DISTRIBUTED DEEP NEURAL NETWORK TRAINING: NCCL ON SUMMIT

Sylvain Jeaugey, NVIDIA
OPTIMIZED INTER-GPU COMMUNICATION

NCCL: NVIDIA Collective Communication Library
Communication library running on GPUs, for GPU buffers.

Binaries: https://developer.nvidia.com/nccl and in NGC containers
Source code: https://github.com/nvidia/nccl
Perf tests: https://github.com/nvidia/nccl-tests
INTER-GPU COMMUNICATION
Intra-node and Inter-node

**Within a system**
- Shared Mem, PCI, NVLink

**Between systems**
- Sockets, Infiniband
- Others (plugin)

<table>
<thead>
<tr>
<th>Effective bandwidth in GB/s</th>
<th>Between systems</th>
<th>Within a system</th>
</tr>
</thead>
<tbody>
<tr>
<td>6-9 QPI (shared memory)</td>
<td>1.2 10GbE, TCP/IP Sockets</td>
<td>6-9 QPI (shared memory)</td>
</tr>
<tr>
<td>9-12 PCI Express Gen3 x16 (P2P)</td>
<td>12 100Gb IB or RoCE, RDMA (IB verbs)</td>
<td>9-12 PCI Express Gen3 x16 (P2P)</td>
</tr>
<tr>
<td>42 NVLink/P9, V100</td>
<td>24 2x 100Gb (Summit)</td>
<td>42 NVLink/P9, V100</td>
</tr>
<tr>
<td>62 NVLink, P100 (P2P)</td>
<td>47 4x 100Gb (DGX1)</td>
<td>62 NVLink, P100 (P2P)</td>
</tr>
<tr>
<td>132 NVLink, V100 (P2P)</td>
<td>82 8x 100Gb (DGX2)</td>
<td>132 NVLink, V100 (P2P)</td>
</tr>
</tbody>
</table>
DEEP NEURAL NETWORK TRAINING
MULTI-GPU TRAINING

Single-GPU

Update

Forward/Backward

batch (e.g. 256 images)

parameters

gradients

Database : GBs of input data : images, sound, ...

parameters

gradients
MULTI-GPU TRAINING

Single-GPU

Update learn

Forward/Backward compute error

parameters

batch (e.g. 256 images)

gradients

Database : GBs of input data : images, sound, ...

compute error

learn

Forward/Backward

parameters
MULTI-GPU TRAINING

Data parallel

parameters

parameters

parameters

parameters

batch

batch

batch

batch

local gradients

local gradients

local gradients

local gradients

NCCL Allreduce: Sum gradients across GPUs

gradients

gradients

gradients

gradients

NCCL Allreduce: Sum gradients across GPUs
NCCL
DL stack

Frameworks (Tensorflow/Horovod, PyTorch, MXNet, Chainer, ...)

NCCL
CUDNN
CUBLAS
CUDA
NVIDIA GPUs
USER INTERFACE
NCCL API
Aims to cover most of MPI-1, plus fault tolerance

// Communicator creation
ncclGetUniqueId(ncclUniqueId* commId);
ncclCommInitRank(ncclComm_t* comm, int nranks, ncclUniqueId commId, int rank);
nncCommSplit(ncclComm_t* newcomm, int group, ncclComm_t comm);

// Communicator destruction / fault tolerance
ncclCommDestroy(ncclComm_t comm);
nncCommAbort(ncclComm_t comm);
nncCommGetAsyncError(ncclComm_t comm, ncclResult_t* asyncError);

// Collectives communication
ncclAllReduce(void* sbuff, void* rbuff, size_t count, ncclDataType_t type, ncclRedOp_t op,
            ncclComm_t comm, cudaStream_t stream);
nncBroadcast(void* sbuff, void* rbuff, size_t count, ncclDataType_t type,
            int root, ncclComm_t comm, cudaStream_t stream);
nncReduce(void* sbuff, void* rbuff, size_t count, ncclDataType_t type, ncclRedOp_t op,
           ncclComm_t comm, cudaStream_t stream);
nncReduceScatter(void* sbuff, void* rbuff, size_t count, ncclDataType_t type, ncclRedOp_t op,
                   ncclComm_t comm, cudaStream_t stream);
nncAllGather(void* sbuff, void* rbuff, size_t count, ncclDataType_t type,
           ncclComm_t comm, cudaStream_t stream);

// Point-to-point communication
ncclSend(void* sbuff, size_t count, ncclDataType_t type, int peer, ncclComm_t comm, cudaStream_t stream);
nncRecv(void* rbuff, size_t count, ncclDataType_t type, int peer, ncclComm_t comm, cudaStream_t stream);

// Aggregation/Composition
ncclGroupStart();
nncGroupEnd();

(functions in grey coming in future versions)
NCCL USAGE

Communicators and GPUs

One ncclComm_t handle = one NCCL rank = one GPU.
Fits any parallel model: multi-process, multi-thread, multi-GPU and any combination.

Creating multiple communicators is discouraged
No guaranteed concurrent progress of operations in CUDA, might lead to deadlocks. See ncclCommSplit.
NCCL CUDA Kernel
- runs on the GPU
- receives and sends from other peers through internal FIFOs
- perform reductions with local buffers, or copies
INTER-GPU COMMUNICATION

Inter-node

Input Buffer

Output Buffer

FIFO

Reduction

Sockets or Infiniband

CPU send proxy thread (host memory)

CPU receive proxy thread (host memory)

Next GPU(s)

Previous GPU(s)
INTER-GPU COMMUNICATION
Inter-node, GPU Direct RDMA

CPU send proxy thread (host memory)

CPU receive proxy thread (host memory)
TOPOLOGY AWARENESS
Extensive support for all platforms

And many others:
ARM, Intel/AMD, Single RC, ...

PCI only
DGX-1
DGX-2

P9/4 V100
P9/6 V100
TOPOLOGY AWARENESS
Extensive support for all platforms

PCI only : 6-12 GB/s

DGX-1 : 48 GB/s

DGX-2 : 85 GB/s

P9/4 V100 : 24 GB/s

P9/6 V100 : 24 GB/s

Extensive topology detection system, multi-path search maximizing performance on all platforms
ALGORITHMS AND PROTOCOLS
For best latency and bandwidth

Point-to-point communication protocols
- LL (1): 25-50% bandwidth, 1us latency (since 2.1)
- LL128 (1): 95% bandwidth, 2us latency (since 2.5)
- Simple: 100% bandwidth, 6us latency (since 2.0)

AllReduce algorithms
- Tree (2): 95% Bandwidth, log latency (since 2.4)
- Ring: 100% bandwidth, linear latency (since 2.0)

(1) LL: low-latency protocol, sending self-flagged data. Receiver polls on data instead of a separate flag
(2) Tree: dual binary tree, arranged in a complementary way to achieve full bandwidth.
SUMMIT

Tree performance

nccl-tests allreduce - 4096 nodes, 6x V100, 2x IB EDR (Summit/Oak Ridge National Laboratory)
FUTURE
NCCL 2.6
SHARPv2 and Adaptive routing

Continuous performance improvement

Add support for accelerated network collectives e.g. SHARPv2 (not available on Summit)

Add support for adaptive routing (might need to set NCCL_IB_SL on Summit)

Preview available: https://github.com/nvidia/nccl/tree/v2.6
ncclCommSplit creates sub-communicators for:
- Hierarchical collectives
- Model parallelism

Multi-communicator topology search to ensure maximum concurrent bandwidth on all sub-communicators.

Shares the same resources with the parent communicator and can detect programming errors leading to deadlocks.
Add `ncclSend` and `ncclRecv` functions to enable Send / Receive, Scatter[v], Gather[v], Alltoall[v,w], neighbor collectives, ...

NCCL 2.8
Point-to-point communication
NCCL
Summary

Optimized inter-GPU communication for DL and HPC
- Optimized for all NVIDIA platforms, most OEMs and Cloud
- Scales to 10,000s of GPUs.

Aims at covering all communication needs for multi-GPU computing.

Only relies on CUDA. No dependency on MPI or any parallel environment.

Binaries: https://developer.nvidia.com/nccl and in NGC containers
Source code: https://github.com/nvidia/nccl
Perf tests: https://github.com/nvidia/nccl-tests