GPU ACCELERATED MULTI-NODE HPC WORKLOADS WITH SINGULARITY

December 2018
AGENDA

What are containers?
Pulling containers
Running multi-node workloads
Building multi-node containers
WHAT ARE CONTAINERS?

- Isolation technology based on Linux kernel namespaces
- Package everything needed to run an application
- Differ from virtualization
  - Containers run on common kernel as host
  - OS virtualization vs hardware abstraction
  - Containers are generally more lightweight and offer better performance than VMs
- Container runtimes Charlie Cloud, Docker, Shifter, Singularity, and more
  - NGC HPC containers are QAed with Docker and Singularity
CONTAINER BENEFITS

- Enabling straddling of distros on a common Linux kernel
- Isolate environment and resources
- Encapsulate dependencies
- Straightforward deployment
- Drop in replacement for many workflows
- Promote reproducibility
- Equivalent performance to baremetal
BARE METAL VS CONTAINERS

**Drivers + Operating System**

**Bare Metal**
- NAMD 2.12
- VMD 1.9.3
- GROMACS 5.1
- NAMD 2.13

**Containers**
- NAMD 2.12 CUDA libraries
- VMD CUDA libraries
- GROMACS CUDA libraries
- NAMD 2.13 CUDA libraries

**CUDA Libraries**
- FFTW 3.2.1
- CUDA 9.0
- Open MPI 3.0.2
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MPI 3.1.0
- CUDA 9.2
- Open MP...
CONTAINER REGISTRIES

- **Docker Hub** - [https://hub.docker.com](https://hub.docker.com)
  - Official repositories for CentOS, Ubuntu, and more
  - NVIDIA: [https://hub.docker.com/r/nvidia/cuda](https://hub.docker.com/r/nvidia/cuda)

- **Singularity Hub** - [https://singularity-hub.org/](https://singularity-hub.org/)
  - Registry of scientific Linux containers

- **NVIDIA GPU Cloud (NGC)** - [https://ngc.nvidia.com](https://ngc.nvidia.com)
  - Optimized HPC, HPC Visualization, Deep Learning, and base containers
## NGC CONTAINER REGISTRY

**Over 40 containers available today**

<table>
<thead>
<tr>
<th>Deep Learning</th>
<th>HPC</th>
<th>HPC Visualization</th>
<th>RAPIDS/ML</th>
<th>NVIDIA/K8s</th>
<th>Partners</th>
</tr>
</thead>
<tbody>
<tr>
<td>caffe</td>
<td>bigdft</td>
<td>index</td>
<td>rapidsai</td>
<td>Kubernetes on NVIDIA GPUs</td>
<td>chainer</td>
</tr>
<tr>
<td>caffe2</td>
<td>candle</td>
<td>parview-holodeck</td>
<td></td>
<td></td>
<td>deep-learning-studio</td>
</tr>
<tr>
<td>cntk</td>
<td>chroma</td>
<td>parview-index</td>
<td></td>
<td></td>
<td>h20ai-driverless</td>
</tr>
<tr>
<td>cuda</td>
<td>gamess</td>
<td>parview-optix</td>
<td></td>
<td></td>
<td>kinetica</td>
</tr>
<tr>
<td>digits</td>
<td>gromacs</td>
<td>vmd</td>
<td></td>
<td></td>
<td>mapd</td>
</tr>
<tr>
<td>inferencetser</td>
<td>lammps</td>
<td></td>
<td></td>
<td></td>
<td>matlab</td>
</tr>
<tr>
<td>mxnet</td>
<td>lattice-microbes</td>
<td></td>
<td></td>
<td></td>
<td>paddlepaddle</td>
</tr>
<tr>
<td>pytorch</td>
<td>milc</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tensorflow</td>
<td>namd</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tensorflowrt</td>
<td>pgI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tensorflowserver</td>
<td>picongpu</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>theano</td>
<td>qmcpack</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>torch</td>
<td>relion</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MULTI-NODE
MPI implementations provide a job launcher, mpirun or mpiexec, that initializes and wires up distributed MPI ranks (i.e., processes) on a multi-node cluster.
MPIRUN + CONTAINERS

"Outside-in"

Container

MPI Library

MPI Runtime

SSH Server

Host OS

mpirun is invoked outside the container

"Inside-out"

Container

MPI Library

MPI Runtime

SSH Server

Host OS

mpirun is invoked inside the container
MPIRUN + CONTAINERS

• “Outside-in”
  • Fits in more “naturally” into the traditional HPC workflow (SSH keys, etc.)
  • `mpirun -hostfile hostfile -n 64 app`
    becomes
    `mpirun -hostfile hostfile -n 64 singularity run app.simg app`
  • Requires a compatible MPI runtime on the host

• “Inside-out”
  • Must insert SSH keys into the container image by some other mechanism
  • Must orchestrate the launch of containers on other hosts
  • Completely self-contained, no host MPI dependencies
MULTI-NODE OUTSIDE-IN MILC RUN

On the cluster

Get the sample dataset
$ mkdir $HOME/milc-dataset && cd $HOME/milc-dataset
$ wget http://denali.physics.indiana.edu/~sg/SC15_student_cluster_competition/benchmarks.tar
$ tar -xf benchmarks.tar

Pull MILC container from NGC
$ module load singularity
$ singularity build milc.simg docker://nvcr.io/hpc/milc:quda0.8-patch4Oct2017

Get a 2 node allocation

Run the container using 2 nodes with 4 GPUs per node
$ module load openmpi
$ mpirun -n 8 -npernode 4 -wdir $HOME/milc-dataset/small singularity run --nv ~/milc.simg
/milc/milc_qcd-7.8.1/ks_imp_rhmc/su3_rhmd_hisq -geom 1 1 2 4 small.bench.in
...

On the cluster
MULTI-NODE SLURM MILC RUN

On the cluster

Get the sample dataset
$ mkdir $HOME/milc-dataset && cd $HOME/milc-dataset
$ wget http://denali.physics.indiana.edu/~sg/SC15_student_cluster_competition/benchmarks.tar
$ tar -xf benchmarks.tar

Pull MILC container from NGC
$ module load singularity
$ singularity build milc.simg docker://nvcr.io/hpc/milc:quda0.8-patch4Oct2017

Run the container using 2 nodes with 8 GPUs per node
$ srun --nodes=2 --ntasks-per-node=8 --mpi=pmi2 singularity run --pwd $HOME/milc-dataset/small --nv milc.simg su3_rhmd_hisq -geom 1224 small.bench.in
GENERIC MULTI-NODE SLURM RUN

On the cluster

Pull container from NGC
$ module load singularity
$ singularity build myapp.simg docker://nvcr.io/hpc/myapp:tag

Run the container using 2 nodes with 8 GPUs per node
$ srun --nodes=2 --ntasks-per-node=8 --mpi=pmi2 singularity run --nv myapp.simg myapp
BUILDING MULTI-NODE CONTAINERS

- Know your target hardware and software configurations
  - If possible, build on your target hardware
- Use multi stage builds to minimize the size of your final container image
  - Don’t include unneeded libraries
  - To get this advantage with Singularity, build a Docker image and convert it to Singularity
- Host integration vs. portability trade off
FOR BEST INTEGRATION

- Exactly match InfiniBand userspace component versions
  - (M)OFED version should match host
    - If available, nv_peer_mem, gdr_copy, and xpmem/knem should match host
  - Exactly match host MPI flavor and version
    - Should match configure options as well
FOR BEST PORTABILITY

- (M)OFED drivers
  - MOFED 4.4+ will maintain forwards/backwards compatibility
  - Otherwise, OFED drivers generally have fewer compatibility issues than MOFED drivers but you will lose out on some features

- Use OpenMPI
  - “Plugin” design can support many systems with choices delayed until runtime
  - Can build support for lots of transport backends, resource managers, filesystem support, etc in a single build
  - If possible, use 3.x or 4.x for best compatibility
FOR BEST PORTABILITY CONT’D

- Use UCX
  - Replaces deprecated openIB OpenMPI component
  - UCX is default starting with OpenMPI 4.0
  - Supports intra/inter node optimized transports
  - When built with nv_peer_mem, gdr_copy, knem, xpmem, cma it will automatically pick the best backend based on host support
HPC CONTAINER MAKER (HPCCM)

- Simplifies the creation of container specification files
- Building block abstraction of components from implementation
  - Best practices for free
  - Updates to building blocks can be leveraged with a re-build
- Full power of Python in container recipes
- User arguments allow a single recipe to produce multiple containers

For more information on HPCCM, reference the “Containers Made Easy with HPC Container Maker” webinar or view the project’s README and source at https://github.com/NVIDIA/hpc-container-maker
GET STARTED TODAY WITH NGC

Sign Up and Access Containers for Free

To learn more about all of the GPU-accelerated software from NGC, visit: nvidia.com/cloud

To sign up or explore NGC, visit: ngc.nvidia.com