MAKING CONTAINERS EASIER WITH HPC CONTAINER MAKER (HPCCM)

Journées Calcul Données (JCAD) 2019, 11 octobre 2019, Toulouse
Frédéric Parienté, Solutions Architect Manager, NVIDIA
NVIDIA
“THE AI COMPUTING COMPANY”

GPU Computing

Computer Graphics

Artificial Intelligence
DEEP LEARNING INSTITUTE

Fundamentals and advanced hands-on training in key technologies and application domains

Deep Learning Fundamentals
Accelerated Computing Fundamentals
AI for Autonomous Vehicles
AI for Digital Content Creation and Game Development
AI for Finance
AI for Healthcare Image Analysis
AI for Healthcare Image Analysis
AI for Intelligent Video Analytics
AI for Robotics

Caffe2
Microsoft Rapids
mxnet
TensorFlow
PyTorch
Keras
Theano
NVIDIA GPU CLOUD (NGC)
GPU-optimized Software Hub. Simplifying DL, ML and HPC Workflows

50+ Containers
DL, ML, HPC

Pre-trained Models
NLP, Classification, Object Detection & more

Model Training Scripts
NLP, Image Classification, Object Detection & more

Industry Workflows
Medical Imaging, Intelligent Video Analytics

Simplify Deployments
Innovate Faster
Deploy Anywhere
WHAT IF A CONTAINER IMAGE IS NOT AVAILABLE FROM NGC?
BARE METAL VS. CONTAINER WORKFLOWS

Login to system (e.g., CentOS 7 with Mellanox OFED 3.4)

$ module load PrgEnv/GCC+OpenMPI

$ module load cuda/9.0

$ module load gcc

$ module load openmpi/1.10.7

Steps to build application

FROM nvidia/cuda:9.0-devel-centos7

Result: application binary suitable for that particular bare metal system
## OPENMPI DOCKERFILE VARIANTS

Real examples - which one should you use?

### A
```bash
RUN apt-get update
   && apt-get install -y --no-install-recommends
   libopenmpi-dev
   openmpi-bin
   openmpi-common
   && rm -rf /var/lib/apt/lists/*
ENV LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/lib/openmpi/lib
```

### B
```bash
RUN OPENMPI_VERSION=3.0.0 &&
    cd openmpi-$OPENMPI_VERSION &&
    ./configure --prefix=/usr/local/openmpi --disable-getpwuid &&
    --prefix=/usr/local/mpi --disable-getpwuid &&
    make -j$(nproc) install &&
    cd .. &&
```n
### C
```bash
COPY openmpi /usr/local/openmpi
WORKDIR /usr/local/openmpi
RUN /bin/bash -c "source /opt/pgi/LICENSE.txt && CC=pgcc CXX=pgc++ F77=pgf77 FC=pgf90 .configure --with-cuda --prefix=/usr/local/openmpi"
RUN /bin/bash -c "source /opt/pgi/LICENSE.txt && make all install"
```

### D
```bash
RUN mkdir /logs
RUN wget -nv https://www.openmpi.org/software/ompi/v1.10/downloads/openmpi-1.10.7.tar.gz &&
    tar -xzf openmpi-1.10.7.tar.gz &&
    cd openmpi-* &&
    ./configure --with-cuda=/usr/local/cuda
    --enable-mpi-cxx --prefix=/usr && tee /logs/openmpi_config &&
    make -j 32 && tee /logs/openmpi_make && make install 2>&1 &&
    cd /tmp &&
```n
### E
```bash
WORKDIR /tmp
ADD http://www.openmpi.org/software/ompi/v1.10/downloads/openmpi-1.10.7.tar.gz /tmp
RUN tar -xzf openmpi-1.10.7.tar.gz &&
    cd openmpi-* &&
    ./configure --with-cuda=/usr/local/cuda
    --enable-mpi-cxx --prefix=/usr &&
    make -j 32 && make install &&
    rm -rf openmpi-*
```

### F
```bash
RUN wget -q -O - https://www.openmpi.org/software/ompi/v3.0/downloads/openmpi-3.0.0.tar.bz2 | tar -xjf - &&
    cd openmpi-3.0.0 &&
    CXX=pgc++ CC=pgcc FC=pgfortran F77=pgf90 .configure --prefix=/usr/local/openmpi --with-cuda=/usr/local/cuda --with-verbs --disable-getpwuid &&
    make -j4 install &&
    rm -rf openmpi-3.0.0
```
HPC CONTAINER MAKER

- Tool for creating HPC application Dockerfiles and Singularity definition files
- Makes it easier to create HPC application containers by encapsulating HPC & container best practices into building blocks
- Experimental support for ARM and Power containers
- Open source (Apache 2.0)
  https://github.com/NVIDIA/hpc-container-maker
- `pip install hpccm`
BUILDING BLOCKS TO CONTAINER RECIPES

Stage0 += openmpi()

hpccm Generate corresponding Dockerfile instructions for the HPCCM building block

```bash
# OpenMPI version 3.1.2
RUN yum install -y \
    bzip2 file hwloc make numactl-devel openssh-clients perl tar wget && \
    rm -rf /var/cache/yum/*
RUN mkdir -p /var/tmp && wget -q -nc --no-check-certificate -P /var/tmp https://www.openmpi.org/software/ompi/v3.1/downloads/openmpi-3.1.2.tar.bz2 && \
    mkdir -p /var/tmp && tar -x -f /var/tmp/openmpi-3.1.2.tar.bz2 -C /var/tmp -j && \
    cd /var/tmp/openmpi-3.1.2 && CC=gcc CXX=g++ F77=gfortran F90=gfortran FC=gfortran ./configure --prefix=/usr/local/openmpi --disable-getpwuid --enable-orterun-prefix-by-default --with-cuda=/usr/local/cuda --with-verbs && \
    make -j4 && \
    make -j4 install && \
    rm -rf /var/tmp/openmpi-3.1.2.tar.bz2 /var/tmp/openmpi-3.1.2
ENV LD_LIBRARY_PATH=/usr/local/openmpi/lib:$LD_LIBRARY_PATH \
    PATH=/usr/local/openmpi/bin:$PATH
```
HIGHER LEVEL ABSTRACTION

Building blocks to encapsulate best practices, avoid duplication, separation of concerns

```python
openmpi(check=False,
        configure_opts=["--disable-getpwuid", ...],
        cuda=True,
        directory='',
        infiniband=True,
        ospackages=["bzip2", 'file', 'hwloc', ...],
        prefix="/usr/local/openmpi",
        toolchain=toolchain(),
        ucx=False,
        version='3.1.2')
```

# run “make check”?
# configure command line options
# enable CUDA?
# path to source in build context
# enable InfiniBand?
# Linux distribution prerequisites
# install location
# compiler to use
# enable UCX?
# version to download

Examples:
openmpi(prefix="/opt/openmpi", version='1.10.7')
openmpi(infiniband=False, toolchain=pgi.toolchain)

Full building block documentation can be found on GitHub
EQUIVALENT HPCCM WORKFLOW

Login to system (e.g., CentOS 7 with Mellanox OFED 3.4)

$ module load PrgEnv/GCC+OpenMPI

$ module load cuda/9.0

$ module load gcc

$ module load openmpi/1.10.7

Steps to build application

Result: application binary suitable for that particular bare metal system

Stage0 += baseimage(image='nvidia/cuda:9.0-devel-centos7')
Stage0 += mlnx_ofed(version='3.4-1.0.0.0')

Steps to build application

Result: portable application container capable of running on any system

Stage0 += gnu()

Stage0 += openmpi(version='1.10.7')
DEMO
SUMMARY

• HPC Container Maker simplifies creating a container specification file
  • Best practices used by default
  • Building blocks included for many popular HPC components
  • Flexibility and power of Python
  • Supports Docker (and other frameworks that use Dockerfiles) and Singularity
  • Experimental support for ARM and Power containers

• DLI Course: High Performance Computing with Containers ($30)
• Open source: https://github.com/NVIDIA/hpc-container-maker
• pip install hpccm
Merci!