NCCL AND LIBFABRIC: HIGH-PERFORMANCE NETWORKING FOR MACHINE LEARNING

Brian Barrett, Principal Engineer
Rashika Kheria, Software Development Engineer

Amazon Web Services

March, 2019
AWS recently announced our Elastic Fabric Adapter for HPC/ML workloads.

Discussion of EFA and Libfabric tomorrow morning, but for now:
- 100 Gbps ethernet network
- OS bypass with UD and custom reliable datagram protocols
- Libfabric primary programming interface

Support for both HPC-like and GPU instance types

How can customers best utilize our GPU instances for large scale training workloads?
SO NCCL?

- NVIDIA’s NCCL (NVIDIA Collective Communication Library) is becoming the middleware of choice for machine learning applications

- NCCL 2 focused on multi-node, multi-GPU training
  - TCP and OFED VERBS support included from NVIDIA
  - Support for external plug-ins

- AWS has built a Libfabric plug-in for NCCL2
NCCL PRIMER

NCCL uses rings to move data across all GPUs and perform reductions.

DGX-1: 4 unidirectional rings
PCle / QPI: 1 unidirectional ring

PCle: 1 ring
DGX: 4 rings
NCCL PRIMER

Inter-node communication using Sockets or Infiniband verbs, with multi-rail support, topology ... bandwidth and create rings across nodes.

Inter-node communication 
PCIe, Infiniband DGX-1: NVLink, 4x Infiniband

Extend to multi-node
NCCL PLUG-IN INTERFACE

- **Initialization / Cleanup**
  - Plug-in provides local host topology information
  - NCCL core handles GPU/NIC mapping
  - Functions for determining capabilities (such as ability to send to/from GPU buffers directly)

- **Listen/Connect/Accept interface for connections**
  - Can be multiple connections between any two processes
  - NCCL has a “ring” construct; each ring will have a set of connections

- **Explicit memory register / deregister APIs**

- **Non-blocking send/receive with Test**
  - Ordered message channel
  - Messages are potentially large
▪ Requires fi_nic support to provide PCI address / endpoint association
▪ Uses FI_EP_RDM endpoints with FI_ORDER_SAS and FI_TAGGED
  • Need multiple ordered channels between any two processes (multiple “rings” in NCCL terms)
  • Wanted to reduce pressure on hardware resources and avoid polling multiple completion queues
  • Scalable endpoints another possibility, but not as widely supported
▪ NCCL, like MPI, doesn’t have a backpressure concept, so implement a queue for messages that can’t be queued in provider
  • Some things never change 😊
▪ ~1500 lines of code, most of which is setup or queueing
▪ Available on GitHub:
  • https://github.com/aws/aws-ofi-nccl/
  • Apache 2.0 License
FUTURE WORK

- **GPUDirect support**
  - Can we keep most of the interface changes in the memory registration logic?
  - Obviously ignoring the kernel part of the GPUDirect discussion

- **Performance tuning**
  - AWS is still tuning the end to end stack for EFA, hard to tightly measure with so much in flight
  - Seeing big advantages for our infrastructure, but comparing TCP to OS Bypass, so…

- **Bug fixes**
  - Many “unknown unknowns.”
• Want to build community Libfabric plug-in for NCCL
• Need provider testing. We’ve tested with EFA and TCP providers, but there are many more
• Performance tuning
• GPUDirect development: let’s get a model everyone can use
THANK YOU

Brian Barrett

Amazon Web Services