Introduction to Charliecloud

Friday, 14 July 2023

ACES Summer Workshop

Wesley Brashear, Megan Phinney, Richard Lawrence, Zhenhua He
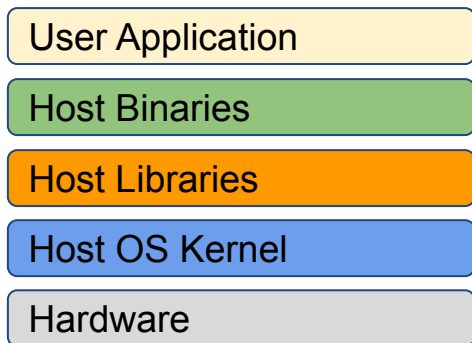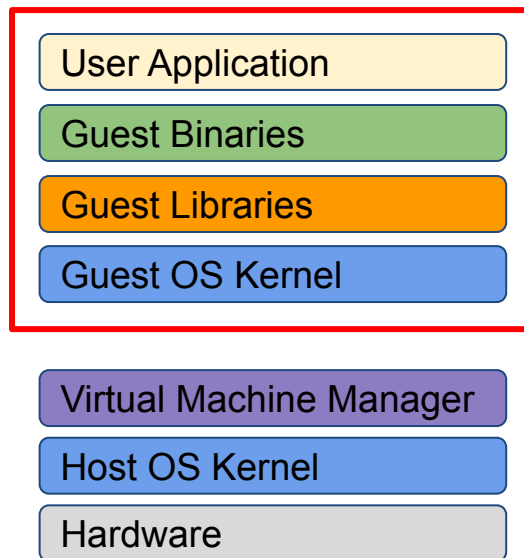
Introduction to Containers

- Containers make applications more portable
- Unlike in VMs, the OS Kernel is not duplicated

Local Build

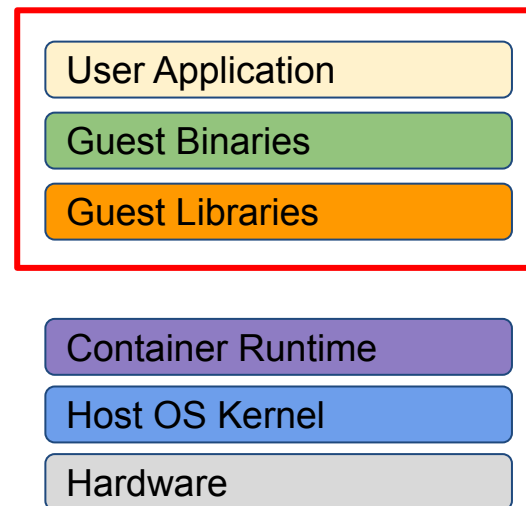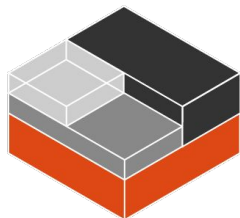| User Application |
| Host Binaries |
| Host Libraries |
| Host OS Kernel |
| Hardware |

Virtual Machine

| User Application |
| Guest Binaries |
| Guest Libraries |
| Guest OS Kernel |

| Virtual Machine Manager |
| Host OS Kernel |
| Hardware |

Container

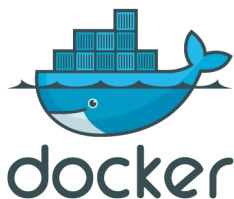| User Application |
| Guest Binaries |
| Guest Libraries |

| Container Runtime |
| Host OS Kernel |
| Hardware |

Popular Containers

Instant deployment to users on different devices!

| LXC 2008 | Docker 2013 | Singularity 2015 | Shifter 2016 | Charliecloud 2017 | Podman 2018 |

Container Basics

Containers come in two parts:

1. **Image:**

   - A file containing all the parts of an environment, libraries, and applications
   - Generally built by experts
   - Found in online repositories

2. **Runtime:**

   - Compatibility layer that translates between the container environment and the host OS
   - Runtime is installed by cluster administrators

Why use containers?

- **Shareability**
  - Share your container image file by uploading to a public repository
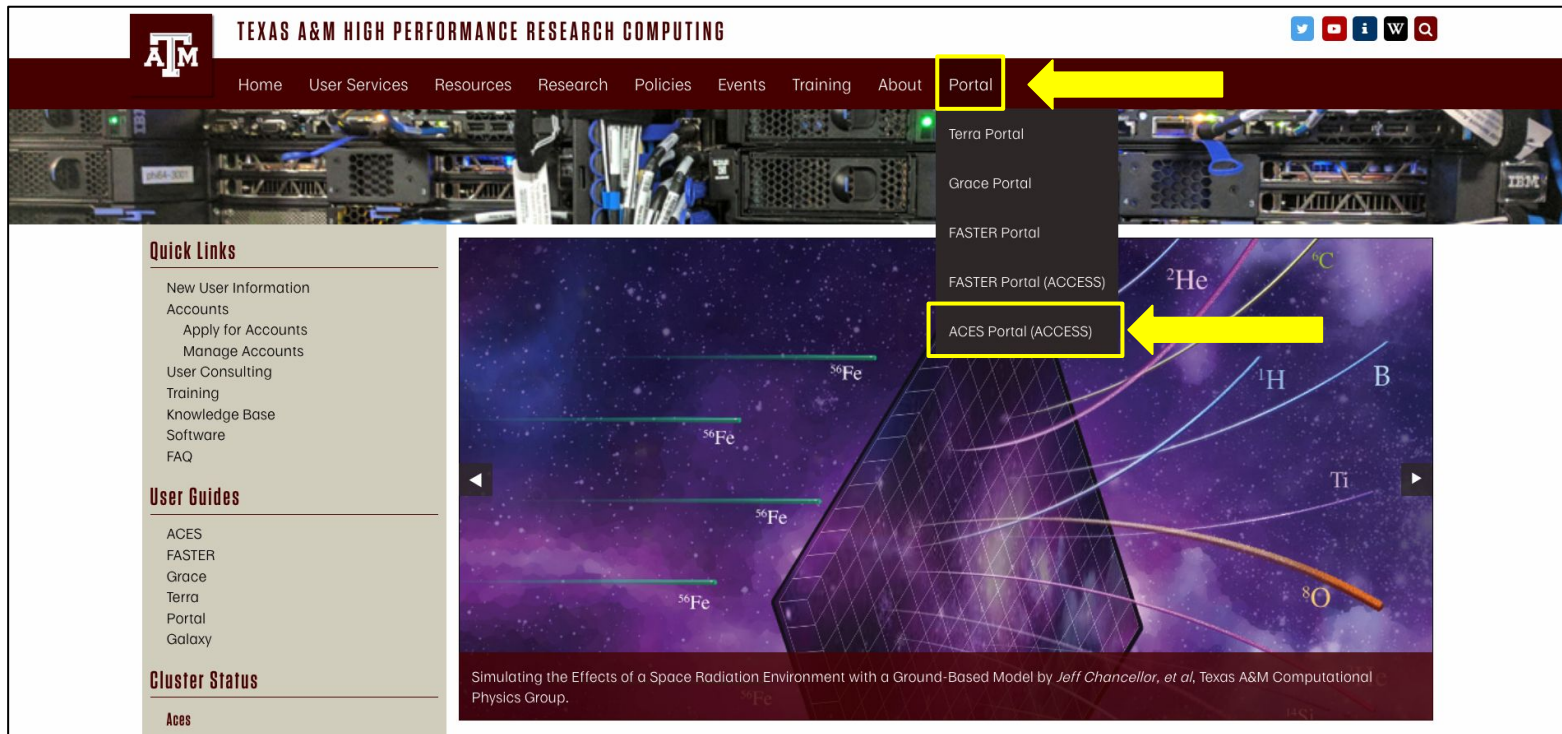 - Use images shared by others

- **Portability**
  - Use images on any computer with the same architecture (x84-64)

- **Reproducibility**
  - Container users are largely unaffected by changes to the cluster environments

Log into ACES through the HPRC Portal

Accessing ACES via the ACES Portal (ACCESS)

Log-in using your ACCESS credentials.

Select the Identity Provider appropriate for your account.

Log into ACES through the HPRC Portal
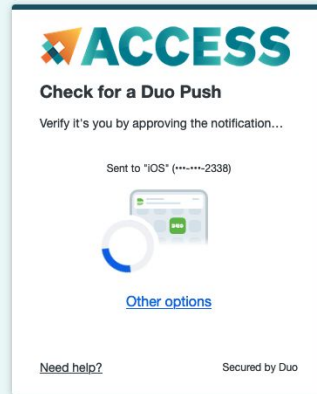
Two-Factor
Authentication

To access an shell on ACES, Click on "Clusters" menu/_aces Shell Access

Success!

Charliecloud: Use cases on ACES

Charliecloud: Use cases on ACES

Let's set up our environment:

```
$ cd $CHORKSHOP
$ export TRAINING=/scratch/training/charliecloud
$ ls $TRAINING
$ mkdir tensorflow
$ mkdir $SCRATCH/ch_parabricks
```

Introduction to TensorFlow

TensorFlow is one of the most popular program frameworks for building machine learning applications.

- Google Brain built **DistBelief** in 2011 for internal usage.
- TensorFlow 1.0.0 was released on Feb 11, 2017
- TensorFlow 2.0 was released in Jan 2018.
- The latest version of TensorFlow is 2.13 released 5 Jul 2023.

Machine Learning with TensorFlow

We'll pull the image for TensorFlow and convert it to SquashFS format with Charliecloud

```
$ cd tensorflow
$ ch-image pull intel/intel-optimized-tensorflow-avx512
$ ch-image list
a
almalinux:8
b
intel/intel-optimized-tensorflow-avx512
$ ch-convert intel/intel-optimized-tensorflow-avx512 intel-tensorflow.sqfs
```

Machine Learning with Tensorflow

Run the container and import TensorFlow:

```
$ ch-run intel-tensorflow.sqfs -- python
Python 3.10.6 (main, Mar 10 2023, 10:55:28) [GCC 11.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
2023-07-08 22:12:20.218530: I
tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is
optimized to use available CPU instructions in performance-critical
operations.
To enable the following instructions: AVX512_VNNI, in other operations,
rebuild TensorFlow with the appropriate compiler flags.
```

Machine Learning with TensorFlow

Check Tensorflow version to make sure it is working and then exit the container:

```
>>> print("TensorFlow version:", tf.__version__)
TensorFlow version: 2.12.0
>>> exit()
```

Machine Learning with TensorFlow

Now we will use an example TensorFlow script to create, train, and test a model using the MNIST database of handwritten images:

```
$ cp $TRAINING/tensorflow_ex.py .
$ ls
intel-tensorflow.sqfs  tensorflow_ex.py
$ cat tensorflow_ex.py
# Import TensorFlow and check version
import tensorflow as tf
print("TensorFlow version:", tf.__version__)

# Set up mnist data
mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
[...]
```

Machine Learning with TensorFlow

We'll need to bind our current working directory to run the example script in the container

```
$ ch-run -b "$PWD:/mnt/1" -c "mnt/1" intel-tensorflow.sqfs -- python \
    ./tensorflow_ex.py
[...]
TensorFlow version: 2.12.0
2023-07-09 20:52:41.737430: I
tensorflow/core/common_runtime/process_util.cc:146] Creating new thread
pool with default inter op setting: 128. Tune using
inter_op_parallelism_threads for best performance.
Epoch 1/10
1875/1875 [==============================] - 2s 1ms/step - loss: 0.2965 -
accuracy: 0.9135
Epoch 2/10
1875/1875 [==============================] - 2s 1ms/step - loss: 0.1414 -
[...]
```

Genomic Analyses with NVIDIA's Clara Parabricks

- GPU-accelerated version of common bioinformatics pipeline
- Works with both RNA-seq and WGS data
- NVIDIA provides images that containers easily integrate with Charliecloud
- Today's exercise will focus on completing the first portion of the pipeline
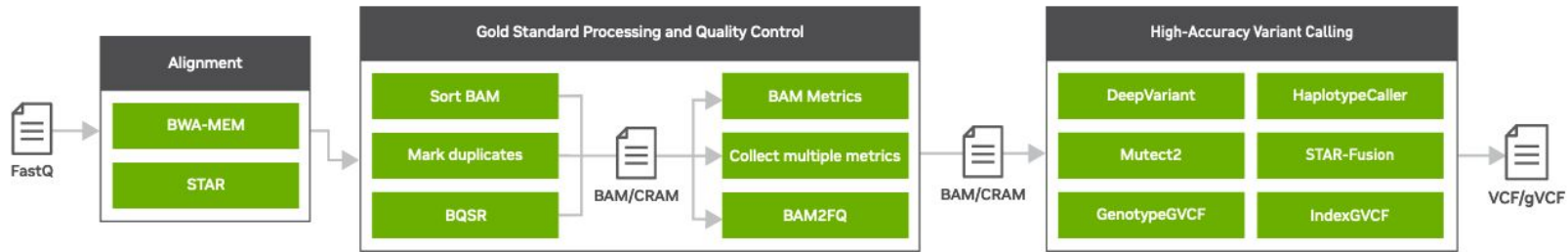
Genomic Analyses with NVIDIA's Clara Parabricks

- Massive speed-up versus CPU-only pipelines:

Performance Comparison
Germline End-To-End Secondary Analysis

1,200 minutes			
	40 minutes	34 minutes	22 minutes
CPU/GATK	8x T4	8x V100	8x A100

Data was generated using publicly available data (https://precision.fda.gov/challenges/truth) for NA12878, deprecating the data to 30X coverage. For the 22-minute runtime, DGX A100 with 320G memory was used. The native GATK4.1 numbers were generated using 32 vCPU (3.1 GHz Intel Xeon® Platinum 8175M) using 320Gb RAM.

Using GPUs with Charliecloud

- Need to "inject" the correct NVIDIA GPU libraries into the container
  - nvidia-container-cli
  - nvidia libraries and executables

- On ACES:
  - nvidia-container-cli is provided as a module
  - Compute nodes with GPUs have matching libraries present

Genomic Analyses with NVIDIA's Clara Parabricks

- Containers need to be created on a node with GPUs
- We will request an interactive session on a compute node equipped with NVIDIA's H100 GPUs

```
$ srun --mem=240G --time=01:00:00 --gres=gpu:h100:1 \
        --partition=gpu --cpus-per-task=48 --pty bash
```

Genomic Analyses with NVIDIA's Clara Parabricks

- Change directories and copy the example material:

```
$ cd $SCRATCH/ch_parabricks
$ export TRAINING=/scratch/training/charliecloud/
$ cp $TRAINING/sample* .
$ cp $TRAINING/Homo* .
$ ls
Homo_sapiens_assembly38.fasta    Homo_sapiens_assembly38.fasta.ann
Homo_sapiens_assembly38.fasta.fai  Homo_sapiens_assembly38.fasta.sa
sample_2.fastq.gz
[...]
```

Genomic Analyses with NVIDIA's Clara Parabricks

- Load the modules required for the exercise

```
$ module load nvidia-container-cli
$ module load WebProxy
```

- Pull the parabricks image from NVIDIA using Charliecloud:

```
$ ch-image pull nvcr.io/nvidia/clara/clara-parabricks:4.1.1-1 \
    parabricks-4.1.1-1
```

Genomic Analyses with NVIDIA's Clara Parabricks

- Check the image we just pulled and convert it to a directory stored on $TMPDIR

```
$ ch-image list
parabricks-4.1.1-1
$ ch-convert parabricks-4.1.1-1 $TMPDIR/parabricks4.1
input:   ch-image  parabricks-4.1.1-1
output:  dir   /tmp/job.2085/parabricks4.1
exporting ...
done
```

Genomic Analyses with NVIDIA's Clara Parabricks

- Inject the necessary NVIDIA libraries (to be able to utilize the GPUs)

```
$ ch-fromhost --nvidia $TMPDIR/parabricks4.1
/sbin/ldconfig.real: Can't stat /usr/local/nvidia/lib: No such file or
directory
/sbin/ldconfig.real: Can't stat /usr/local/nvidia/lib64: No such file or
directory
/sbin/ldconfig.real: Can't stat /usr/local/lib/x86_64-linux-gnu: No such
file or directory
done
```

Genomic Analyses with NVIDIA's Clara Parabricks

- Convert the container to a SquashFS file:

```
$ ch-convert $TMPDIR/parabricks4.1 parabricks4.1.sqfs
input:   dir    /tmp/job.2085/parabricks4.1
output:  squash parabricks4.1.sqfs
packing ...
Cannot stat exclude dir/file /tmp/job.2085/parabricks4.1/.git because No
such file or directory, ignoring
Cannot stat exclude dir/file /tmp/job.2085/parabricks4.1/.gitignore
because No such file or directory, ignoring
Parallel mksquashfs: Using 96 processors …
```

Genomic Analyses with NVIDIA's Clara Parabricks

- Now we are ready to run Parabricks!

```
$ ch-run -b "$PWD:/mnt/1" -c "mnt/1" parabricks4.1.sqfs \
        pbrun fq2bam -- --ref Homo_sapiens_assembly38.fasta \
        --in-fq sample_1.fastq.gz sample_2.fastq.gz --out-bam test.bam
```

Success! It's that easy!

https://hprc.tamu.edu

HPRC Helpdesk:

help@hprc.tamu.edu
Phone: 979-845-0219

Help us help you. Please include details in your request for support, such as, Cluster (ACES, FASTER, Grace, Terra, ViDaL), NetID (UserID), Job information (JobID(s), Location of your jobfile, input/output files, Application, Module(s) loaded, Error messages, etc), and Steps you have taken, so we can reproduce the problem.