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*, and Yu Ding*

† Hong Kong University of Science and Technology *Alibaba Group

Motivation

Challenges in scheduling ML workloads

- Characteristics:
  - Heterogeneous ML workloads and GPU machines
- Problems
  - Low utilization caused by fractional GPU uses
  - Long queueing delays for short-running task instances
  - Hard to schedule high-GPU tasks
  - Load imbalance
  - Bottleneck on CPUs

Key insights

- Key insights that the paper leverages to solve the problem
    - GPU sharing
    - Predictable Duration for Recurring Tasks (Shortest Job First)
- Key contributions
    - Profiling of PAI traces
        - Temporal pattern
            - Recurring tasks
            - short-running instances usually spend a larger portion of time in queueing
        - Spatial pattern
            - Heavy tail distribution
            - CPU bottleneck
    - New scheduling algorithm

Shortest Job First scheduling

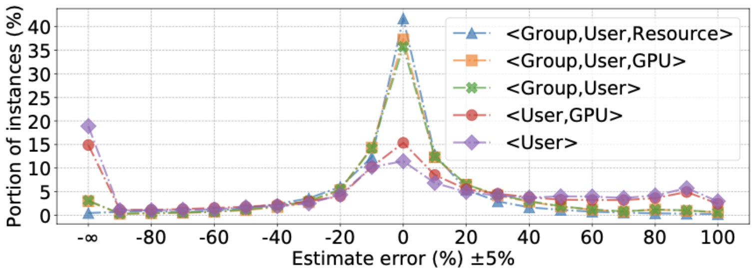


Figure 12: Percentage prediction error, i.e., $(true-pred)/true$ in percentage, of duration estimates with different features.

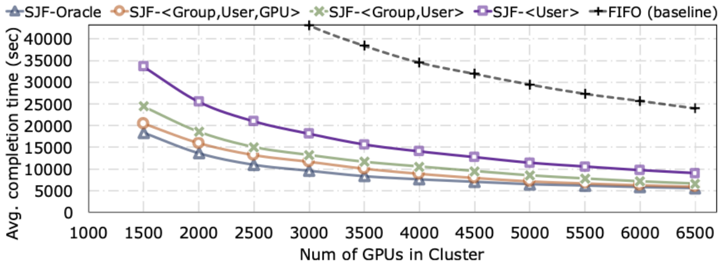


Figure 13: Average task completion time given different GPU cluster sizes and various scheduling policies in simulation.

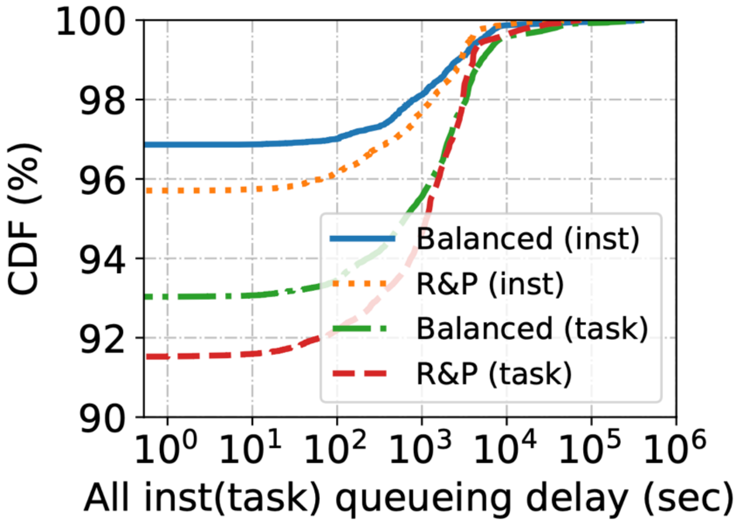
Predicting duration of recurring tasks by hashing metadata

Lower avg completion time using SJF

System

- Scheduling policy
  - Reserving-and-packing scheduling policy
    - Prioritize high-GPU tasks (by definition of computation efficiency)
      - a performance model that accounts for many task features, such as the degree of parallelism, the used ML model, the size of embedding
  - Load balancing
    - prioritizes instance scheduling to machines with low *allocation rate*
- Tradeoffs
  - Reserving-and-packing >> Load-balancing
  - Fairness of reserving-and-packing

Reserving-and-packing vs Load-balancing



(a) Queueing delays of all instances and tasks.

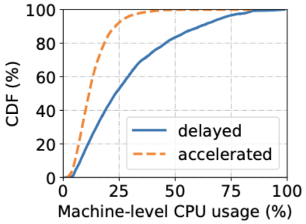
Evaluation

- Open Challenges
  - Mismatch between machine specs and instance requests (#CPUs vs #GPUs)

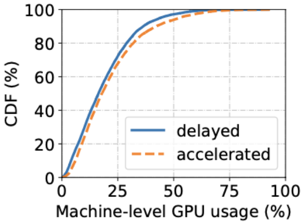
Table 2: Mismatch between machine specs and instance requests, in terms of the provisioned/requested CPUs per GPU.

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

  - Overcrowded weak-GPU machines vs less crowded high-end machines
  - CPU bottleneck
    - Especially for some ML workloads (CTR)



(a) CDF of machine CPU usage.



(b) CDF of machine GPU usage.

Discussion

- Strengths
  - Comprehensive profiling of the system
  - Identified the insight of recurring tasks
    - Go into the details of recurring tasks → SJF scheduling algo
  - Prediction of task duration is accurate and well evaluated
  - Graphs show CDF of queueing delay
- Critique
  - Could have done more evaluation of the improved scheduling algorithm
    - Comparison of R&P vs load-balancing doesn’t show the interplay of the two
    - Missing comparison of the final algorithm vs the original
  - What are some other alternatives
    - Other ways of leveraging the properties identified
  - More details on GPU sharing

Discussion

- Clarifying questions
  - What are some intuitive reasoning on how different algorithms have different distribution of IO/GPU/CPU time
  - Details of scheduling algorithm
    - What constitutes an allocation plan? What are the buckets of machines?
- Discussion and Debate:
  - Benefits and Challenges of having heterogeneous machines
  - GPU sharing mechanism
    - How it is done, see paper 2
  - De-coupling CPU work from GPU work
  - CPU bottleneck: research to reduce CPU time in data processing