Exploiting HPC for DL: Challenges, Overview of Solutions, and Need for Co-Design

Hari Subramoni
The Ohio State University
E-mail: subramon@cse.ohio-state.edu
http://www.cse.ohio-state.edu/~subramon
How to efficiently scale-out a Deep Learning (DL) framework and take advantage of heterogeneous High Performance Computing (HPC) resources?
Research Challenges to Exploit HPC Technologies

1. What are the fundamental issues in designing DL frameworks?
   - Memory Requirements
   - Computation Requirements
   - Communication Overhead

2. Why do we need to support distributed training?
   - To overcome the limits of single-node training
   - To better utilize hundreds of existing HPC Clusters
3. What are the new design challenges brought forward by DL frameworks for Communication runtimes?
   - Large Message Collective Communication and Reductions
   - GPU Buffers (CUDA-Awareness)

4. Can a Co-design approach help in achieving Scale-up and Scale-out efficiently?
   - Co-Design the support at Runtime level and Exploit it at the DL Framework level
   - What performance benefits can be observed?
   - What needs to be fixed at the communication runtime layer?
Solutions and Case Studies: Exploiting HPC for DL

- **NVIDIA NCCL**
- Baidu-allreduce
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL
NCCL2: Multi-node GPU Collectives

![Graph showing CNTK scaling for ResNet50, images/s. The graph compares Ideal, MPI, and NCCL performance with increasing number of nodes.]

Optimized designs in MVAPICH2-GDR 2.3rc1 offer better/comparable performance for most cases.

MPI_Allreduce (MVAPICH2-GDR) vs. ncclAllreduce (NCCL2) on 16 GPUs:

- MVAPICH2-GDR is ~1.2X better than NCCL2 for most message sizes.
- MVAPICH2-GDR is ~3X better than NCCL2 for larger message sizes.

Platform: Intel Xeon (Broadwell) nodes equipped with a dual-socket CPU, 1 K-80 GPUs, and EDR InfiniBand Inter-connect.
Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL
Baidu-allreduce in TensorFlow

- Baidu uses large message Allreduce collectives
- Evaluation with OpenMPI Allreduce showed performance degradation
- Proposed Solution:
  - Implement a Ring-Allreduce algorithm on top of point to point MPI primitives (Send/Recv) at the application level
- 2.5-3X better than OpenMPI Allreduce
- Used in the Deep Speech 2 paper*


Data Parallel Training with Baidu-allreduce

- Near-linear speedup for DNN training throughput (samples/second)
- The Allreduce design has been integrated in a TensorFlow contribution
- Details of the design are available from the Github site: https://github.com/baidu-research/tensorflow-allreduce

Courtesy: http://research.baidu.com/bringing-hpc-techniques-deep-learning/
MVAPICH2: Allreduce Comparison with Baidu and OpenMPI

- 16 GPUs (4 nodes) MVAPICH2-GDR vs. Baidu-Allreduce and OpenMPI 3.0

*Available with MVAPICH2-GDR 2.3a*
Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL

Deep Learning and Machine Learning Frameworks

- CNTK
- Caffe/OSU-Caffe
- Caffe2
- TensorFlow
- MXNet

Major Computation and Communication Phases in DL Frameworks

- Model Propagation
- Forward Backward
- Gradient Aggregation

Communication Runtimes (MPI/NCCL/Gloo/MLSL)

- Point-to-Point Operations
- CUDA-Awareness
- Large-message Collectives
  - Hierarchical Reduce (HR)
  - NCCL-Bcast/MPI_Bcast

HPC Platforms

- CPU
- InfiniBand
- GPU

Co-Design Opportunities

Network Based Computing Laboratory

5194.01
MPI+NCCL: Can we exploit NCCL to accelerate MPI?

- CUDA-Aware MPI provides excellent performance for small and medium message sizes
- NCCL has overhead for small messages but provides excellent performance for large messages
- Can we have designs that provide good performance for intra-node communication and inter-node scalability?
  - Exploit NCCL1 for intra-node inter-GPU communication
  - Design and utilize existing Inter-node communication in MVAPICH2-GDR

Application Performance with Microsoft CNTK (64 GPUs)

- Microsoft CNTK is a popular and efficient DL framework
- CA-CNTK is a CUDA-Aware version developed at OSU
- Proposed Broadcast provides up to 47% improvement in Training time for the VGG network
• NVIDIA NCCL
• Baidu-allreduce
• Facebook Gloo
• Co-design MPI runtimes and DL Frameworks
  – MPI+NCCL for CUDA-Aware CNTK
  – OSU-Caffe
• TensorFlow (Horovod)
• Scaling DNN Training on Multi-/Many-core CPUs
• PowerAI DDL
OSU-Caffe: Scalable Deep Learning

- Benefits and Weaknesses
  - Multi-GPU Training within a single node
  - Performance degradation for GPUs across different sockets
  - Limited Scale-out
- OSU-Caffe: MPI-based Parallel Training
  - Enable Scale-up (within a node) and Scale-out (across multi-GPU nodes)
  - Scale-out on 64 GPUs for training CIFAR-10 network on CIFAR-10 dataset
  - Scale-out on 128 GPUs for training GoogLeNet network on ImageNet dataset

OSU-Caffe publicly available from [http://hidl.cse.ohio-state.edu/](http://hidl.cse.ohio-state.edu/)
Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Facebook Gloo
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL
Baidu had a TensorFlow design that utilizes MPI library for gradient aggregation via a custom Allreduce design
  – Part of TensorFlow/contrib/
Uber has built Horovod inspired by Baidu’s approach but it provides a separate and easier installation process via pip
Horovod uses MPI_Allreduce or ncclAllreduce depending on the build process a user follows
TensorFusion optimization in Horovod to exploit efficient large message exchange
More details available from:
https://github.com/uber/horovod
• MVAPICH2-GDR offers excellent performance via advanced designs for MPI_Allreduce.
• Up to **22% better** performance on Wilkes2 cluster (16 GPUs)
Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Facebook Gloo
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL

Deep Learning and Machine Learning Frameworks

- CNTK
- Caffe/OSU-Caffe
- Caffe2
- TensorFlow
- MXNet

Major Computation and Communication Phases in DL Frameworks
- Model Propagation
- Forward Backward
- Gradient Aggregation

Communication Runtimes (MPI/NCCL/Gloo/MLSL)
- Point-to-Point Operations
- CUDA-Awareness
- Large-message Collectives
  - Hierarchical Reduce (HR)
  - NCCL-Bcast/MPI_Bcast

HPC Platforms
- CPU
- InfiniBand
- GPU
Optimizing and Scaling DL on Intel CPUs

![Image showing optimization and scaling of deep learning computations on Intel CPUs]

Courtesy: https://www.nextplatform.com/2016/06/21/knights-landing-solid-ground-intels-stake-deep-learning/
Optimizing NeuralTalk on Intel CPUs with Intel MKL

Optimization of NeuralTalk

Performance (images/s)

Original | Intel Compiler +MKL | Middleware Changes | User Code Changes | Parallel Strategy | MCDRAM as Cache

0.91 | 1.5 | 5.7 | 10 | 25 | 28

Intel® Xeon® processor E5-2650 v4 (2 sockets)
Intel® Xeon Phi™ processor 7210 (KNL)

Courtesy: https://colfaxresearch.com/isc16-neuraltalk/
TensorFlow Optimization for Intel CPUs

26x Speedup From New Optimizations – available through Google’s TensorFlow Git

Intel Machine Learning Scaling Library (MLSL)

- Intel MLSL is built on top of MPI primitives
  - [https://github.com/01org/MLSL](https://github.com/01org/MLSL)
- Works across various interconnects: Intel(R) Omni-Path Architecture, InfiniBand*, and Ethernet
- Common API to support Deep Learning frameworks (Caffe*, Theano*, Torch*, etc.)

<table>
<thead>
<tr>
<th>MLSL::Activation</th>
<th>A wrapper class for operation input and output activations</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLSL::CommBlockInfo</td>
<td>A class to hold block information for activations packing/unpacking</td>
</tr>
<tr>
<td>MLSL::Distribution</td>
<td>A class to hold the information about the parallelism scheme being used</td>
</tr>
<tr>
<td>MLSL::Environment</td>
<td>A singleton object that holds global Intel MLSL functions</td>
</tr>
<tr>
<td>MLSL::Operation</td>
<td>A class to hold information about learnable parameters (parameter sets) and activations corresponding to a certain operation of the computational graph</td>
</tr>
<tr>
<td>MLSL::OperationRegInfo</td>
<td>A class to hold Operation registration information</td>
</tr>
<tr>
<td>MLSL::ParameterSet</td>
<td>A wrapper class for operation parameters</td>
</tr>
<tr>
<td>MLSL::Session</td>
<td>A class to represent a collection of Operation objects with the same global mini-batch size</td>
</tr>
<tr>
<td>MLSL::Statistics</td>
<td>A class to measure and store performance statistics of communication among processes that perform computation in the computational graph</td>
</tr>
</tbody>
</table>

Courtesy: [https://github.com/01org/MLSL](https://github.com/01org/MLSL)
Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Facebook Gloo
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL
IBM PowerAI DDL

IBM PowerAI Platform

PowerAI Software Distribution

Deep Learning Frameworks
- Caffe
- NVIDIA Caffe
- IBM Caffe
- TensorFlow
- theano
- Chainer

Supporting Libraries
- DIGITS
- OpenBLAS
- Distributed Frameworks
- Bazel
- NCCL

IBM Power System for HPC, with NVLink
Breakthrough performance for GPU accelerated applications, including Deep Learning and Machine Learning.

PowerAI DDL Performance

Caffe with PowerAI DDL on ResNet-50 model using the ImageNet-1K data set on 64 Power8 servers

Courtesy: