Quick start on the ALCF Blue Gene/Q

and more

ATPESC
July 31, 2016

Ray Loy
Performance Engineering
Argonne Leadership Computing Facility
References

- **Sample files**
  - On Vesta, Mira, Cetus, or Cooley:
    - /projects/ATPESC2016/examples/getting-started

- **Online docs**
  - www.alcf.anl.gov/user-guides
Cryptocard tips

- The displayed value is a hex string. Type your PIN followed by all letters as CAPITALS.
- If you fail to authenticate the first time, you may have typed it incorrectly
  - Try again with the same crypto string (do NOT press button again)
- If you fail again, try a different ALCF host with a fresh crypto #
  - A successful login resets your count of failed logins
- Too many failed logins → your account locked
  - Symptom: You get password prompt but login denied even if it is correct
- Too many failed logins from a given IP → the IP will be blocked
  - Symptom: connection attempt by ssh or web browser will just time out
**Softenv**

- Similar to **modules** package
- Keys are read at login time to set environment variables like PATH.
  - Mira, Cetus, Vesta: `~/.soft`
  - Cooley: `~/.soft.cooley`
- To get started:
  ```
  # This key selects XL compilers to be used by mpi wrappers
  +mpiwrapper-xl
  @default
  # the end – do not put any keys after the @default
  ```
- After edits to `.soft`, type "resoft" or log out and back in again
Using compiler wrappers

- **IBM XL cross-compilers:**
  - SoftEnv key: +mpiwrapper-xl
  - Non-thread-safe: mpixlc, mpixlcxx, mpixlf77, mpixlf90, mpixlf95, mpixlf2003, etc.
  - **Thread-safe** (add _r suffix): mpixlc_r, mpixlcxx_r, mpixlf77_r, etc.
  - Example: mpixlc -O3 -o hellompi hellompi.c

- **GNU cross-compilers:**
  - SoftEnv key: +mpiwrapper-gcc
  - mpicc, mpicxx, mpif77, mpif90

- **CLANG cross-compilers:**
  - SoftEnv key: +mpiwrapper-bgclang
  - mpiclang, mpiclang++, mpiclang++11

BG/Q Job script

- Sample:

```bash
#!/bin/bash
#COBALT –n 32 –t 30 –q training –A ATPESC2016
# -p is mode (how many ranks per node)
# --np is number of ranks
runjob –p 16 --np 32 --block $COBALT_PARTNAME : hellompi
# Note: exit status of this script is runjob's status
```

- Some args use `single` dash and some `double` dash (`man runjob`)
- Don't forget `--block`. COBALT_PARTNAME is set automatically by Cobalt.
- You can do multiple runjobs in succession
  - Use normal shell redirection to separate output
- Must use `--envs` to pass environment variables into your program
- Output to `<jobid>.{output,error,cobaltlog}` (use `-O` to change prefix)
Cooley Job Script

- More like a typical Linux cluster
- Job script different than BG/Q.
  - Example test.sh:
    ```
    #!/bin/sh
    NODES=`cat $COBALT_NODEFILE | wc -l`
    PROCS=$((NODES * 12))
    mpirun -f $COBALT_NODEFILE -n $PROCS myprog.exe
    ```
  - Submit on 5 nodes for 10 minutes
    ```
    qsub -n 5 -t 10 -A ATPESC2016 ./test.sh
    ```
  - Refer to online user guide for more info
Submitting your job

- qsub –A <project> -q <queue> -t <time> -n <nodes> --mode script ./jobscript.sh
  E.g.
  qsub –A ATPESC2016 –q training –t 10 –n 32 –mode script ./jobscript.sh
  
  Note: runs on Mira should use "default" queue

- If you specify your options in the script via #COBALT, then just:
  - qsub jobscript.sh

- Make sure jobscript.sh is executable

- Without "-q", submits to the queue named "default"

- Without "-A", uses environment variable COBALT_PROJ if set
  - export COBALT_PROJ=ATPESC2016

- man qsub for more options
Managing job

- `qstat` – show what's in the queue
  - `qstat -u <username>`  # Jobs only for user
  - `qstat <jobid>`  # Status of this particular job
  - `qstat -fl <jobid>`  # Detailed info on job

- `qdel <jobid>`

- `showres` – show reservations currently set in the system

- `man qstat` for more options
Interactive job

- Useful for short tests or debugging
- Submit the job with –I (letter I for Interactive)
  - Default queue and default project
    - qsub –I –n 32 –t 30
  - For the workshop:
- Wait for job's shell prompt
  - This is a new shell with settings COBALT_PARTNAME, COBALT_JOBID
  - Exit this shell to end your job
- Run "wait-boot" ← Important!
- From job's shell prompt, run just like in a script job:
- After job expires, runjob will fail. Check qstat $COBALT_JOBID
Access to computing resources

- ALCF resources
  - Vesta (2-rack BG/Q)
    - Queue "training" (maps to 1K nodes 24/7, and reserved during evening hands-on sessions. See showres)
    - Queue "default" to access the rest of Vesta
  - Cooley – x86 cluster with NVIDIA GPUs
    - Hands-on reservation queues named R.ATPESC_* (see showres)
    - Queue "default" for other use
  - Mira (48-rack BG/Q), Cetus (4-rack BG/Q)
    - Queue "training" (8K nodes, 7-10PM nightly).
    - Queue "default" (for large/long jobs ask for score boost ascovel@anl.gov)
    - Test your Mira setup
      - Use Cetus (4-rack BG/Q) in queue "training" (1K nodes 24/7, reserved during evening hands-on) or "default" queue
**ALCF resources for ATPESC**

- Vesta is the main BG/Q resource for ATPESC jobs
  - run your jobs on Vesta unless larger nodecounts/longer walltimes are necessary
  - Queue "training" limit: 1hr walltime, 1 running job, 5 queued jobs

- In "training" queues, jobs have priority and/or access reserved nodes. In "default" queues you will be competing with non-ATPESC users for resources

- Mira will be used for students with
  - a) greater ability to scale, and
  - b) who wish to run larger/longer jobs during scheduled hands-on sessions.

- Mira, Cetus, and Cooley share the same filesystem (different from Vesta)
  - Avoid using Cetus for jobs less than 128 nodes in size
  - Cetus has a max partition size of 2048.
Aside: NERSC and OLCF

- **NERSC**
  - Accounts default to project "ntrain"
  - Use regular queues

- **OLCF**
  - Use queue "TRN001"
  - See additional info in your registration packet.
About node count and mode

- **Node count**
  - Minimum physical partition sizes available depend on machine
    - Vesta: 32  Cetus: 128  Mira: 512
    - Your job will get the smallest available size >= what you ask for
      - It is reserved for you; you are charged for entire partition

- **Mode**
  - How many MPI ranks per node
    - Possible values: 1,2,4,8,16,32,64
  - A node has 16 cores, each can run 4 threads
    - For modes < 16, an MPI rank will be assigned more than one core
    - Example: "-p 4" can run up to 16 threads per MPI rank
Using OpenMP

- Shared-memory parallelism is supported within a single node
  - Use MPI across compute nodes, OpenMP within a compute node
- **For XL compilers, thread-safe compiler version should be used** (mpixlc_r etc.) with any threaded application (either OMP or Pthreads)
  - OpenMP standard directives are supported (version 3.1)
  - Compile with –qsmp=omp,noauto  (Note: debugging use noopt)
  - Increase default thread stack size using environment value XLSMPOPTS=stack=NNN  (value per thread, e.g. 10M)
- Setting number of OpenMP threads
  - set using environment variable OMP_NUM_THREADS
  - must be exported to the compute nodes using runjob –envs

- Example: 32 nodes / 512 ranks / 4 threads per rank:
  ```bash
  #!/bin/bash
  #COBALT –n 32 –t 30
  runjob –block $COBALT_PARTNAME –p 16 --np 512 --envs OMP_NUM_THREADS=4 : a.out
  ```
Hands-on

- Questions/problems with your pre-class assignment?