Hi!

CS 744: POLLUX

Shivaram Venkataraman

Fall 2021

ADMINISTRIVIA

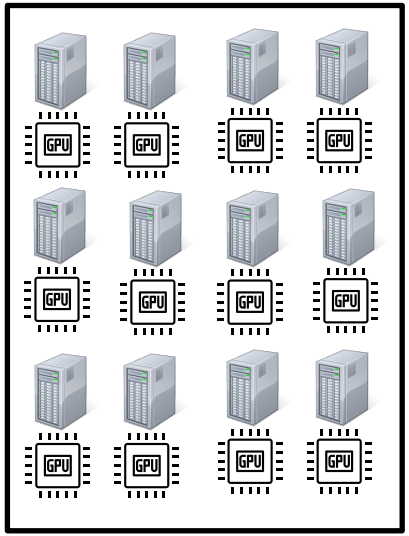
- Course project assignments
  - Emails will go out today (Oct 14th)
  - Introductions due Oct 25th → LaTex template

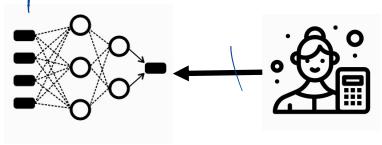
- Midterm Exam
  - In class on Oct 28th
  - Includes everything from beginning to the end of ML

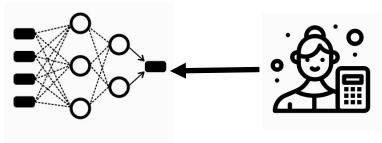
includes Pollux, Nexus

MACHINE LEARNING: TRAINING

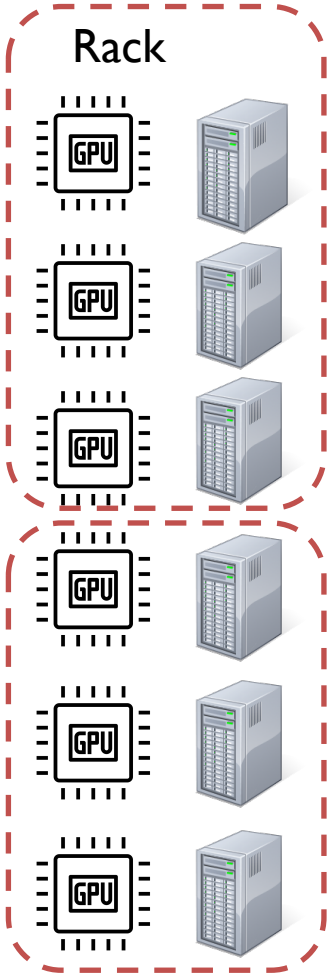
How do I parallelize this
What is the API







From one job training to multiple jobs!



WORKLOAD CHARACTERISTICS

150 epochs ~ few minutes ⇒ many hours

Long running tasks

all the processes for a job need to run at the same time

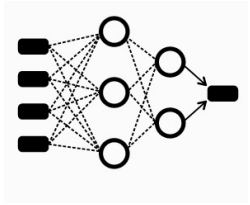
Gang scheduling

GPU Sharing?

Isolation primitives like containers to split cores, memory

EXISTING DL SCHEDULER INTERFACE

Job 1



Run job Resnet18
With BatchSize = 64
on Num GPUs = 4

Job 2

num gpus = 2

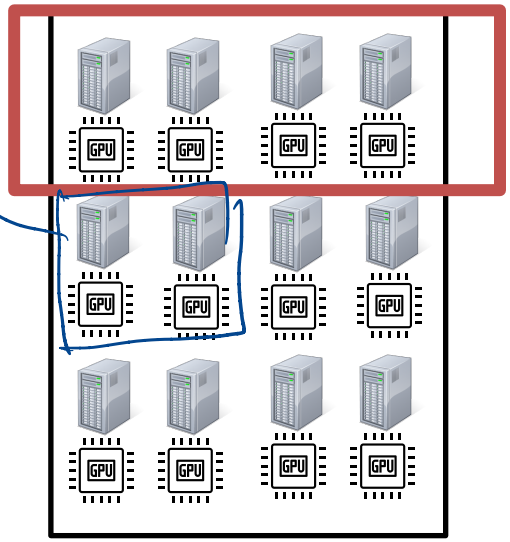
Scheduler

Goals:

- Maximize throughput
- Fairness
- Minimize JCT
- …

Job 2

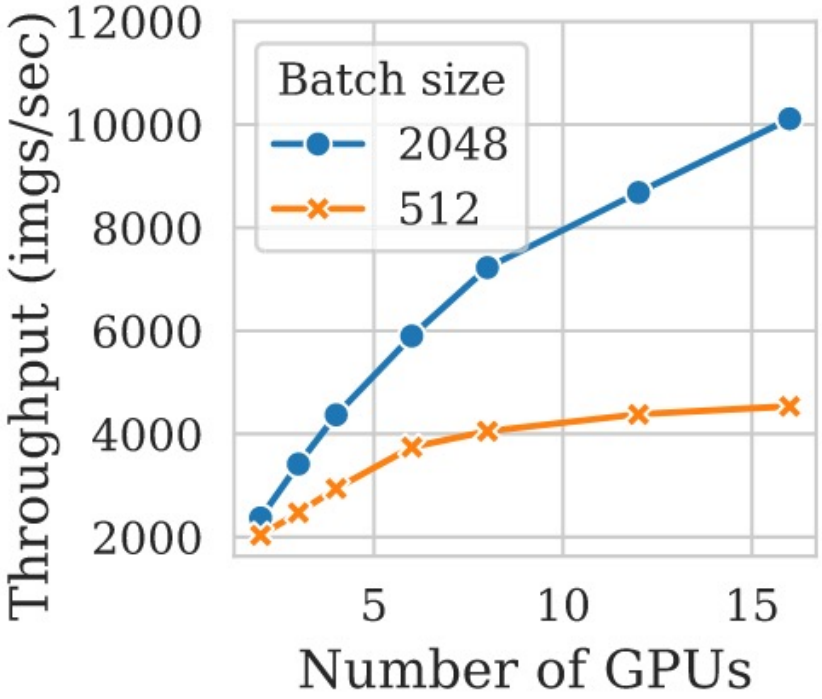
Job 1



2048/16 = 128

BATCH SIZE VS THROUGHPUT

ResNet18 on CIFAR-10



bs = 512 as you increase GPUs tput flattens out

tput = BS / T_{iter} =

$$Time_{iter} = T_{grad} + T_{sync}$$

Inc num GPUs T_{grad} ↓

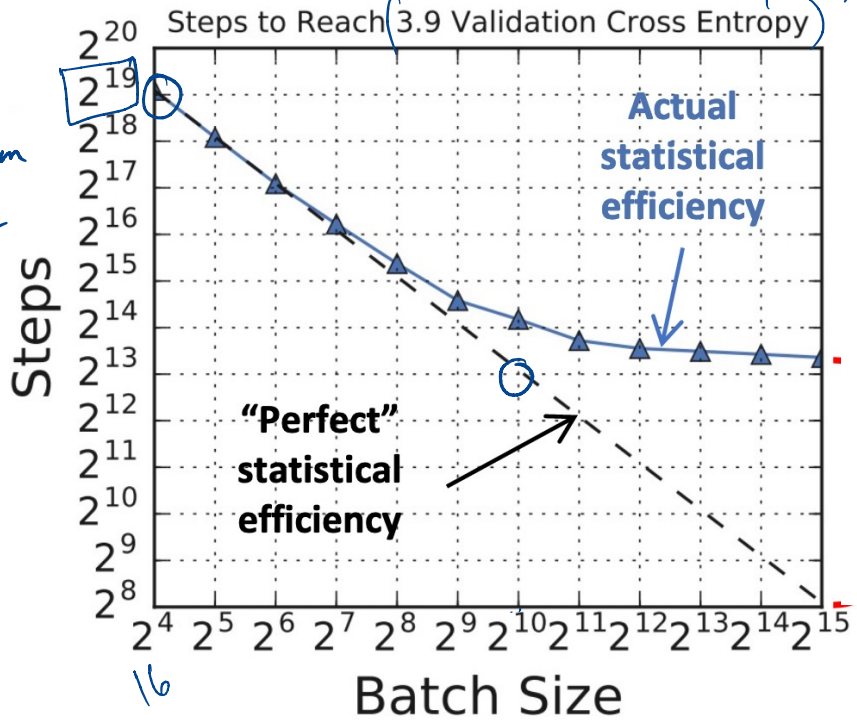
T_{sync} ↑

The way to get better tput with more GPUs = inc. batch size!

BATCH SIZE VS STATISTICAL EFFICIENCY

Statistical Efficiency: Amount of progress made *per training example*

GPU ✓ → parallelism in your compute



SGD, goal to minimize loss each iter "progress"

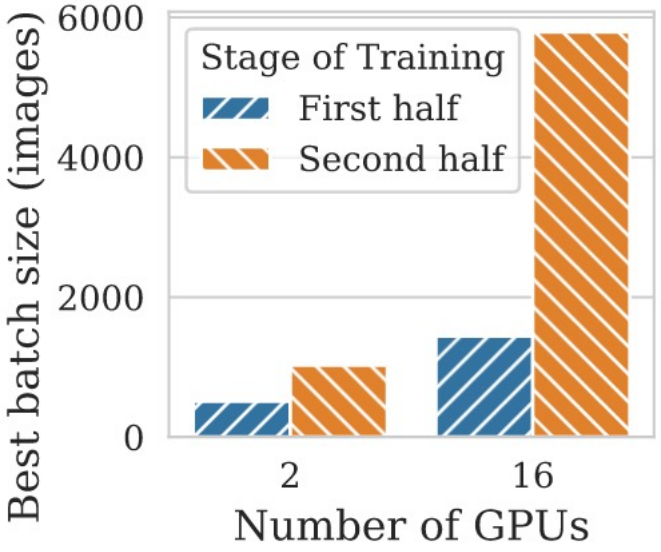
ideal scaling = double batch size then we take 1/2 num steps

---

larger batch sizes lead to lower statistical efficiency.

GRADIENT NOISE SCALE (GNS)

Scale batch size *during* training based on gradient noise



The optimal batch size can vary during training

GNS is correlated with optimal batch size (for stat. efficiency)

Initial iterations → smaller batch size

Towards end of training → larger batch size

POLLUX: GOAL, APPROACH

Maximize goodput across all jobs in the cluster → by the scheduler

$$\texttt{GOODPUT}_t(\star) = \texttt{THROUGHPUT}(\star) \times \texttt{EFFICIENCY}_t(M(\star)),$$

Approach

1. Job submitted with initial batch size, num GPUs
2. Profile jobs during execution to model throughput, efficiency
3. Tune batch size/GPU and num GPUs based on resource availability

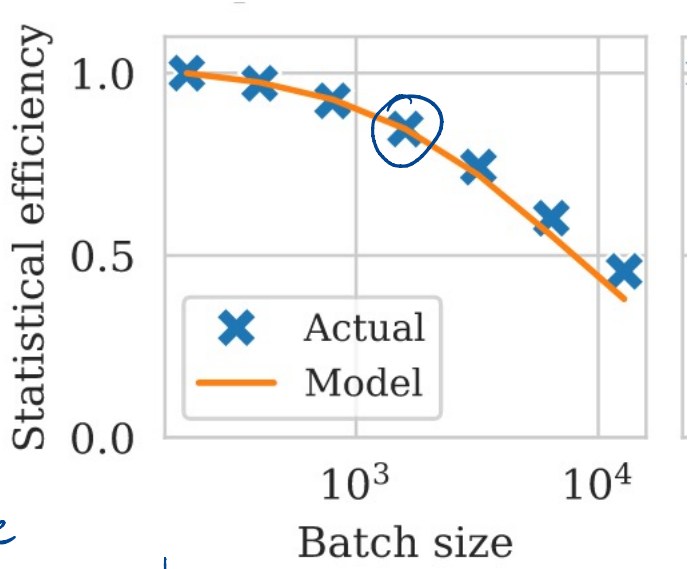
MODELING STATISTICAL EFFICIENCY

For a batch size M, initial batch size M_0

$$\text{EFFICIENCY}_t(M) = \frac{\varphi_t + M_0}{\varphi_t + M}.$$

t: iteration number

φ_t : Computed using gradients noise (GNS) using a **specific batch size**



Run the job using M' as batch size

Measure φ_t at iteration t using M'

Use formula above to infer eff. at other batch sizes

MODELING SYSTEM THROUGHPUT

num GPUs, batch size per GPU, grad. accu steps, batch size, time per iter

$$\text{THROUGHPUT}(a,m,s) = M(a,m,s)/T_{iter}(a,m,s).$$

fixed cost, per example cost

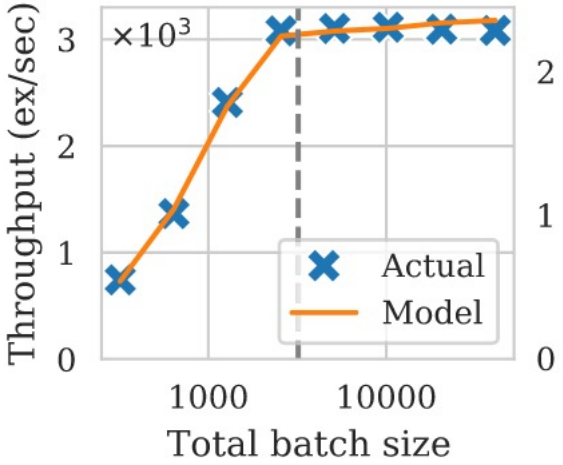
$$T_{grad}(m) = \alpha_{grad} + \beta_{grad} \cdot m,$$

$$T_{iter}(a,m,0) = \left(T_{grad}(a,m)^{\gamma} + T_{sync}(a)^{\gamma}\right)^{1/\gamma};$$

Account for overlap

$\gamma = 1$ → no overlap ⇒ $T_{grad} + T_{sync}$

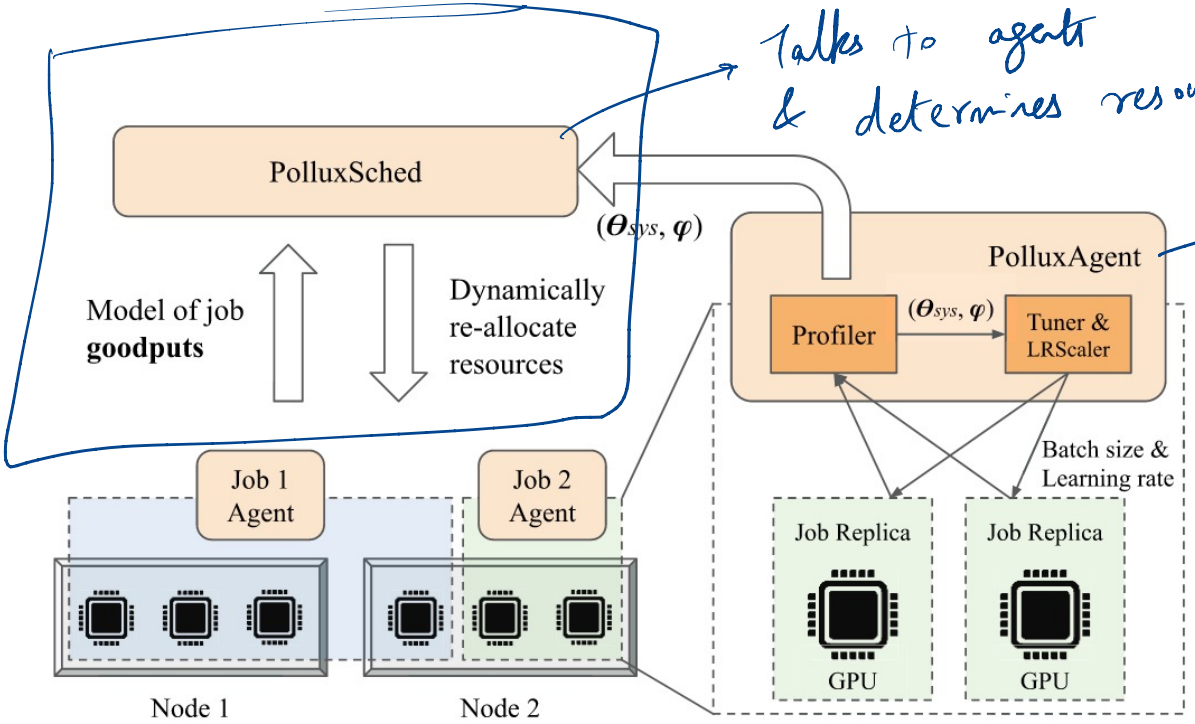
$\gamma = \infty$ → perfect overlap ⇒ $\max(T_{grad}, T_{sync})$



Takeaway : Learn gamma, alpha, beta etc. by running the job

Predict throughputs!

POLLUX: ARCHITECTURE



talks to agents & determines resources for each job

That manages every job

⇒ profiles job computes α, β, γ GNS

⇒ set batch size & LR for this job

POLLUX: CLUSTER WIDE OPTIMIZATION

→ optimization problem solved by the scheduler

Maximize fitness across all jobs

$$\texttt{FITNESS}_p(A) = \left(\frac{1}{J} \sum_{j=1}^{J} \texttt{SPEEDUP}_j(A_j)^p \right)^{1/p}.$$

J jobs in my queue

A_j = Allocation for this job

Solve to find A_j that will lead to max. fitness

p → fairness knob

p = 1 then fitness is just the sum

$p = -\infty$ minimum speedup ⇒ max-min fairness on SPEED UP

p = -1 used in eval.

Approach: Use a search algorithm (genetic algorithm) to find allocation

SUMMARY

DL Workloads Scheduling: Batch Size, Num GPUs

Pollux: Optimal batch size, num GPUs varies

Across jobs and during a job's execution

Models for system throughput, statistical efficiency

Cluster-wide optimization

DISCUSSION

https://forms.gle/hQTrk53W3wwkEu9A8

What are some similarities or differences between Mesos and DL schedulers like Pollux?

Similarity

- Goals like resource utilization
- Flexibility of workloads
  - ↳ Mesos: MPI, Spark
  - ↳ Pollux: only DL
    - short jobs,
    - long jobs.
- Decomposition
  - ↳ Job-level & Centralized

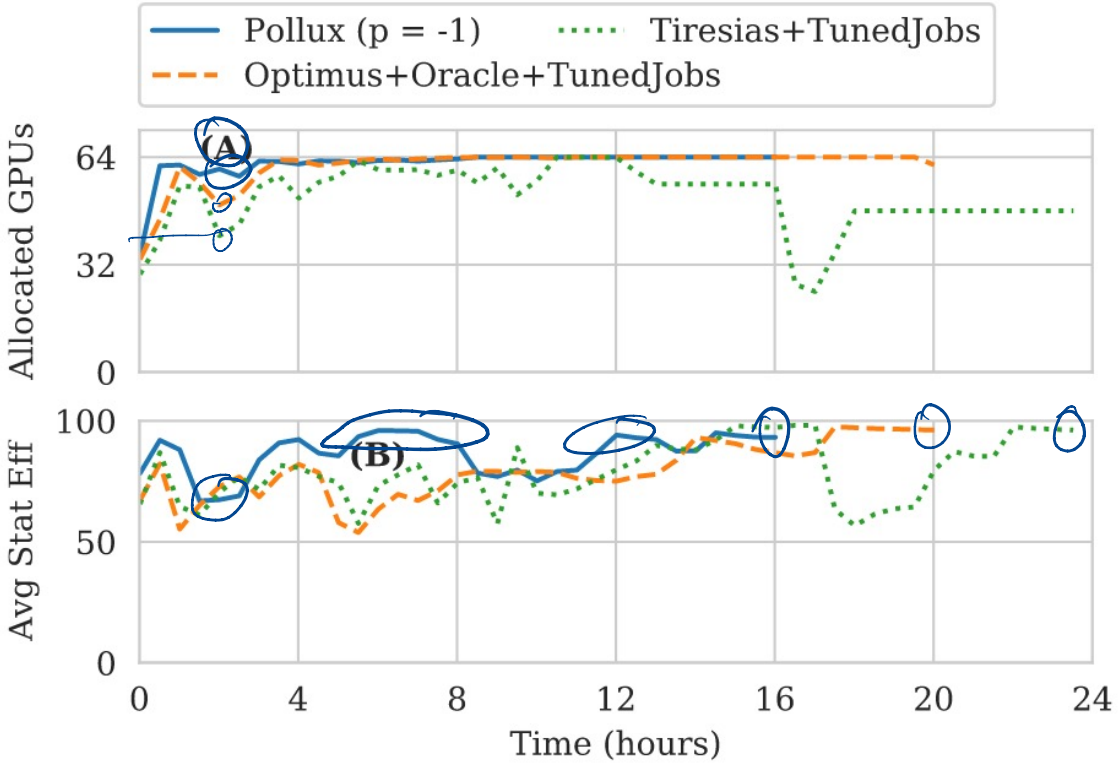
Diff

- Mesos only does allocation.

  Pollux allocation + Job config (batch size, LR)
- General API Mesos
  - ↳ rejecting offers to the job

  Pollux centralized sched. makes alloc.

At Point (A) demand is low ⇒ Pollux can adapt But statistical efficiency is not higher!



Pollux is finishing faster

higher statistical efficiency

NEXT STEPS

Next Class: Nexus

Course Project Introductions!

Midterm after that