# Optimizing Resource Management for Machine Learning Workloads in High-Performance Clusters

Di Zhang
Computer Science Department
University of North Carolina at Charlotte
dzhang16@uncc.edu

Dong Dai
Computer Science Department
University of North Carolina at Charlotte
ddai@uncc.edu

***Abstract*—Resource management and job scheduling are the key to high-performance computing (HPC) clusters for high system utilization, short user wait time, and fair resource allocation. The effectiveness of job scheduling policies normally depends on how well they fit the characteristics of clusters. With the growing proportions of machine learning jobs in HPC clusters, it is mysterious what the hybrid HPC traces are like, whether traditional scheduling policies of HPC still work efficiently and how is the performance if applying scheduling policies from Machine Learning (ML) datacenters to HPC clusters. This study unveils the mystery by characterizing and comparing the similarities and differences between HPC traces and ML traces. We firstly synthesize hybrid traces with awareness of the characteristics. We also summarize scheduling policies from large-scale HPC clusters and ML datacenters and compare them on the hybrid traces to shed light on the design of scheduling policies for new-era HPC clusters.**

## I. Introduction

Resource management and job scheduling are critical in HPC platforms to determine the order of execution of batch jobs submitted by users. The quality of job scheduling policies has a large impact on the effectiveness of jobs, the utilization of computing resources, and the satisfaction level of users. The jobs in HPC platforms were traditionally dominated by large scale simulation programs. Recently, with the rapid development of Artificial Intelligence (AI) and Machine Learning (ML), a variety of AI/ML jobs have occupied a considerable part of batch jobs [5], [6], which leads to a new challenge of Hybrid Jobs Scheduling.

Recently, various studies of job characterizations have been done to understand either HPC jobs or ML jobs. Patel et al. [4] analyzed a decade of HPC jobs to support several long-standing traditional knowledge and identify many previously unknown trends and their consequences. Hu et al. [1] investigated the characteristics of ML jobs in their own datacenter and compared the findings with a previously similar study, conducted by Jeon et al. [2], on ML jobs.

Understanding the characteristics of HPC jobs and ML jobs separately can properly help administrators manage and operate large clusters where resources for different types of jobs are dedicated. However, the learning for each type of jobs can not directly be applied to the scenario where jobs are hybrid. For example, ML jobs are known for high cancel rates [2], [1] but HPC jobs are not. The scheduling policies taking advantage of the high cancel rate may not thrive in clusters with hybrid jobs. To derive a sophisticated scheduling policy for hybrid jobs, investigating the characteristics of hybrid jobs is crucial. Unfortunately, the lack of online-available traces of hybrid jobs makes the effect of directly applying current scheduling policies to them unknowable, and also hinders the exploration of scheduling policies that are most suitable for them.

## II. Background

### A. Scheduling Policies

Job scheduling serves as the key part in HPC platforms and ML datacenters to determine the order of execution of jobs submitted by users. The quality of batch job scheduling has a large impact on the effectiveness of jobs, the utilization of computing resources, and the satisfaction level of users. To study and compare scheduling policies of the systems under analysis, we need to know in detail what scheduling policies they used. Table I displays five scheduling policies and the systems where the corresponding policies are proposed or used. First Come First Serve (FCFS) is the simplest but widely-used scheduling policy. It prioritizes waiting jobs by the submit_time ($s_t$) of them, and the less $score$ means the higher priority. Blue Waters does not publish its exact scheduling policy. However, it mentioned some high-level factors, such as *"Larger jobs generally get priority over smaller jobs."* [3]. As a result, we use Largest Job

TABLE I: List of scheduling policies

| Name | System | Function |
|---|---|---|
| FCFS | N/A | $score(t) = s_t$ |
| Largest | Blue Waters | $score(t) = -n_t$ |
| QSSF | Helios | $score(t) = predicted\_run\_time_t$ |
| WFP3 | Intrepid/Mira | $score(t) = -(w_t/r_t)^3 * n_t$ |
| UNICEF | Intrepid/Mira | $score(t) = -w_t/(log_2(n_t) * r_t)$ |

First (Largest) as an approximation of its scheduling policy. $n_t$ represents the requested nodes of $job_t$. The administrators of SenseTime designed a novel scheduling policy, Quasi-Shortest-Service-First (QSSF) [1], for their own Helios datacenter, which basically prioritizes jobs based on the predicted run time. WFP3 and UNICEF were used for production jobs and development jobs respectively on Intrepid, where $w_t, r_t$ represent wait time and run time.

## III. Methodology

In this study, we generate hybrid job traces from several online available HPC traces and ML traces. Specifically, we sample jobs from HPC traces and ML traces and then mix these jobs together as a single hybrid trace.

Such a mix is straightforward but also biased since it highly depends on the selected HPC traces and ML traces. If these traces are not representative, the mix of them will only be a corner case with no insights. Even with the help of the synthetic hybrid trace, several critical questions are still not easily answered. For example, which option is better, to have separate dedicated resources for different types of jobs, or to share the common resources? Should ML jobs and HPC jobs share the same scheduling policy?

Based on the characterization, we have a better understanding of each trace and then consciously synthesize various hybrid job traces by mixing jobs with different properties from different traces in multiple mixing ratios. As a result, the synthetic hybrid traces cover a series of possibilities and will not be too biased toward the selected traces. To investigate the efficiency of recent scheduling policies, we apply them to the hybrid job traces to see the performance.

We summarize our contributions into threefold:

- We perform an analysis on a large-scale HPC trace and firstly compare the characteristics of HPC jobs with ML jobs to deeply understand their similarities and difference, and shed light on the future strategy of resource management for a combination of these jobs.
- We propose a way to synthesize hybrid traces (HPC jobs and ML jobs) with awareness of the properties of each type of jobs.
- We conduct evaluations to show the performance of a series of successful scheduling policies on hybrid traces which can potentially be used as baselines for the design of future scheduling policies.

TABLE II: Performance of different scheduling policies on Hybrid traces with various proportions of ML jobs.

| | *Different Proportions of ML Jobs* | | | | |
|---|---|---|---|---|---|
| *Scheduling Policies* | *8:1* | *6:1* | *4:1* | *2:1* | *1:1* |
| FCFS | 94042 | 75837 | 63725 | 40094 | 44235 |
| Largest | 134867 | 114159 | 105921 | 42950 | 50572 |
| QSSF | 5308 | 4119 | 5538 | 1121 | **284** |
| WFP3 | **1209** | **1003** | **768** | **336** | 426 |
| UNICEF | 38183 | 32004 | 26928 | 10021 | 14965 |

## IV. Preliminary Results

In HPC clusters, the ratio of ML jobs tends to be divergent. To formulate a reasonable guide for future work, we cover a series of ratios of ML jobs. It is noticed that we assume for HPC clusters, the number of HPC jobs should be dominant hence the proportion of ML jobs is no more than 50% in our settings. Table II shows the median wait time of each scheduling policy on hybrid traces with different ratios of ML jobs. We consider median over mean because there are quite a few outliers that may wait tens of days which biases the mean value a lot. The results are consistent for ratios of HPC jobs and ML jobs from 8:1 to 2:1 where WFP3 has the lowest median wait time. When ML jobs have a comparable amount with HPC jobs (1:1), QSSF becomes the best scheduling policy, but WFP3 still outperforms the other policies with a large margin.

## References


[1] Qinghao Hu, Peng Sun, Shengen Yan, Yonggang Wen, and Tianwei Zhang. 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*, pages 1–15, 2021.

[2] Myeongjae Jeon, Shivaram Venkataraman, Amar Phanishayee, Junjie Qian, Wencong Xiao, and Fan Yang. Analysis of {Large-Scale}{Multi-Tenant}{GPU} clusters for {DNN} training workloads. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, pages 947–960, 2019.

[3] NCSA. Blue waters scheduling. https://bluewaters.ncsa.illinois.edu/queues-and-scheduling-policies.

[4] Tirthak Patel, Zhengchun Liu, Raj Kettimuthu, Paul Rich, William Allcock, and Devesh Tiwari. Job characteristics on large-scale systems: long-term analysis, quantification, and implications. In *SC20: International conference for high performance computing, networking, storage and analysis*, pages 1–17. IEEE, 2020.

[5] Di Zhang, Dong Dai, Youbiao He, Forrest Sheng Bao, and Bing Xie. Rlscheduler: an automated hpc batch job scheduler using reinforcement learning. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15. IEEE, 2020.

[6] Di Zhang, Dong Dai, and Bing Xie. Schedinspector: A batch job scheduling inspector using reinforcement learning. In *Proceedings of the 31st International Symposium on High-Performance Parallel and Distributed Computing*, HPDC '22, page 97–109, New York, NY, USA, 2022. Association for Computing Machinery.