Optimus: An Efficient Dynamic Resource Scheduler for Deep Learning Clusters

Yanghua Peng
The University of Hong Kong
yhpeng@cs.hku.hk

Yixin Bao
The University of Hong Kong
yxbao@cs.hku.hk

Yangrui Chen
The University of Hong Kong
yrchen@cs.hku.hk

Chuan Wu
The University of Hong Kong
cwu@cs.hku.hk

Chuanxiong Guo
Bytedance Inc.
guochuanxiong@bytedance.com

Presenter: Silvery Fu
What is the problem?

- What is the optimal resource allocation strategy for deep learning workloads?

Why? Improve training completion time and resource efficiency.

Specifically, avg. job completion time and makespan - across multiple jobs.

How? Leverage more application-level semantics.
What is the problem?

- What is the optimal resource allocation strategy for deep learning workloads?

- Why? Improve training completion **time** and resource **efficiency**.
  - specifically, avg. job completion time and makespan - across multiple jobs
What is the problem?

- What is the optimal resource allocation strategy for deep learning workloads?

- Why? Improve training completion time and resource efficiency.
  - specifically, avg. job completion time and makespan - across multiple jobs

- How? Leverage more application-level semantics.
Application semantics: DL

- **What semantics?**
  - DL's job-task model: parameter server tasks and worker tasks
  - JCT correlates w/ convergence, iterative, ...

![Diagram showing worker, parameter server, and data flow](image-url)
Application semantics: DL

● What semantics?
  ○ DL's job-task model: parameter server tasks and worker tasks
  ○ JCT correlates w/ convergence, iterative, ...

● How to capture these semantics?
  ○ Performance modeling
Application semantics: DL

● What semantics?
  ○ DL's job-task model: parameter server tasks and worker tasks
  ○ JCT correlates w/ convergence, iterative, ...

● How to capture these semantics?
  ○ Performance modeling

● How to use these semantics?
  ○ Tune knobs: num_worker and num_parameter_server
  ○ ..minimize makespan and average job completion time
Performance modeling

- Goal: predict the JCT for each DL training job
  - given the knobs: \texttt{num\_worker} and \texttt{num\_parameter\_server}
  - \[ \text{JCT} = F(\texttt{num\_worker}, \texttt{num\_parameter\_server}) \]
Performance modeling

● Goal: predict the JCT for each DL training job
  ○ given the knobs: num_worker and num_parameter_server
  ○ JCT = F(num_worker, num_parameter_server)

● Hypothesis about F? Use application semantics, e.g.,
  ○ JCT is the sum of the time to complete each training step (process one minibatch)
  ○ How many training steps remaining? Estimate how far away from convergence.
  ○ The duration of each training step breaks down to:
    ■ forward/back propagation time
    ■ data transfer time
    ■ ...
  ○ Express these JCT breakdowns in terms of num_worker and num_parameter_server
Performance modeling

- **Goal**: predict the JCT for each DL training job
  - given the knobs: `num_worker` and `num_parameter_server`
  - \( \text{JCT} = F(\text{num}_\text{worker}, \text{num}_\text{parameter}_\text{server}) \)

- **Hypothesis about **\( F **? Use application semantics, e.g.,
  - JCT is the sum of the time to complete each training step (process one minibatch)
  - How many training steps remaining? Estimate how far away from convergence.
  - The duration of each training step breaks down to:
    - forward/back propagation time
    - data transfer time
    - ...
  - Express these JCT breakdowns in terms of `num_worker` and `num_parameter_server`

graybox vs. blackbox?
Performance modeling

- Curve 1: convergence curve
  - loss-based training convergence
  - $f$: completed epochs $\rightarrow$ training loss

\[ l = \frac{1}{\beta_0 \cdot k + \beta_1} + \beta_2 \]

- Curve 2: resource-learning speed curve
  - $f$: resource configuration $\rightarrow$ learning speed

Given the two curves:
- Optimus decides the numbers of parameter servers and workers for each job

Figure 5: Training loss curves for different DL jobs
Performance modeling

- Curve 1: convergence curve
  - loss-based training convergence
  - \( f: \text{completed epochs} \rightarrow \text{training loss} \)
  \[
  l = \frac{1}{\beta_0 \cdot k + \beta_1} + \beta_2
  \]

- Curve 2: resource-learning speed curve
  - \( f: \text{resource configuration} \rightarrow \text{learning speed} \)
  \[
  f(p, w) = \left( \theta_0 \cdot \frac{M}{w} + \theta_1 + \theta_2 \cdot \frac{w}{p} + \theta_3 \cdot w + \theta_4 \cdot p \right)^{-1}
  \]
Performance modeling

- **Curve 1: convergence curve**
  - loss-based training convergence
  - $f$: completed epochs $\rightarrow$ training loss

  \[
  l = \frac{1}{\beta_0 \cdot k + \beta_1 + \beta_2}
  \]

- **Curve 2: resource-learning speed curve**
  - $f$: resource configuration $\rightarrow$ learning speed

  \[
  f(p, w) = \left(\theta_0 \cdot \frac{M}{w} + \theta_1 + \theta_2 \cdot \frac{w}{p} + \theta_3 \cdot w + \theta_4 \cdot p\right)^{-1}
  \]

- Given the two curves:
  - Optimus decides the numbers of parameter servers and workers for each job
Performance modeling

- Curve 1: convergence curve
  - loss-based training convergence
    - \( f: \text{completed epochs} \to \text{training loss} \)

- Curve 2: resource-learning speed curve
  - \( f: \text{resource configuration} \to \text{learning speed} \)

- Given the two curves:
  - Optimus decides the numbers of parameter servers and workers for each job

\[
\begin{align*}
\text{minimize} \quad & \sum_{j \in J} t_j \\
\text{subject to:} \quad & t_j = \frac{Q_j}{f(p_j, w_j)} \\
& \sum_{j \in J} (w_j \cdot O_j^r + p_j \cdot N_j^r) \leq C_r \quad \forall \ r \in R \\
& p_j \in \mathbb{Z}^+, w_j \in \mathbb{Z}^+ \quad \forall \ j \in J
\end{align*}
\]

- Optimus decides the numbers of parameter servers and workers for each job

- learned

- online

- adapted
Task placement

- Given the number of workers and the number parameter servers, decide where to place them at each scheduling iteration.
Task placement

- Given the number of workers and the number parameter servers, decide where to place them at each scheduling iteration.

- Optimal placement should minimize the data transfer

The paper has a proof for this.
Put together:
Key results: avg. JCT and Makespan
Key results: cpu utilization

higher CPU usage

Normalized CPU Usage (%)

Time (h)

Optimus
DRF
Tetris
Prediction Accuracy vs. Performance

Figure 11: Performance comparison

Figure 15: Sensitivity to prediction errors
Prediction Accuracy vs. Performance

How much accuracy do we need?

Figure 11: Performance comparison

Figure 15: Sensitivity to prediction errors
Prediction Accuracy vs. Performance

how much accuracy do we need?

Figure 11: Performance comparison

Figure 15: Sensitivity to prediction errors
Prediction Accuracy vs. Performance

how "cheap" is accuracy?

how much accuracy do we need?

Figure 8: Estimation errors of training speeds

Figure 15: Sensitivity to prediction errors
What are the metrics of success?

- Resource efficiency
  - makespan

- Training time
  - average job completion time

- Others:
  - scalability: scheduling overhead, scaling overhead
  - easy adoption
Long-term impact

Deep Learning

• Increasing deep learning workloads in production clusters
  ▪ Speech recognition
  ▪ Object classification
  ▪ Machine translation

• Many machine learning frameworks
  ▪ TensorFlow
  ▪ MXNet
  ▪ PaddlePaddle
Discussion

- "..Optimus has a relatively **small configuration space** (i.e., the number of tasks) and **5-10 sample** runs are enough for fitting the performance model quite accurately."
  - is this a useful assumption in practice? (each container fixed resources, horizontal scaling)
Discussion

- "..Optimus has a relatively **small configuration space** (i.e., the number of tasks) and **5-10 sample** runs are enough for fitting the performance model quite accurately."
  - is this a useful assumption in practice? (each container fixed resources, horizontal scaling)

- Who will use this? Who will run this?
  - Application user? Cluster operators?
Discussion

● "..Optimus has a relatively small configuration space (i.e., the number of tasks) and 5-10 sample runs are enough for fitting the performance model quite accurately."
  ○ is this a useful assumption in practice? (each container fixed resources, horizontal scaling)

● Who will use this? Who will run this?
  ○ Application user? Cluster operators?

● Graybox vs. blackbox
  ○ can we extract the "formula" from offline profiling for a given type of application?

\[ f(p, w) = (\theta_0 \cdot \frac{M}{w} + \theta_1 + \theta_2 \cdot \frac{w}{p} + \theta_3 \cdot w + \theta_4 \cdot p)^{-1} \]