MOCHI BOOT CAMP

Argonne NATIONAL LABORATORY

SESSION 2: HANDS-ON

PHILIP CARNS
Argonne National Laboratory

U.S. DEPARTMENT OF ENERGY
Argonne National Laboratory is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC.

October 24-26, 2019

SESSION OBJECTIVE

It’s simple(ish): run your first Mochi program!

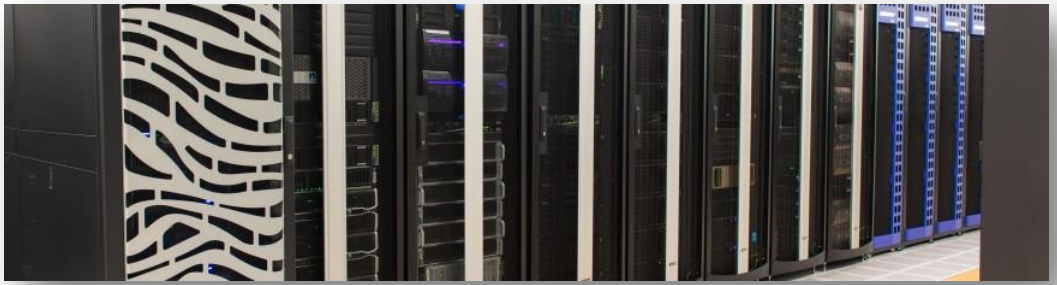
- Instructions for this session can be found by following the “Session 2: Hands-on” link in the README.md for the boot camp repository
- Some key requirements:
  - Logging on to the test system (JLSE)
  - Setting up a Spack environment
  - Running jobs
  - Observing the Mochi version of “Hello World”

- If you are ready to run on your laptop or a system at your home institution, that’s fine too! The important thing is to end this session ready to run code, though, and our test environment is likely the fastest way to start.

THE TEST SYSTEM

JLSE “it” nodes

- The JLSE is a test environment (not production)
- We have the use of a set of Broadwell nodes with local SSDs this week
  - Feel free to hold an individual for the entire day as an interactive job
  - All experiments can be run on a single node using shared-memory communication (all clients and servers on the same node)
  - If you decide to try a multi-node job, please keep them shorter



SPACK PACKAGE MANAGEMENT

and why it's relevant to Mochi

- Spack is a software package management system for HPC, developed at LLNL
- Things it can do:
  - Download, compile, and install a package and all of its dependencies
  - Maintain platform-specific configuration options (e.g., what network transport to use)
  - Doesn't need root access
- Spack is not a hard requirement for Mochi: any component can be built manually
  - But it is ***really*** **helpful for Mochi** because of our fundamental philosophy: many small components that are combined as needed. Spack excels at this.

RUNNING JOBS WITH COBALT

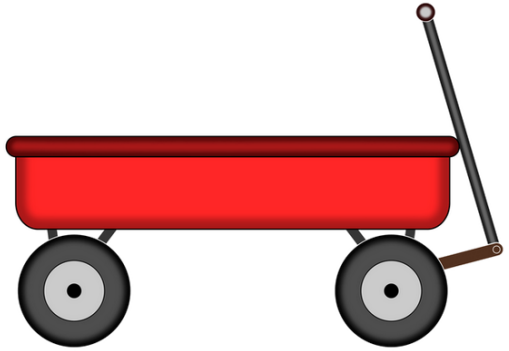
Scheduling and interactive jobs

- Compile things on the login node, but please don’t run your jobs there
- Run your experiments on a compute node
- All compute nodes in this environment are controlled by a batch scheduler: Cobalt
- Cobalt is roughly analogous to Slurm, Torque, or PBS
- You are welcome to create batch jobs, but in this session the easiest thing to do is to submit a long (hours) interactive job
  - Ssh to the node to open as many terminals as you want
  - Observe the examples in real time

See details in the mochi-boot-camp repository.

TAKING THIS SHOW ON THE ROAD

What to do to adapt this to another system



- You can't take JLSE nodes home with you
- What do you need to change to run these examples elsewhere?
  - The most crucial compile element is the "packages.yaml" file in Spack
  - Think of this as the recipe for building on your platform
  - Edit it to change:
    - Mercury variants (to tell Mercury what transports to support)
    - What existing software to use on the system (like autoconf)
- At run time:
  - Use address strings appropriate for your transport.
  - For example, in place of "na+sm" use "ofi+verbs"
  - We'll cover some transport details in another session

TRY IT OUT, AND LET US KNOW IF YOU HAVE QUESTIONS