Build GPU Cluster Hardware for Efficiently Accelerating CNN Training

YIN Jianxiong
Nanyang Technological University
jxyin@ntu.edu.sg
Visual Object Search

Private Large-scale Visual Object Database → Domain Specific Object Search Engine → Singapore Smart City Plan
Deep Learning Dedicated Cluster

2013
Servers w/ 2GPU
- 24x E5-2630
- 5x K20m
- 4x Titan Black
- 4x GTX 780

2014
4x 4GPU Server
- 8x E5-2630v2
- 16x K40

2015
8GPU Server Test Cluster
- 4x E5-2650v2
- 16x K40
- 4x K80
- 8x Titan X
- 36x IB FDR Switch

2016
Expanded 8GPU Server Cluster
- 12x E5-2650v2
- 16x K40
- 8x K80
- 8x M60
- 8x Titan X
- 36x FDR IB Switch
Parallel CNN Training is Important

CNN go deeper and complicated

Stopping Single Thread Performance increment
# Multi-node Parallel Training supportive Frameworks

<table>
<thead>
<tr>
<th>Training Algorithm</th>
<th>Multi-GPU in Single Server</th>
<th>Multi-GPU across Servers</th>
<th>Data Parallelism</th>
<th>*Model Parallelism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caffe</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>CNTK</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>MXNet</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Tensor Flow</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
Workflow and Job Handlers

- **CPU** prepares mini-batch
- Caches Mini-batch samples
- CPU Handle mini-Batch Preparation
- GPUs compute Forward Training
- AllReduce Loss Sync or Collect + Broadcast by GPU or CPU (Slower)
- GPUs compute Backward Training

Details:
- **Mini-Batch Preparation**
- **Mini-Batch Caching**
- **Mini-Batch Loading**
- **Forward**
- **Backward**
- **Synchronization**
Typical HPC architecture

Typical HPC Hardware

Typical HPC Cluster
GPU Optimal HPC Architecture

**GPU Optimal HPC Hardware**

- High GPU:CPU ratio (>2)
- High GPU density
- Direct P2P links
- Prioritized GPU Cooling

**Lower TCO Monetary Cost**
- Lower Maintenance Cost
- Better Performance
# Key Performance Impactors and Corresponding Hardware

<table>
<thead>
<tr>
<th></th>
<th>Data Parallelism</th>
<th>Model Parallelism</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sync Timing</strong></td>
<td>After every forward computing</td>
<td>After each layer is computed</td>
</tr>
<tr>
<td><strong>Sync Frequency</strong></td>
<td>Proportional to # of iterations</td>
<td>Proportional to # of layers x # of iterations</td>
</tr>
<tr>
<td><strong>Synced Msg body</strong></td>
<td>The trained model, Output Activation Volume of each layer</td>
<td></td>
</tr>
<tr>
<td><strong>Synced Msg size</strong></td>
<td>Mostly large payload, &gt; 1MB,</td>
<td>Very tiny payload, ~ 4KB</td>
</tr>
</tbody>
</table>

\[
P_n = \frac{n \times Q_{iter} \times P_1 \times T_{comp}}{n \times Q_{iter} \times T_{iter}}
\]

- Single GPU Computing Capacity
- Optimized Algo, e.g., cuDNN, fbFFT
- Feed in example#
- CNN Model Structure
- GPU Memory Size
- Sync Message Payload size
- Sync Frequency
- GPU-GPU Interconnect Topology
- Intra/inter-node GPU-GPU Link capacity
- Traffic Optimization
GPU Card
Must-have GPU features

Software Features
- W/ updated cuDNNv4 Support
- W/ GPUDirect P2P/RDMA
- W/ NVIDIA Collective Communication Library support

Hardware Features
- Single Precision Performance Matters
- Larger Read Cache preferred
- Maxwell cards strongly preferred over Kepler or other previous gen.

![Graph showing throughput comparison between K40 and Titan X]
Multi-GPU training reduces per Card VRAM footprint, enables large model training

- Hardware Setup:
  - 2x Intel Xeon E5-2650v2, 2x 8GT/s QPI
  - 192GB, 1TB SATA-III SSD
  - 8x NVIDIA GeForce Titan X
  - Mellanox ConnectX-3 Pro FDR IB adapter

- Software Setup:
  - Ubuntu 14.04.3
  - CUDA 7.5
  - GPU Driver: 352.39, 3.0-OFED.3.0.2.0.0.1
  - GDR, cuDNNv4 enabled

- Training Config:
  - NN: Inception-bn, batch size=128, lr_factor=0.94, lr=0.045, synchronized training, num_classes=1000
GPU Memory And System Throughput

• Larger GPU memory allows larger Batch Size, which pushes GPU to higher utilization.

MXNet
8x K80 Chip,
Batch Size=512
Model: VggNet

MXNet
8x K80 Chip,
Batch Size=256
Model: VggNet
GPU Throughput And Scalability

K40 (12GB GDDR5, 384bit)  

Titan X (12GB GDDR5, 384bit)

Tesla M40 / Quadro M6000, 24GB VRAM
GPU Form Factors and Availability

Right GPU Cooler for Multi-GPU Setup

When identical in specs, passive cooling cards are:
1) less inter-card heat interference, increases stableness
2) Lower operation temperature
3) no Clock speed throttling
4) Cooling failure resistant

Passive Cooling

Active Cooling
GPU-GPU Interconnect
When and How to Sync

Reducing Time

- Reduce Handling Processor
  - CPU handling
  - GPU handling
  - Hybrid handling

Sync Payload Size

- >1MB Large
- <4KB Small
- Traffic Optimize

Sync Schedule & Rules

- Asynchronous
- Synchronous
- Frequent Sync
- Long Interval Sync
- Hybrid Sync

Transfer Overhead

- Mainboard IO layout
- Bandwidth
- Latency

Data Path Capacities

- CPU handling
- GPU handling
- Hybrid handling

Hardware

Software

Software

Hardware
CPU vs GPU Reducer

When D2D and H2D/D2H bandwidth is similar,

Overhead Of
CPU Reduce-Broadcast

Overhead Of LARC
AllReduce by CPU+GPUs

Overhead Of LARD
AllReduce by GPUs

In practise, $\text{BW}_{\text{H2D/D2H}} \approx 1.5 \times \text{BW}_{\text{D2D}}$

Reduce Handler
Benchmarking w/ K40 GPU
Extend System to Multiple Nodes

Key Issues:

Inter-Node Links only have:
1) much smaller link BW (6GB/s),
2) longer latency (up to 10x more than PCI-E link),
3) limited interface BW to system, thus suffers from low intra-node BW to peer GPU cards.

Pipeline Compute & Sync:
- Sacrifice Immediate Consistency
- Hurts Accuracy

Traffic Optimization
- Local Aggregation
- Message Data Compression

H/W Arch Innovation
- Faster Inter-node links
- Bottleneck-removing P2P alternatives
Intra-node P2P Topologies

Topo-1

Topo-2

Topo-3

Topo-4

Examples per second vs. Mini-Batch Size

Topo-1 vs. Topo-2

Topo-3 vs. Topo-4
Both Link Bandwidth and Latency Matter for Data Parallelism

HW Setup: 2x Intel Xeon E5-2650v2, 2x 8GT/s QPI 192GB, 1TB SATA-III SSD, NVIDIA Tesla K40, Mellanox ConnectX-3 Pro FDR IB adapter,
Training Config: NN: Inception-bn, batch size=128, lr_factor=0.94, lr=0.045, synchronized training, num_classes=1000

Typical Root Node Traffic over IB (caffe variation)
NUMA based or Chassis based

**NUMA-based**

**Chassis-based**

**H/W Setup:**
- 2x Intel Xeon E5-2650v2, 2x 8GT/s QPI
- 192GB, 1TB SATA-III SSD
- 8x NVIDIA Tesla K40
- Mellanox ConnectX-3 Pro FDR IB adapter

**S/W Setup:**
- Ubuntu 14.04.3, CUDA 7.5, GPU Driver: 352.39, 3.0-OFED.3.0.2.0.0.1
- GPUDirect P2P/RDMA, cuDNNv4 enabled

**Training Config:**
- NN: Inception-bn, batch size=128, lr_factor=0.94, lr=0.045
- synchronized training, num_classes=1000
Other Major Components
The length above are not proportional to the actual time spent on each sub-task. The actual time cost depends on structure of the model.

*Phase#3 handling is implementation dependent, e.g., MXNET supports both GPU and CPU
## CPU Selection

<table>
<thead>
<tr>
<th>Specs</th>
<th>Empirical Min Suggestions</th>
<th>Elaborations</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU series</td>
<td>Xeon or Core does not matter,</td>
<td>Xeon Processor preferred because it has better MB.</td>
</tr>
<tr>
<td>CPU micro architecture</td>
<td>Ivy Bridge EP or higher architecture processor, Sandy Bridge CPU not recommended[1].</td>
<td>Sandy Bridge Processor's PHB throughput is capped to 0.8GB/s to 5.56GB/s for small and large message respectively.</td>
</tr>
<tr>
<td>PCIe lane</td>
<td>CPU with 40 PCI-E lanes</td>
<td>Even for EP architect processors, there is still low PCIe lane model.</td>
</tr>
<tr>
<td>CPU core</td>
<td>2x CPU threads for each GPU chip</td>
<td>CPU mainly handles IO requests, and decoding of batch sample.</td>
</tr>
<tr>
<td>CPU clock</td>
<td>Base clock 2.4GHz, Up to 3.2GHz</td>
<td>IO handling overhead could be hidden using pipeline, CPU clock effects</td>
</tr>
</tbody>
</table>

Host RAM & Storage Selection

Host RAM:
- **Total Amount:**
  - 2x Total VRAM size
- **Specs:**
  - DDR3-1066 and 1866 hardly makes difference
  - Coz H2D memory IO is hidden by pipeline

Storage: Implementation Dependent
- SSD is good for check-point saving (CPS) speed, which is fairly important for fault-tolerant
- NFS storage hurts CPS speed
- Local SATA Drive preferred save PCIe Lane resource

Save extra RAM budget, and invest on SSD, GPU card, better layout & cooling efficient servers.
Multi-GPU-friendly Server

Airflow Design

GPU is major power horse
Wrap-up
Design Goals and System Arch Mapping

**Requirement Input**
- Throughput
- Scalability
- Monetary

**Relevant Key Specs**
- Unit GPU Computing Capacity
- Node IO layout; Inter node link capacity;
- Per throughput $; Scale Up $;

**Major Metrics**
- GPU Processor Density
- GPU Architecture
- GPU Clock Speed
- GPU RAM Size
- GPU Mem BW
- PCIe Version
- PCIe 3.0 16x slot Qty
- PCIe 3.0 8x slot Qty
- PCIe Switch Layout
- Link Latency
- PCIe 3.0 16x slot Qty versus CPU socket Qty ratio
Wish list Architecture

<table>
<thead>
<tr>
<th>Scalability</th>
<th>Avoid Short Board Link; Partitioned Workload; Balanced Traffic;</th>
<th>• P2P Large BW &amp; low latency intra / inter node(s) link</th>
<th>NVLINK + EDR/XDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximize Performance</td>
<td>Max Processor Utilization</td>
<td>• Large high BW Dedicated Local GPU RAM</td>
<td>Pascal + HBM</td>
</tr>
<tr>
<td>Pipeline Throughput</td>
<td>Hide overhead and delay</td>
<td>• Decoupled to-GPU Storage IO and P2P Compute IO</td>
<td>NVLINK + PCIe/CAPI</td>
</tr>
</tbody>
</table>
Server Node Recommendations

Balanced

Max GPU Density

EDR Ready

80GB/s G2G Link

Balanced

Max GPU Density

EDR Ready

80GB/s G2G Link
ACKNOWLEDGEMENT

Credits:
Pradeep Gupta (NVIDIA), Project Manager;
Wang Xingxing (ROSE Lab, NTU, Singapore);
Xiao Bin (ex WeChat, Tencent),

Funding Agencies

Industry Partner

Open-source Project:
Q & A