SHARP: In-Network Scalable Hierarchical Aggregation and Reduction Protocol

Gil Bloch | Lugano

April, 2019
Unleashing the Power of Data

Exponential Growth of Data & Real Time Analytics Requires Mellanox High Throughput, Low Latency, Smart Interconnect Solutions
Mellanox Accelerates Leading HPC and AI Systems
World’s Top 3 Supercomputers

1. Summit CORAL System
   World’s Fastest HPC / AI System
   9.2K InfiniBand Nodes

2. Sierra CORAL System
   #2 USA Supercomputer
   8.6K InfiniBand Nodes

3. Wuxi Supercomputing Center
   Fastest Supercomputer in China
   41K InfiniBand Nodes
The Need for Intelligent and Faster Interconnect

Faster Data Speeds and In-Network Computing Enable Higher Performance and Scale

CPU-Centric (Onload)

Must Wait for the Data Creates Performance Bottlenecks

Data-Centric (Offload)

Analyze Data as it Moves! Higher Performance and Scale
In-Network Computing to Enable Data-Centric Data Centers

- MPI
- SHARP
- SHIELD
- GPUDirect
- NVMe Over Fabrics
- MultiHOST SocketDirect

Scalable Hierarchical Aggregation and Reduction Protocol
Accelerating All Levels of HPC/AI Frameworks

**Application Framework**
- Data Analysis
- Configurable Logic

**Communication Framework**
- SHARP – Data Aggregation
- MPI Tag Matching
- MPI Rendezvous
- SNAP - Software Defined Virtual Devices

**Network Framework**
- Network Transport Offload
- RDMA and GPU-Direct
- SHIELD (Self-Healing Network)
- Adaptive Routing and Congestion Control

**Connectivity Framework**
- Multi-Host
- Enhanced Topologies
- Dragonfly+
Accelerating HPC and AI Applications

Accelerating HPC Applications

- Significantly reduce MPI collective runtime
- Increase CPU availability and efficiency
- Enable communication and computation overlap

Enabling Artificial Intelligence Solutions to Perform Critical and Timely Decision Making

- Accelerating distributed machine learning
AllReduce Example – Trees

- Many2One and One2Many traffic patterns – possible network congestion
- Probably not a good solution for large data
- Large scale requires higher tree / larger radix
- Result distribution – over the tree / MC
AllReduce (Example) - Recursive Doubling

- The data is recursively divided, processed by CPUs and distributed
- The rank’s CPUs are occupied performing the reduce algorithm
- The data is sent at least 2x times, consumes at least twice the BW
Which Offload Should We Suggest?

- Switches can aggregate the data while it is going through the network...
  - Reducing the amount of data moving through the network
  - It will reduce the latency because data will go through a shorter path
  - Free up the CPU/GPU - the operation will be fully offloaded
Scalable Hierarchical Aggregation Protocol

Reliable Scalable General Purpose Primitive, Applicable to Multiple Use-cases

- In-network Tree based aggregation mechanism
- Large number of groups
- Multiple simultaneous outstanding operations
- Streaming aggregation

Accelerating HPC applications

- Scalable High Performance Collective Offload
  - Barrier, Reduce, All-Reduce, Broadcast
  - Sum, Min, Max, Min-loc, max-loc, OR, XOR, AND
  - Integer and Floating-Point, 16 / 32 / 64 bit
  - Up to 1KB payload size (in Quantum)
- Significantly reduce MPI collective runtime
- Increase CPU availability and efficiency
- Enable communication and computation overlap

Accelerating Machine Learning applications

- Prevent the many-to-one Traffic Pattern
- CUDA, GPUDirect RDMA
HCOLL: SHARP vs No-SHARP

Step 1 - Recursive Doubling

Step 2 - SHARP
SHARP AllReduce Performance Advantages (128 Nodes)

SHARP enables 75% Reduction in Latency
Providing Scalable Flat Latency

Scalable Hierarchical Aggregation and Reduction Protocol

SHARP - 8B
SHARP - 128B
Software - 8B
Software - 128B

SHARP - 1024B
SHARP - 2048B
Software - 1024B
Software - 2048B
SHARP AllReduce Performance Advantages
1500 Nodes, 60K MPI Ranks, Dragonfly+ Topology

SHARP Enables Highest Performance
Scalable Hierarchical Aggregation Protocol

- **SHARP Tree is a Logical Construct**
  - Nodes in the SHArP Tree are IB Endnodes
  - Logical tree defined on top of the physical underlying fabric
  - SHArP Tree Links are implemented on top of the IB transport (Reliable Connection)
  - Expected to follow the physical topology for performance but not required

- **SHARP Operations are Executed by a SHARP Tree**
  - Multiple SHArP Trees are Supported
  - Each SHArP Tree can handle Multiple Outstanding SHArP Operations
  - Within a SHArP Tree, each Operation is Uniquely Identified by a SHArP-Tuple
    - GroupID
    - SequenceNumber
SHARP Principles of Operation - Request

Aggregation Request

SHARP Tree Root
SHARP Principles of Operation – Response

Aggregation Response
GPU Direct™ RDMA

- Network adapter can directly read data from GPU device memory
- Avoids copies through the host
- Eliminates CPU bandwidth and latency bottlenecks
- Uses remote direct memory access (RDMA) transfers between GPUs
- Resulting in significantly improved MPISendRecv efficiency between GPUs in remote nodes
- Fastest possible communication between GPU and other PCI-E devices
- Allows for better asynchronous communication
SHARP & GPUDirect Performance Advantage for AI

- TensorFlow Horovod running ResNet50 benchmark
- E5-2650V4, 12 cores @ 2.2GHz, 30M L2 cache, 9.6GT QPI, 256GB RAM: 16 x 16 GB DDR4
- P100 NVIDIA GPUs, ConnectX-6 HCA, IB Quantum Switch (EDR speed)
- RH 7.5, Mellanox OFED 4.4, HPC-X v2.3, TensorFlow v1.11, Horovod 0.15.0

---

**ResNet50 Performance**

- **Images per Second**
  - NCCL2 without SHARP
  - SHARP

**8 Nodes, 16 GPUs, InfiniBand**

- **12% Improvement**

---

**ResNet50 Performance**

- **Images per Second**
  - NCCL2 without SHARP
  - SHARP

**8 Nodes, 22 GPUs, InfiniBand**

- **16% Improvement**
SHARP SW Overview
Mellanox HPC-X™ Scalable HPC Software Toolkit

- Complete MPI, PGAS OpenSHMEM and UPC package
- Maximize application performance
- For commercial and open source applications
- Best out of the box experience
Mellanox HPC-X™ Scalable HPC Software Toolkit

- Allow fast and simple deployment of HPC libraries
  - Both Stable & Latest Beta are bundled
  - All libraries are pre-compiled
  - Includes scripts/module files to ease deployment

- Package Includes
  - OpenMPI / OpenSHMEM
  - BUPC (Berkeley UPC)
  - UCX
  - HCOLL (FCA)
  - SHARP
  - KNEM
    - Allows fast intra-node MPI communication for large messages
  - Profiling Tools
    - Libibprof
    - IPM
  - Standard Benchmarks
    - OSU
    - IMB
HPCX/SHARP SW architecture

- **HCOLL**
  - Optimized collective library
  - Easy to integrate with multiple MPIs (OpenMPI, MPICH, MVAPICH*)

- **Libsharp.so**
  - Implementation of low level sharp API

- **Libsharp_coll.so**
  - Implementation of high level sharp API for enabling sharp collectives for MPI
  - Uses low level libsharp.so API
  - Easy to integrate with multiple MPIs (OpenMPI, MPICH, MVAPICH*)

Diagram:

```
  MPI (OpenMPI) --> HCOLL (libhcoll) --> SHARP (libsharp/libsharp_coll) --> InfiniBand Network
```
SHARP Software Architecture
SHARP: Configuring Subnet Manager

- Edit the opensm.conf file.

- Set the parameter “sharp_enabled” to “2”.

- Run OpenSM with the configuration file.
  - % opensm -F <opensm configuration file> -B

- Verify that the Aggregation Nodes were activated by the OpenSM, run "ibnetdiscover".
  - For example:
    - vendid=0x0
    - devid=0xcf09
    - sysimmguid=0x7cfe900300a5a2a0
    - caguid=0x7cfe900300a5a2a8
    - Ca 1 "H-7cfe900300a5a2a8" # "Mellanox Technologies Aggregation Node"
    - [1](7cfe900300a5a2a8) "S-7cfe900300a5a2a0"[37] # lid 256 lmc 0 "MF0;sharp2:MSB7800/U1" lid 512 4xFDR
NVIDIA Collective Communication Library (NCCL)

- Used by HCOLL
- Used by many Deep Learning Frameworks (Tensorflow/Horovod, PyTorch, MXNet, Chainer, ...)

- NCCL 2.4
  - Hierarchical tree algorithm
NCCL Ring

- Simple
- Full (half?) Bandwidth
- Linear Latency
NCCL Tree

- Support added in NCCL-2.4
- Keep Intra-node chain
- Node leaders participate in tree
- Binary double tree
- Multiple rings -> Multiple trees
NCCL 2.4

Performance

**NCCL latency**

Allreduce, 8 bytes

- NCCL 2.4 – Trees
- NCCL 2.3 – Rings

**NCCL bandwidth**

Allreduce, 64MB

- NCCL 2.4 – Trees
- NCCL 2.3 – Rings

59656 - Distributed Training and Fast Inter-GPU Communication with NCCL

nccl-tests allreduce - 4096 nodes, 6x V100, 2x IB EDR (Summit/Oak Ridge National Laboratory)
NCCL SHARP
NCCL SHARP

- Collective network Plugin
- Replace Inter-node tree with SHARP Tree
- Keeps Intra-node ring
- Aggregation in network switch
- Streaming from GPU memory with GPU Direct RDMA
- 2x BW compared to NCCL-TREE

**SHARP Enables 2X Higher Data Throughput for NCCL**
NCCL-SHARP Performance – DL Training

System Configuration: (4) HPE Apollo 6500 systems configured with (8) NVIDIA Tesla V100 SXM2 16GB, (2) HPE DL360 Gen10 Intel Xeon-Gold 6134 (3.2 GHz/8-core/130 W) CPUs, (24) DDR4-2666 CAS-19-19-19 Registered Memory Modules, HPE 1.6 TB NVMe SFF (2.5") SSD, ConnectX-6 HCA, IB Quantum Switch (EDR speed), Ubuntu 16.04
Performance Example
**Setup**

- **4 nodes, 16 GPUs**
  - Intel(R) Xeon(R) Gold 6150 CPU @ 2.70GHz
  - Volta NVIDIA GPUs
  - InfiniBand ConnectX-6 HCA

- **Ubuntu-16.04, Mellanox OFED 4.5, HPC-X v2.3, TensorFlow v1.12, Horovod 0.15.2**

- **IB Quantum Switch (EDR speed)**

- **NCCL : 1 Ring, NVLink with in the Node.**

- **TensorFlow/Horovod running ResNet50 benchmark**

- **SHARP: Using 4 channels (4 ports) directly participating in SAT operation**
Allreduce - SHARP

SHARP Latency

SHARP Streaming aggregation

© 2019 Mellanox Technologies
Allreduce - GPU Direct & SHARP

**GPU Direct**

- **Message size**:
  - 4 to 16384
- **Latency (us)**:
  - 0 to 50

**GPU Direct & SHARP**

- **Message Size**:
  - 8388608 to 268435456
- **Latency (us)**:
  - 0 to 80000

**Graphs**

- **NCCL**
- **SHARP**

**Comparison**

10x improvement in latency for SHARP compared to NCCL.
Horovod – Resnet50

- NCCL: 88%
- SHARP: 95%
Thank You