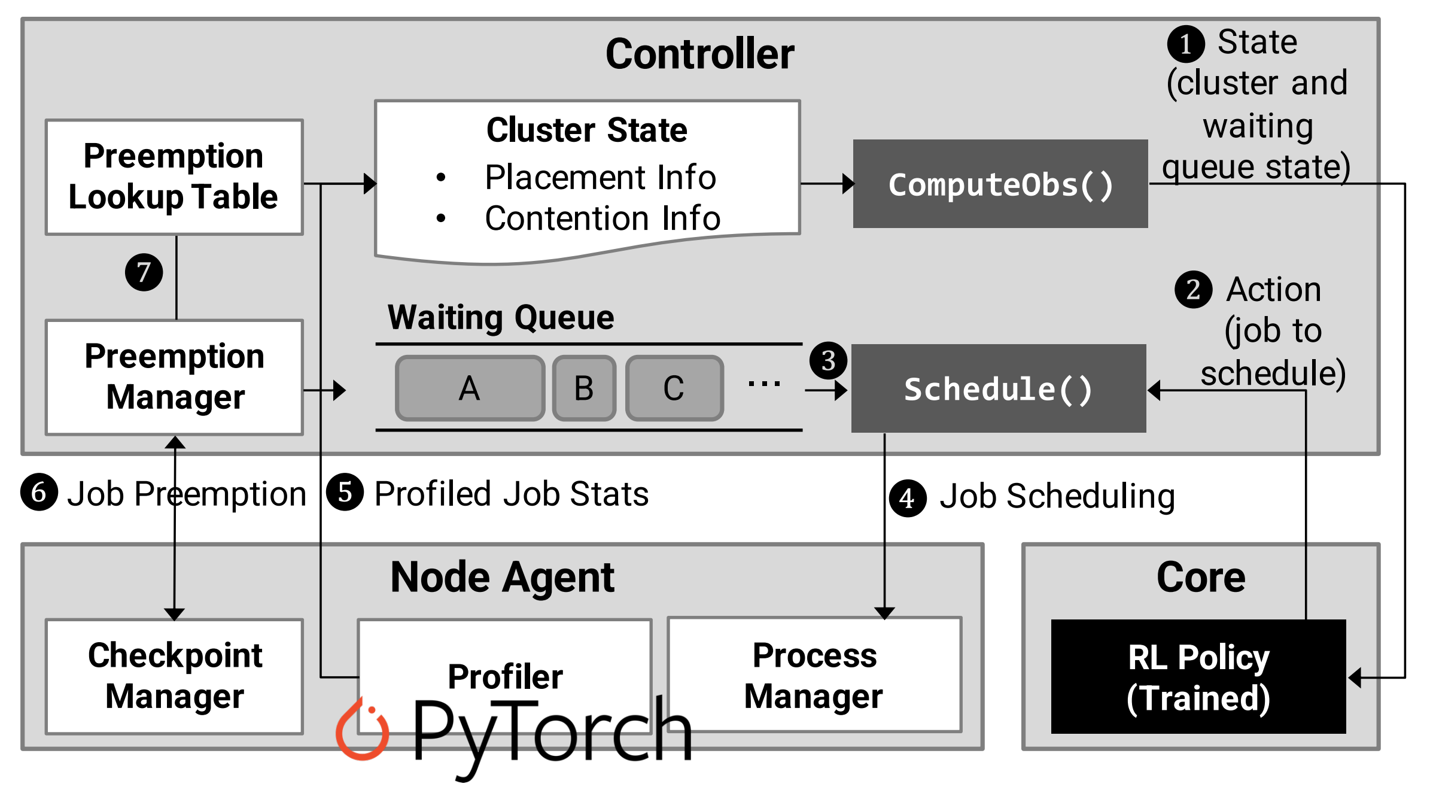
Benchmark policies
Initial scheduling policies trained with above RL

- RL-base: Make scheduling decisions only based on the action of trained policy (May decide not to schedule even when cluster has enough resources to avoid increase in performance degradation due to contention)
- RL-hybrid: Decision-level multiplexing of trained policy and simple rule-based policy (Usually follow trained policy, but may try safety rule of trying greedy scheduling to prevent low utilization)

System

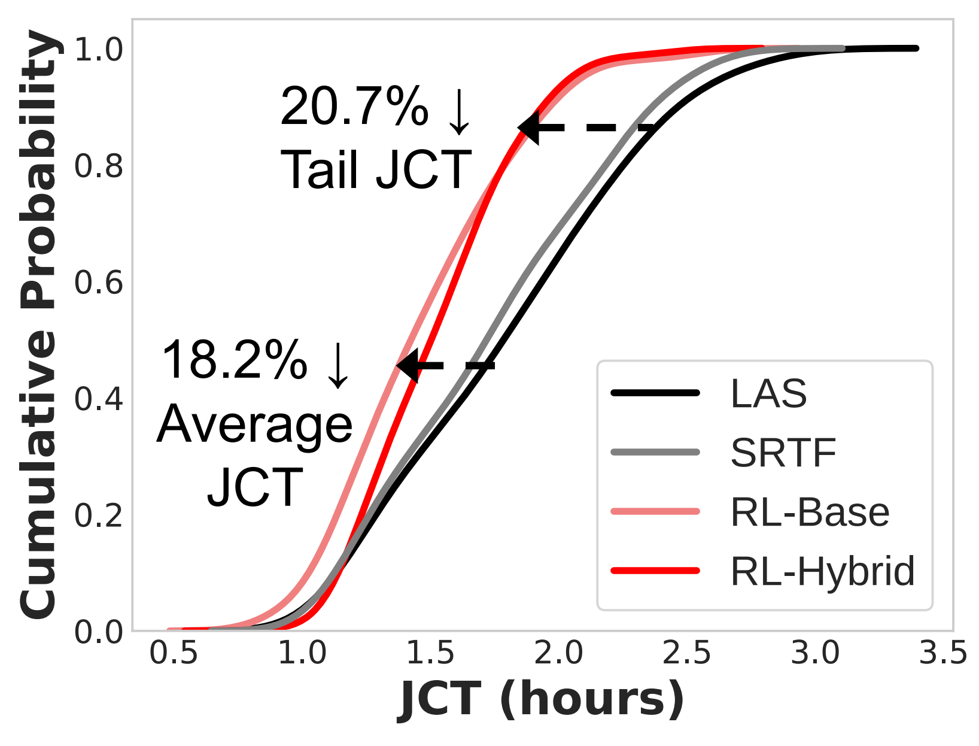


Detailed workflow

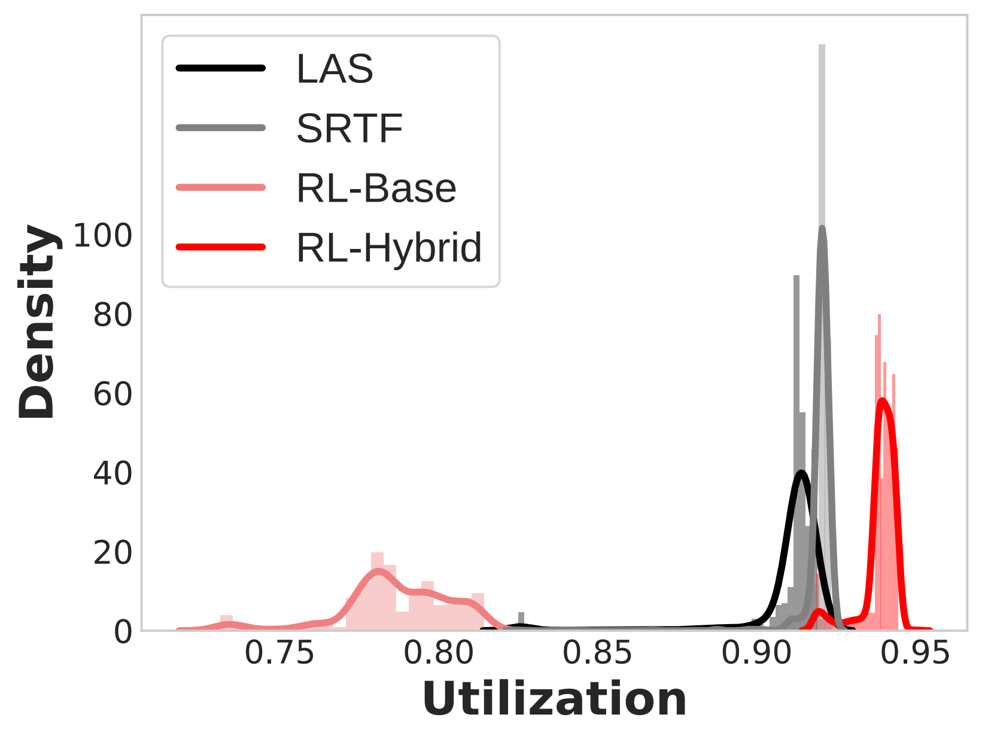


Evaluation

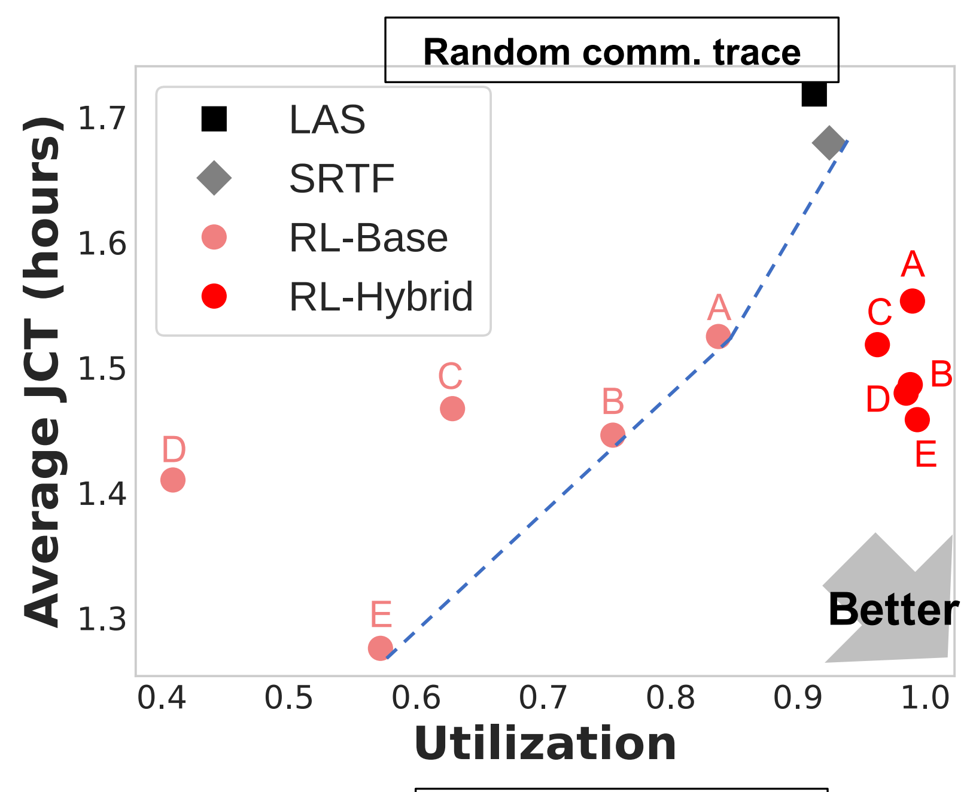
Hardware	Workload	Baseline
• 4 homogeneous nodes • Each node equipped with • 8 NVIDIA TITAN XP GPUs • PCIe Gen3 (16GB/s) • Mellanox ConnectX-4 NIC • InfiniBand (40Gb/s)	Four DL job traces with different communication intensities • Contain ten job sets (256 jobs) • Each job has • GPU demand: ≤ 32 • Expected training time: 1 hour • Random initialized configuration	Widely used scheduling policy • Least Attained Service (LAS) • Shortest Remaining Time First (SRTF)

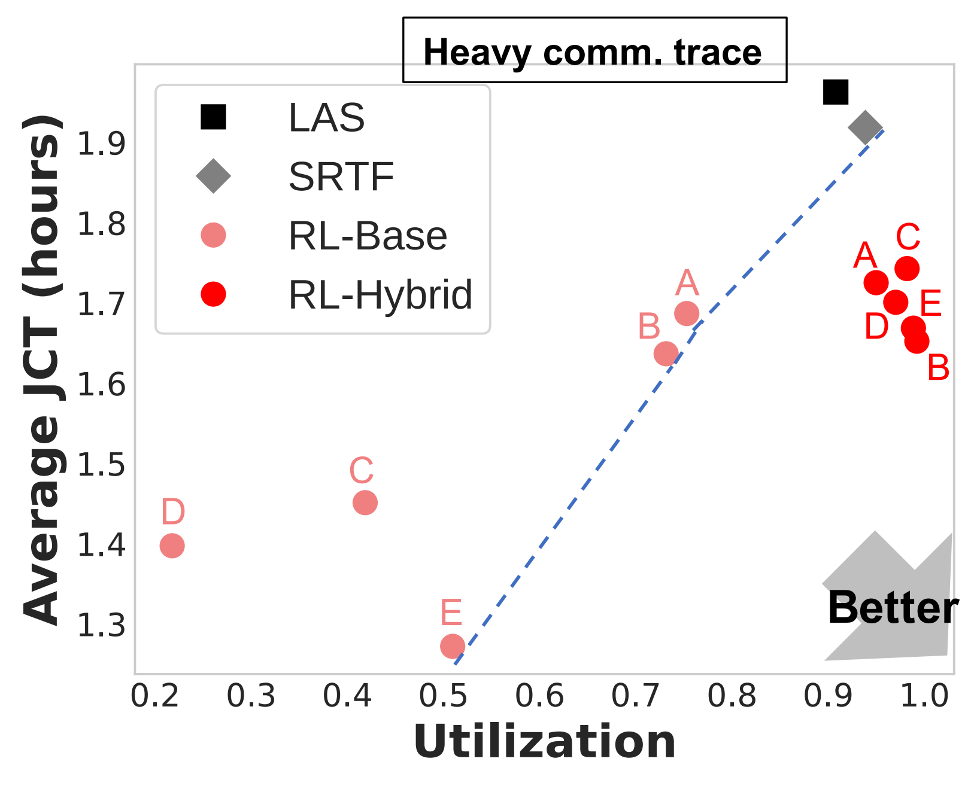


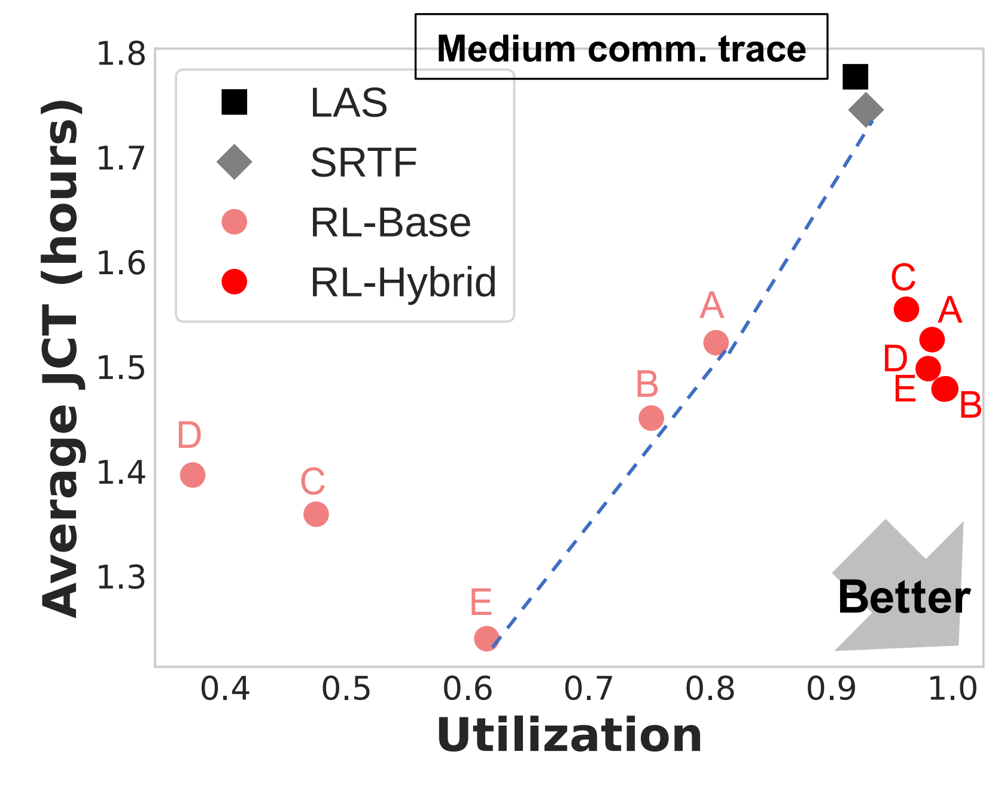
Larger gain on tail JCT means effectively avoids scheduling decisions that trigger worst contention situations

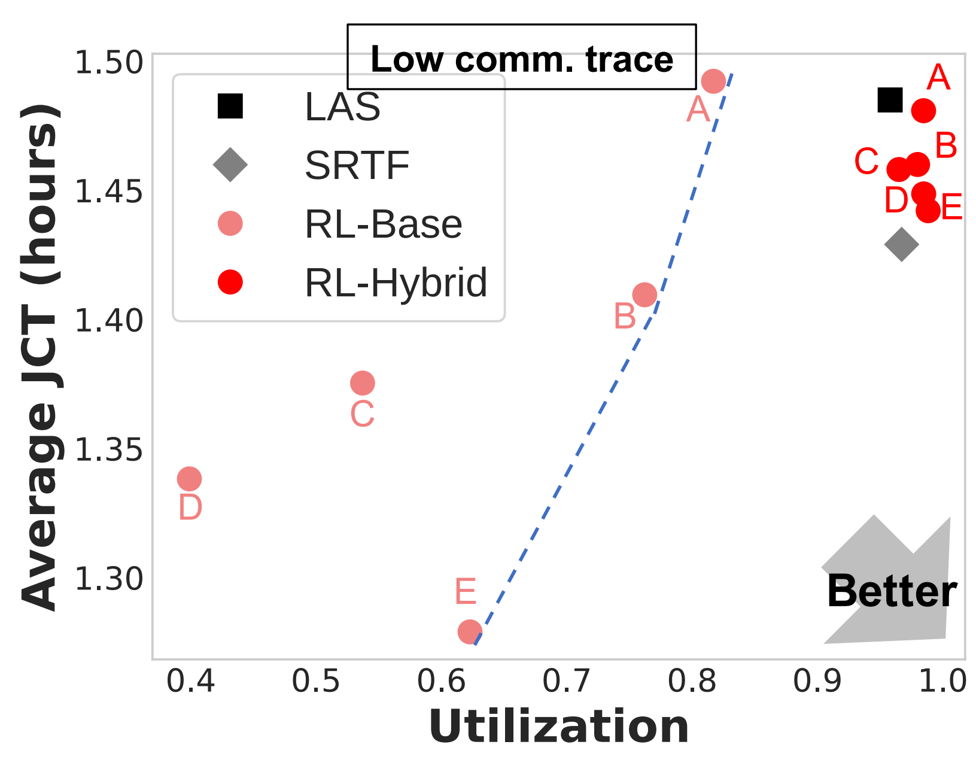


RL-Hybrid trades small JCT cost for large utilization improvement









RL-Base shows balanced trade-off with baseline, while RL-Hybrid shows best performance