Performance Analysis of Large Scale Deep Learning Systems

January 15, 2017 – July 14, 2017

- **Project Information**
  - Project title: **Performance Analysis of Large Scale Deep Learning Systems**
  - PI: William Gropp, University of Illinois Urbana-Champaign
  - Collaborators: Roy Campbell, Sayed Hadi Hashemi, University of Illinois Urbana-Champaign
  - Contact: wgropp@illinois.edu
- **Executive summary (150 words)**

In recent years, there has been a substantial growth of interest in neural networks with many layers usually referred to as Deep Learning. It has been observed that increasing the number of training data and the number of parameters of these models can improve their accuracy significantly. This observation led to huge interest in large-scale training of these models. However, existing distributed implementations of the deep learning training process lacks efficiency across a large set of machines, limiting their scalability. The efficiency loss is caused by the high overhead of message passing between multiple machines as well as CPU/GPU data transfer within nodes. We have built TensorFlow on Blue Waters, enhanced the runtime tracing, and benchmarked two different communication steps.

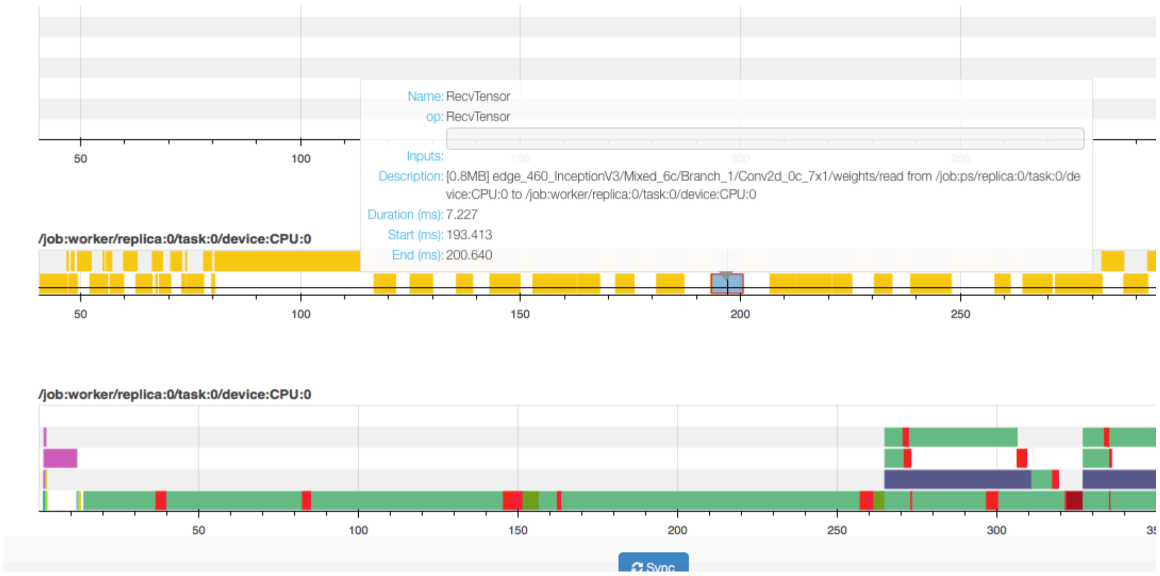
- **Description of research activities and results**
  - *Key Challenges:* The most common deep learning methods do not parallelize well on distributed memory systems.
  - *Why it Matters*: Increasingly complex deep learning models as well as large data sets make training these models increasingly expensive. Improved parallelization can dramatically reduce training times, enabling faster generation and evaluation of different approaches.
  - *Why Blue Waters:* Blue Waters provides a good environment for conducting these tests because of the availability of both GPU and non-GPU nodes, a very fast I/O system, and sufficient numbers of nodes to permit testing at scale.
  - *Accomplishments:*

Our main contributions were:

1. Distributed TensorFlow (TF) on Blue Waters: We have successfully installed and deployed the latest version of TF on Blue Waters. Since Blue Waters has an old unsupported glibc, we ended up running TF in a container.
2. Distributed Runtime Tracing Tools: We have extended the built-in tracing capability in TensorFlow to record network transfer activities in a distributed fashion (This change has been merged to the TF codebase).

Furthermore, we have developed a visualization tool for distributed traces. The tool is publicly accessible. Figure 1 shows a sample of the visualization.

3. Parameter Server vs. In-Graph All-Reduce: We studied the performance impact of the synchronization method in model-replica training jobs. We implemented two all-reduce algorithms (bucket and halving-doubling) as an in-graph operation in TensorFlow. This makes the implementation independent of the underlying network. Figure 2 shows the result of our work. While Parameter Server (PS) exposes better performance with fewer workers, the All-reduce (AR) has better scalability for larger numbers of workers.
4. Partial Ordering of Network Transfers: We observed that the order of network transfers have a significant performance impact on distributed execution runtime. We have proposed an ordering algorithm and developed an ordering enforcing mechanism on TF. The result is up to 8x less variation in step time, and up to 68% reduction of step time.


Name: RecvTensor
op: RecvTensor
Inputs:
Description: [0.8MB] edge_460_InceptionV3/Mixed_6c/Branch_1/Conv2d_0c_7x1/weights/read from /job:ps/replica:0/task:0/device:CPU:0 to /job:worker/replica:0/task:0/device:CPU:0
Duration (ms): 7.227
Start (ms): 193.413
End (ms): 200.640

50
100
150
200
250
/job:worker/replica:0/task:0/device:CPU:0
/job:worker/replica:0/task:0/device:CPU:0
300
Sync

Figure 1: Example of tracing visualization.

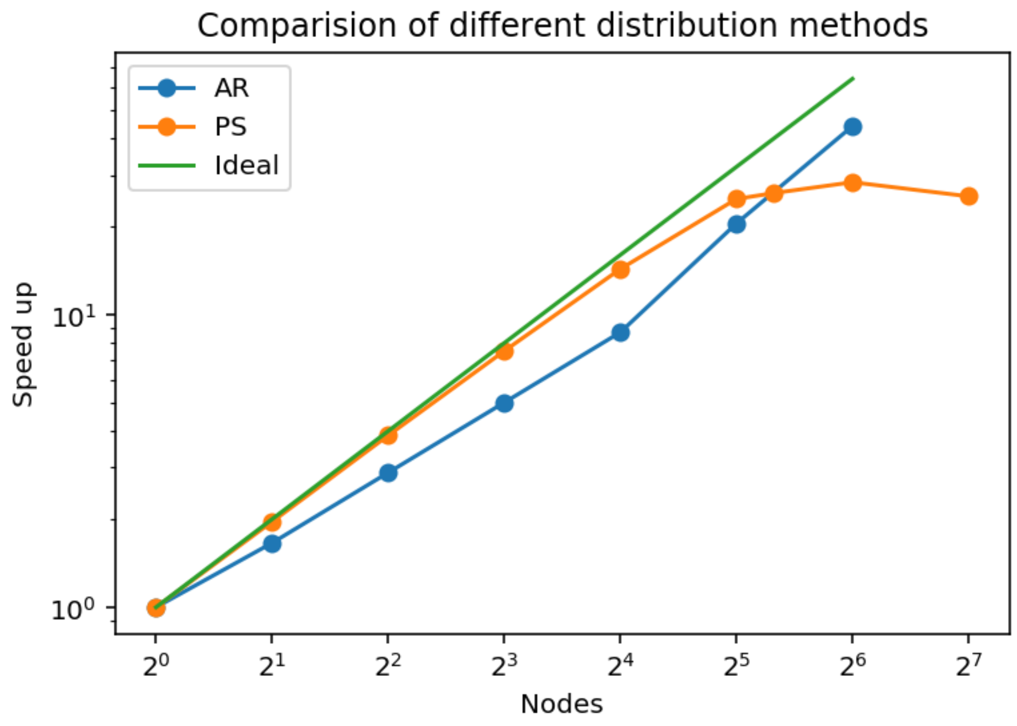


Figure 2: Comparison of scaling using parameter server (PS) and All-reduce (AR) methods

- **List of publications, data sets associated with this work**

Hashemi, Sayed Hadi, Sangeetha Abdu Jyothi, and Roy Campbell. "On The Importance of Execution Ordering in Graph-Based Distributed Machine Learning Systems." SysML 2018.

Hashemi, Sayed Hadi, Sangeetha Abdu Jyothi, and Roy Campbell. "Communication Scheduling as a First-Class Citizen in Distributed Machine Learning Systems." Submitted to USENIX ATC ’18 (under review).

Hashemi, Sayed Hadi, Sangeetha Abdu Jyothi, and Roy Campbell. Network Efficiency through Model-Awareness in Distributed Machine Learning Systems" Submitted to USENIX NSDI ’18 (under review).