Pollux: Co-adaptive Cluster Scheduling for Goodput-Optimized Deep Learning

Seminar-Advanced Topics in AI for Networking

Aurick Qiao, Sang Keun Choe. et. al
Petuum, Inc. Carnegie Mellon University. UC Berkeley. MBZUAI.
15th USENIX Symposium & Awarded best paper!

August 16, 2022

Presented by Abdullah Amawi. University of Göttingen

Table of contents

Introduction

Background (Current Workflow)

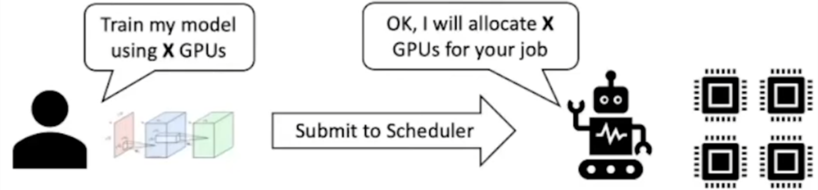

Figure 1: Shared-Cluster DL training workflow[1]

Considerations:

- Cluster contention: dynamic change based on usage
- Job scalability: which needs expert knowledge
- Training parameters: batch size & learning rate tuning

Background (Parallel Cluster DL)

- Distributed deep learning training requires a lot of optimizations for batch size and learning rate.

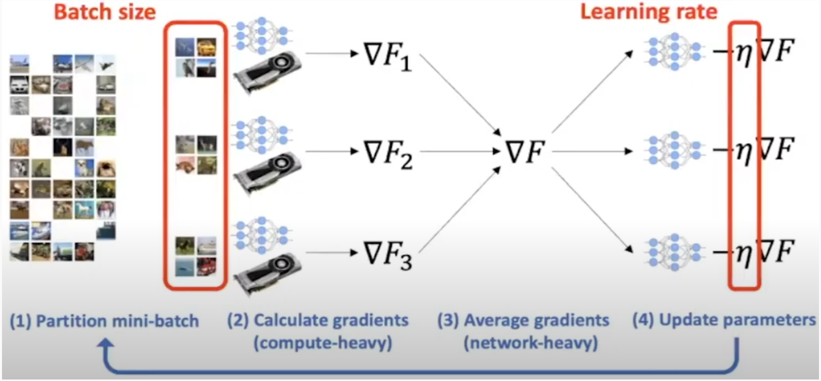

Figure 2: Distributed deep learning data parallelism[1]

Background(DL GPUs & Efficiency)

- Many GPUs doesn't translate to better throughput without optimizations; System statistical efficiency is still an issue!.

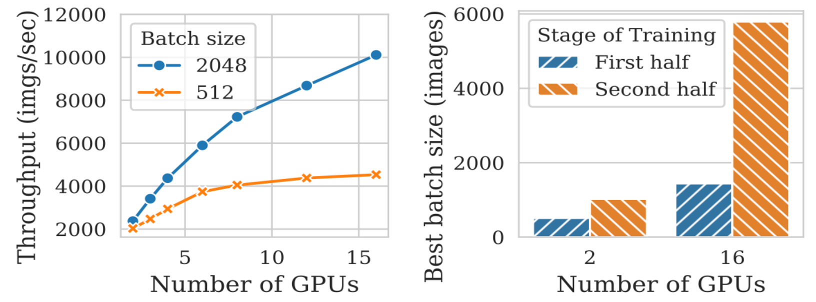

Figure 3: Number of GPUs and system efficiency [1]

Background(DL Optimal batch size)

- Batch size relationship to training performance, system throughput and statistical efficiency changes dynamically.

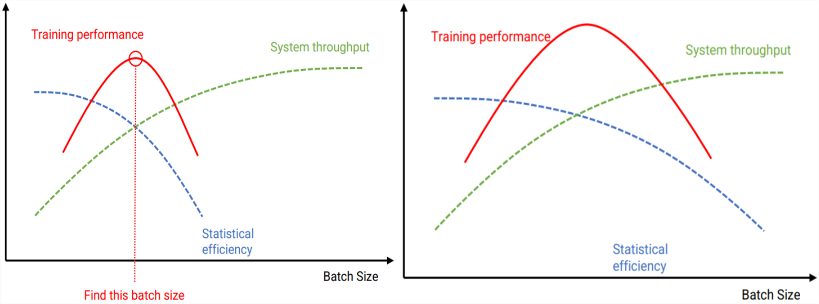

Figure 4: Batch size & effects on different training stages[1]

Problem Definition - Cluster Scheduling Challenges

- Choosing good GPU allocation for DL depends on batch size, while batch size depends on the allocated GPUs, we have inter-dependency.
- GPU allocation is cluster-wide which relates to fairness and contention.
- Inter-dependant factors makes it very hard for users while submitting the jobs.

Existing Solutions & Related Works

- Existing DL schedulers Tiresias & Optimus[1] exist but with downfalls.
- Adaptive DL schedulers: Optimus adapts the number of GPUs only, Tiresias doesn't adapt dynamically.
- Adaptive batch size training: Many works exist[1] but only Pollux addresses realistic scenarios and cluster scheduling.
- Hyper_parameter tuning: Pollux aims to improve that with dynamic adaptation of parameters and inter-dependant factors.

Pollux.

Pollux Idea

Pollux revolves around these main ideas:

- Dynamic tuning of batch size and learning rate to utilize resources better
- Pollux introduces ”Goodput”, a new measure of DL training performance that combines throughput w.r.t efficiency.
- Pollux also considers cluster-wide performance, fairness and dynamically re-allocates resources to come up with better Goodput.

System Architecture

- Pollux system Architecture can be visualized by the following:

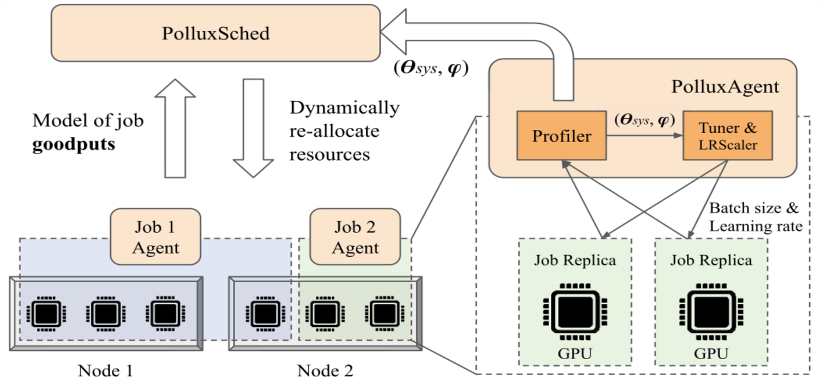

Figure 5: System architecture of Pollux[1].

- Pollux cluster scheduler illustration:

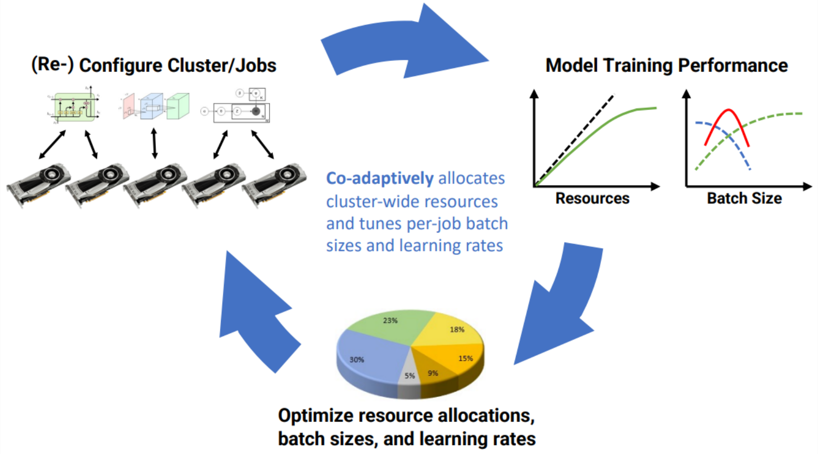

Figure 6: Pollux cluster scheduler illustration[1].

Pollux System Modelling

Pollux GOODPUT

$$GOODPUT_t(a, m, s) = \underbrace{THROUGHPUT(a, m, s)}_{\substack{\text{System throughput} \\ \text{(training examples / second)}}} X \quad \underbrace{EFFICIENCY_t(M)}_{\substack{\text{Statistical efficiency} \\ \text{(progress/training example)}}}$$

- a: Allocation vector (Number of GPUs)
- m Per-GPU batch size
- s Gradient accumulation steps (to enable total batch sizes larger than GPU limit)
- M: Total batch size.
- a, m, s are automatically determined by Pollux.

Pollux System Throughput Modelling

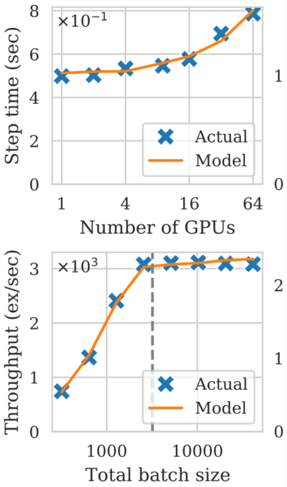

Figure 7: System throughput(Imagenet)[1]

- Models throughput per GPU count > choose right number of GPU and batch size.
- Models gradients accumulation > increase batch size beyond GPU memory limits.
- Models GPU node allocation > pack job's GPU onto fewer nodes to minimize network overhead.

Pollux Statistical Efficiency Modelling

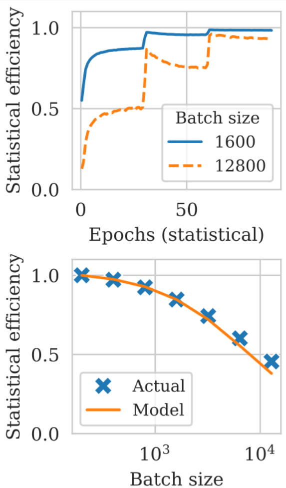

Figure 8: Statistical efficiency(Imagenet)[1]

- Lower statistical efficiency for larger batch sizes > improves later in training.
- Accurately predicts statistical efficiency for different batch sizes > improves GOODPUT.
- Pollux predicts GOODPUT > even before running the job.

Evaluation & Results

Evaluation

- Pollux vs two SOTA DL schedulers(Optimus & Tiresias)
- Compared both in Makespan and average DL job completion time.
- Testbed
  - 16 AWS nodes w/ 64 GPUs (Nvidia T4, 4 each node)
  - 160 DL jobs submitted over 8 hours.
  - 48 CPUs, 192GB memory, and 900GB SSD.

Evaluation-Scheduling Vs Expert-tuned jobs

Policy	Avg job time	Makespan
Pollux (p = -1)	**0.76h**	**16h**
Optimus+Oracle+TunedJobs	1.5h	20h
Tiresias+TunedJobs	1.2h	24h

- 37-50% faster average training time in comparison to previous SOTA schedulers
- Note that Pollux is dynamic, compared to expert-configured jobs in competitors.

Evaluation - Statistical Efficiency

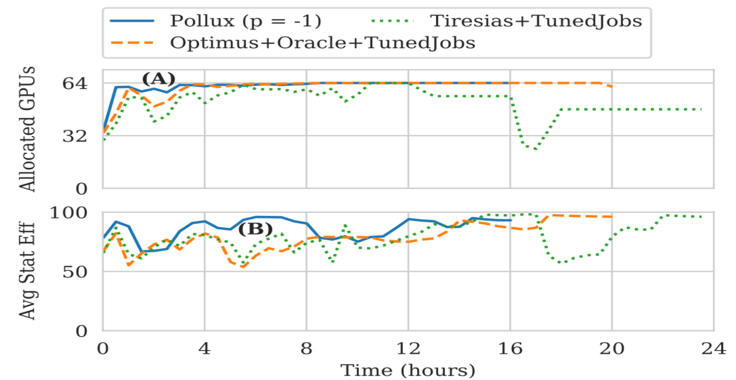

Figure 9: Pollux statistical cluster-wide efficiency [1]

- Reduces GPUs & batch size in high cluster contention(4-8)
- Accepts lower efficiency when we have lower cluster contention(8-12)

Conclusion, Opinion & Future Work

Conclusion.

- The paper presents Pollux, DL-aware cluster scheduler.
- Pollux co-optimizes both cluster-wide & per-job parameters for DL training.
- Pollux improves average DL training time in shared clusters by up to 50% even against very tuned baselines.
- Pollux co-adaptively allocates resources.

Opinion Of The Paper.

- On the positive side:
  - Novel idea that vastly improves DL jobs cluster scheduling.
  - Clear presentation of the problem and how Pollux solves it.
  - First work that considers GPU number, learning rate, and batch size dynamic job allocation in a cluster.
  - Best paper in a major conference.
- On the negative side:
  - No future work section; Some comments in the paper.
  - One note may be the usage of Pytorch and not Tensorflow which is more production ready; But this is debatable.

Future Work

- Tensorflow implementation for being more production ready.
- Usage of different, more capable GPUs.
- Comparison between own private cloud & public cloud such as the used AWS.
- Cloud auto-scaling system based on GOODPUT.
- Full evaluation on Pollux affects of different hyper-parameter algorithms.

Thank you!

Questions?

Additional resources

Terms

- Goodput: Measure for system throughput w.r.t statistical efficiency.

References

References i

A. Qiao, S. K. Choe, S. J. Subramanya, W. Neiswanger, Q. Ho, H. Zhang, G. R. Ganger, and E. P. Xing.
Pollux: Co-adaptive cluster scheduling for goodput-optimized deep learning.
In *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, pages 1–18. USENIX Association, July 2021.