High Performance Deep Learning In Containers

Bruce D’Amora (damora@us.ibm.com)
Khalid Ahmed (khalida@ca.ibm.com)
Evolving Technical Computing Landscape

Moving deep into problems in industry verticals
Increased dependency on Cognitive in workflow

Past

Application Performance

HPC Systems
BlueGene/Q

Today

• Workflow of Analytics, Cognitive, and Modeling & Simulation applications
• Distributed Deep Learning
• Heterogeneous compute

Future

• Improved Cognitive capabilities via Accelerators
• Seamless enablement of heterogeneous compute
• On premise and in the cloud

Exascale Systems

World’s First Fully Data Centric Systems
Sierra (LLNL), Summit ORNL

7X Performance

Early 2020s

50X Performance
IBM Research invents the jet engine of deep learning

The IBM Research team took on this challenge, and through innovative clustering methods has built a “Distributed Deep Learning” (DDL) library that hooks into popular open source machine learning frameworks like TensorFlow, Caffe, Torch and Chainer. DDL enables these frameworks to scale to tens of IBM servers leveraging hundreds of GPUs. Effectively, IBM Research has invented the jet engine of deep learning.

Building and Maintaining Technical Computing Workflows

Challenges to developing, building and maintaining technical computing applications, e.g. the PowerAI stack

- Software Dependencies
  - Many applications rely on open source libraries and toolkits
    - OpenMPI v. 1.10 vs 2.1 vs 2.2…etc
    - BLAS, Atlas, Lapack, ESSL, …. etc.
  - Library configuration options vary
    - OpenMPI with CUDA support or Infiniband support or specific scheduler support

- Build Environment varies
  - make vs Cmake vs Scons, etc.

- Runtime Dependencies
  - OS, kernel extensions/driver, e.g. which version of CUDA ?

- Reproducibility and extensibility
  - Can user B pick up where user A left off …. easily?

Resolving these dependencies usually requires sysadmin to install and/or configure application dependencies
Container Technology Offer Solutions

Docker is de facto industry standard

Developer specified dependencies in single Dockerfile; Less dependency on sysadmin

Image versioning and distribution mechanisms already supported

But ... challenges persist for technical computing community – Distributed computing problem

Shifter originated by NERSC provides framework for HPC use of Docker

Singularity provides its own image format
Can also import Docker image
Distributed Computing Requirements for Deep Learning

Message Passing Interface commonly used

- **Gather**
- **Scatter**
- **Broadcast**
- **Reduction**

**MPI support in containers requires access to high performance network, Infiniband**

**Containers require access to host network drivers and firmware**

**Container orchestration service, kubernetes, does not directly support MPI**
The MPI Environment

MPI runtime environment

- `mpirun` starts multiple instances of apps across a cluster
- `mpirun` requires list of host machines to schedule the multiple instances of apps
- User specifies how many instances of application should be scheduled

MPI runtime environment needs information from container orchestration service like kubernetes
Kubernetes Pod Scheduler

- Kubernetes master schedules pods on available nodes
- Single pod does not span nodes
- Each pod contains one or more Docker containers

**mpirun** needs to know which nodes Kubernetes scheduled PODs on
**mpirun** needs to be able to connect to those nodes (sshkeys)
**mpirun** needs to know the name of application and runtime parameters
Kubernetes and MPI Integration

Steps
1. Create a Docker base image
2. Create python-kube program that queries kubernetes for POD placement in cluster
3. Create yaml or json file to launch kubernetes job

Dockerfile
- ubuntu
- Install python,
- Install build-essential, openssh-server,
- install openssl.git, ...
- install kubernetes, python-kubernetes
- get openmpi-<y.z>
- configure openmpi
- including support for IB verbs, GPU
- make openmpi
- create ssh-key
- copy bootmpi to /usr/local/bin

bootmpi
- Import netifaces, netaddr, sockets, etc
- use python-kubernetes client to query names of PODs
- if HostNetwork,
  - Get the ip addr for each PODs allocated host
  - Figure out which net interface to use
- if POD Network, get the ip addr of each POD
- mpiexec is entry point for bootmpi

YAML file
- specify kubernetes Job
- specify completions, parallelism levels
- volume mount sshkey of host
- volume mount IB driver paths
- set command to bootmpi
- argument list:
  - network mask
  - job name
  - command
- set restartPolicy to Never
Demo

- Simple MPI application scheduled with kubernetes
- Test cluster has 1 master and 2 Power8 Minsky nodes
  - `kubectl create -f MPI-pi.yaml`
  - `kubectl get pods`
  - `kubectl logs -f <pod name>`
Summary and Next Steps

- MPI required for Distributed Deep Learning
- Container Technology facilitates development, maintenance and deployment of technical computing applications
- MPI support required for containerized distributed deep learning
- Additional work to support
  - topological scheduling
    - controlling how PODS are mapped to nodes in cluster
  - affinity of MPI processes to specific processors
  - expose many other MPI runtime options not available in kubernetes
DevOps/Microservices & Deep Learning
Deep Learning Workflows in Context

- Scheduler controls job start and placement
- Supports both on-line and batch processing
- Applications exchange data as needed
  - Producers
  - Consumers
  - Both
- Remote users receive/provide feedback
- Integrate with Enterprise Applications
Example: Run Tensorflow Pipeline In Kubernetes

- Docker image of Tensorflow running environment
- Kubernetes BatchJob to manage Tensorflow training job lifecycle
- Kubernetes Volume to share the data
- Kubernetes Deployment/Service to provide Tensorflow serving service
- Kubernetes platform to work with other services and resources
- Kubernetes platform for general data center platform

![Diagram of Tensorflow Pipeline in Kubernetes](image-url)
GPU Support & Batch Are Important!

- Auto-discovery GPU resources
- GPU scheduling
- Monitor GPU resource utilization
- GPU driver injection

Turbocharging Kubernetes Batch with Spectrum LSF [http://ibm.biz/BdjZ8s](http://ibm.biz/BdjZ8s)
Extend the Pipeline to Iterative Development

- Kubernetes Deployment/Service for rolling upgrade
- Integrate with CI/CD utilities
Broaden To Encompass Microservices

What else do we need for Microservices?

- Visibility
- Resiliency & Efficiency
- Traffic Control
- Security
- Policy Enforcement

Enter Istio

Intelligent Routing and Load Balancing
Fleet Wide Policy Enforcement
Resiliency across Languages and Platforms
In-Depth Telemetry and Reporting
A private cloud platform for enterprises to develop and run their workloads locally

An integrated platform consisting of Kubernetes and developer services necessary to create, run, and manage cloud applications

Support for Deep Learning, Batch Processing, GPUs, Microservices, CI/CD,…

Platform to deliver modernized IBM middleware and data services to enterprise customers

https://ibm.biz/cloud-private