Analysis Pipeline on the Cloud

Docker and AWS Batch
Presentation

● Overview
  ○ Docker
  ○ AWS Batch Service

● Importance of Docker

● Docker Images and the Analysis Pipeline
  ○ Summary of the Docker Images
  ○ Analysis Pipeline using AWS Batch Service

● Examples
  ○ Using Docker
  ○ Analysis Pipeline using AWS Batch Service
Docker Overview

- Platform for developing, deploying and running applications or systems
- A *Docker image* is:
  - built containing all software necessary to run the application
    - Usually built from a base image (e.g., *ubuntu*)
    - Includes all additional software to support an application or system (e.g., *gnu c/c++, python*)
    - Typically composed of multiple layers (e.g., *ubuntu layer, development tools layer, R layer*)
  - a read-only template with instructions for creating a *Docker container*
Docker Overview (cont)

- A Docker container is:
  - a runnable instance of an image on a local or host computer (e.g., Windows 10, macOS, Ubuntu)
  - what the image becomes in memory when executed
  - runs natively on Linux
  - runs a Virtual Machine on macOS and Windows with access to host resources via a hypervisor
  - the container is considered *stateless* - when the container stops all changes to code and data are discarded (except for data on local host that is mapped to the container)
Docker Overview (cont)

What about accessing data on local host?

- Data is typically not included in the *Docker image*
- Data accessible on the local host can be mapped (or *bind mounted*) to the *Docker container*
- Any changes to data that is mapped to the local host is persisted when the *Docker container* stops

1. On macOS, file sharing is specified in the *Docker Preferences*
Docker Overview (Docker Images)

- HPC (e.g., MKL)
- Develop Tools (e.g., c++)
- Ubuntu 16.04

Docker Image: Ubuntu-hpc

- R 3.5.0
- Ubuntu-hpc

Docker Image: r-3.5.0

- Analysis Pipeline
- R Packages
- r-3.5.0

Docker Image: topmed
Docker Overview (Docker Container)

Linux, macOS or Windows Computer

Analysis Pipeline
R Packages

r-3.5.0

Data Volume
Overview AWS Batch Service

AWS Cloud

- Computer Instance
- Docker Image/Container

- Batch API
- AUTO SCALE AND LAUNCH
- BATCH Services
- Queue Definitions
- Job Definitions
- Compute Environments

Analysis Pipeline on the Cloud
Importance of Docker (General)

- Develop a docker image with a completely configured system for running applications
- Deploy and run the same docker image on multiple platforms (e.g., *ubuntu*, *macOS*, *Windows*)
- Facilitates integrating applications in different environments (e.g., *AWS*, *Azure*, *Google Cloud*, *Seven Bridges*)
- Significantly reduces administrative cost in configuring computer systems to support the often numerous and diverse software required by applications
Importance of Docker (Analysis Pipeline)

- Easily deploy the base environment of software, libraries, and R packages associated with the analysis pipeline:
  - R
  - R packages
  - Math Kernel Library
  - Development environment (e.g., c++, python)
- Integrate with AWS Batch Services and its high-performance, parallel computing environment
- Integrate with Seven Bridges Genomics
- Potential to integrate in other high-performance, parallel computing environments
Docker Images and the Analysis Pipeline

- **Summary of the Docker Images**
  - uwgac/r-3.5.0-mkl
  - uwgac/topmed-master
  - uwgac/topmed-rstudio

- **Analysis Pipeline using AWS Batch Service**
  - Provide high performance data access
  - Integrate analysis pipeline with AWS batch service
  - Run the Docker image interactively
Analysis Pipeline using AWS Batch Service

○ Provide high performance data access
  ■ Sharing data between computer instances via NFS
  ■ Mounting shared data to computer instances
  ■ Mapping shared data on computer instances to *Docker containers*
Analysis Pipeline using AWS Batch Service (cont)

○ Integrate analysis pipeline with AWS batch service
  ■ Define jobs, queues, and compute environments in AWS batch service
  ■ Provide a Docker image to AWS batch service (job definition)
  ■ Within analysis pipeline (AWS_Batch class), utilize python API to submit jobs
Analysis Pipeline using AWS Batch Service (cont)

○ Run the *Docker image* interactively
  ■ Copy AWS security credentials
  ■ Map shared data
  ■ Execute analysis pipeline commands (e.g., `assoc.py`)
  ■ Submit jobs to AWS Batch Service via python API
Examples - Using Docker

- Reference:
- Example 1 - Running RStudio server

  mkdir ~/workshop_2018
  cd ~/workshop_2018
  git clone https://github.com/uw-gac/docker_helpers
  alias rs_docker='~/workshop_2018/docker_helpers/Rstudio_docker.py'
  rs_docker

  # connect via browser http://localhost:8787
Examples - Analysis Pipeline Using AWS Batch Services

- Reference:

- Example
  # connect to aws instance running docker
  ssh -i ~/.ssh/xxx.pem kuraisa@52.27.98.54
  # get docker helpers (done previously)
  #git clone https://github.com/uw-gac/docker_helpers
  alias pipeline='~/docker_helpers/analysis_pipeline.py'
  # change working directory to shared data work folder
  cd /projects/topmed/analysts/kuraisa/workshop/burden
  pipeline --help
  #
  # run interactively docker image/container uwgac/topmed-master
  # (similar to connecting to head node of a linux cluster)
  #
  pipeline
Examples - Analysis Pipeline Using AWS Batch Services (cont)

- Example (cont)

```bash
# now within the docker container (head node) - get info about job
# without submitting
/usr/local/analysis_pipeline/assoc.py \
    single testdata/assoc_window_burden.config \
    --cluster_type AWS_Batch \
    --cluster_file custom_batch.json --print > single_burden_print.log
2>&1

more single_burden_print.log

# submit the job
/usr/local/analysis_pipeline/assoc.py \
    single testdata/assoc_window_burden.config \
    --cluster_type AWS_Batch \
    --cluster_file custom_batch.json > single_burden.log 2>&1
```
Examples - Analysis Pipeline Using AWS Batch Services (cont)

- Example (cont)
  # after submitting jobs,
  # monitor from aws console (AWS Batch dashboard)

  # wait for instance to start 5-10 mins
  # (using spot may affect time)

  # after job is running, view dashboard on console and
  # list files on the "head" node
  ls
Summary

- Overview of Docker and AWS Batch
- Importance of Docker
- Examples using Docker
- Example executing analysis pipeline on AWS Batch
- Next Presentation: Analysis Pipeline on *Seven Bridges Genomics*
Questions