Welcome to
Getting Started at the ALCF

Agenda

- **Part I**
  - Blue Gene/Q hardware overview
  - Building your code
  - Considerations before you run
  - Hands-on session

- **Part II**
  - Queuing and running
  - After your job is submitted
  - Potential problems
  - Hands-on session

Part I

Section:

Blue Gene/Q hardware overview

ALCF resources

- **Mira** (Production) – IBM Blue Gene/Q
  - 49,152 nodes / 786,432 cores
  - 768 TB of memory
  - Peak flop rate: 10 PF
  - Linpack flop rate: 8.1 PF
- **Cetus** (Test & Devel.) – IBM Blue Gene/Q
  - 4,096 nodes / 65,536 cores
  - 64 TB of memory
  - 838 TF peak flop rate
- **Vesta** (Test & Devel.) – IBM Blue Gene/Q
  - 2,048 nodes / 32,768 cores
  - 32 TB of memory
  - 419 TF peak flop rate
- **Cooley** (Visualization) – Cray + NVIDIA
  - 126 nodes / 1512 x86 cores (Haswell)
  - 126 NVIDIA Tesla K80 GPUs
  - 47 TB x86 memory / 3 TB GPU memory
  - 293 TF peak flop rate

Detail shot of Mira

Mira and her cables

IBM Blue Gene/Q

- **Storage**
  - Scratch: 27 PB usable capacity,
    330 GB/s bw (GPFS)
    aggregate over 2 file systems
  - Home: 1.1 PB usable capacity,
    45 GB/s bw (GPFS)

ALCF Resources

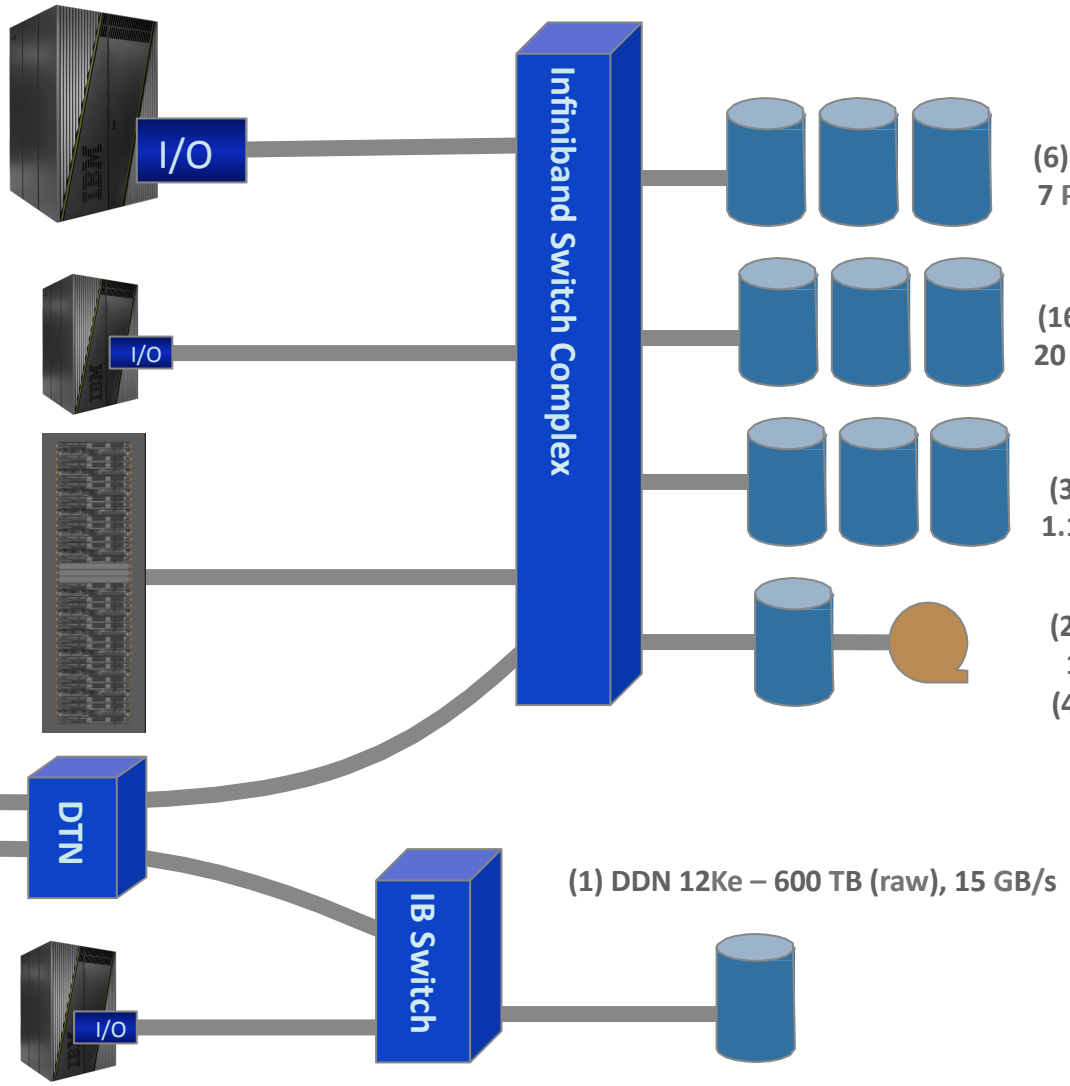
Mira
48 racks/768K cores
768 TB RAM
10 PF

Cetus (Dev)
4 rack/64K cores
64 TB RAM
838 TF

Cooley (Viz)
126 nodes/1512 cores
126 NVIDIA GPUs
47 x86 TB / 3 TB GPU RAM
293 TF

Networks – 100Gb
(via ESnet, internet2
UltraScienceNet,)

Vesta (Dev)
2 racks/32K cores
32TB RAM
419 TF



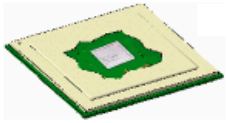
Blue Gene Features

- **Low speed, low power**
  - Embedded PowerPC core with custom SIMD floating point extensions
  - Low frequency: 1.6 GHz on Blue Gene/Q
- **Massive parallelism**
  - Many cores: 786,432 on Mira
- **Fast communication network(s)**
  - 5D Torus network on Blue Gene/Q
- **Balance**
  - Processor, network, and memory speeds are well balanced
- **Minimal system overhead**
  - Simple lightweight OS (CNK) minimizes noise
- **Standard programming models**
  - Fortran, C, C++ & Python languages supported
  - Provides MPI, OpenMP, and Pthreads parallel programming models
- **System-on-a-Chip (SoC) & Custom designed ASIC (Application Specific Integrated Circuit)**
  - All node components on one chip, except for memory
  - Reduces system complexity and power, improves price / performance
- **High reliability**
  - Sophisticated RAS (Reliability, Availability, and Serviceability)
- **Dense packaging**
  - 1024 nodes per rack

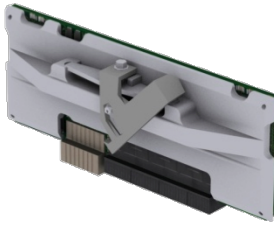
Blue Gene/Q

Chip
16+2 cores

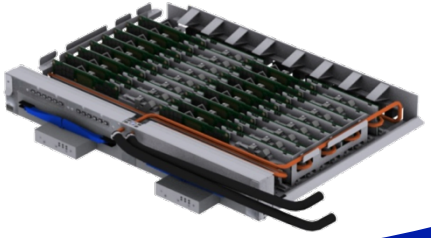
Module
Single chip



Compute card
One single chip module
16 GB DDR3 Memory
Heat Spreader for H_2O Cooling

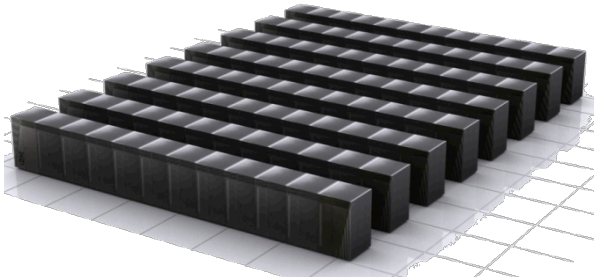


Node board
32 compute cards,
optical Modules,
link chips; 5D Torus



Rack
1 or 2 midplanes
0, 1, 2, or 4 I/O drawers

Multi-rack system
Mira: 48 racks, 10 PF/s

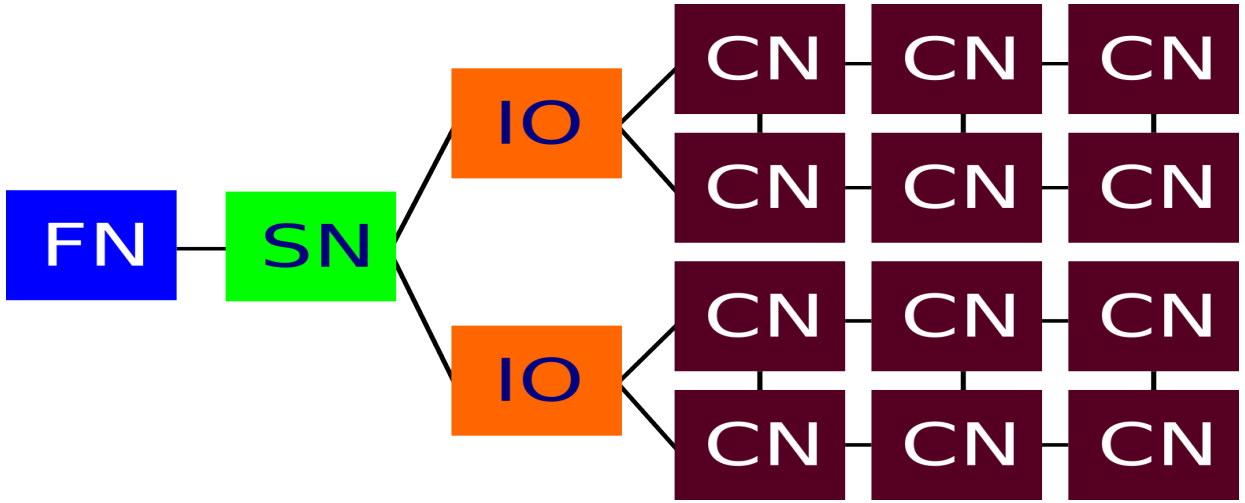


I/O drawer
8 I/O cards w/16 GB
8 PCIe Gen2 x8 slots
3D I/O Torus

Midplane
16 node boards

Blue Gene/Q system components

- **Front-end nodes** – dedicated for user's to login, compile programs, submit jobs, query job status, debug applications. **RedHat Linux OS.**
- **Service nodes** – perform partitioning, monitoring, synchronization and other system management services. Users do not run on service nodes directly.
- **I/O nodes** – provide a number of Linux/Unix typical services, such as files, sockets, process launching, signals, debugging; run Linux.
- **Compute nodes** – run user applications, use simple **compute node kernel (CNK)** operating system, ships I/O-related system calls to I/O nodes.



Partition dimensions on Blue Gene/Q systems

Mira

Nodes	A	B	C	D	E
512	4	4	4	4	2
1024	4	4	4	8	2
2048	4	4	4	16	2
4096	4/8	4	8/4	16	2
8192	4	4	16	16	2
12288	8	4	12	16	2
16384	4/8	8/4	16	16	2
24576	4	12	16	16	2
32768	8	8	16	16	2
49152	8	12	16	16	2

Command: partlist

http://www.alcf.anl.gov/user-guides/machine-partitions

Cetus

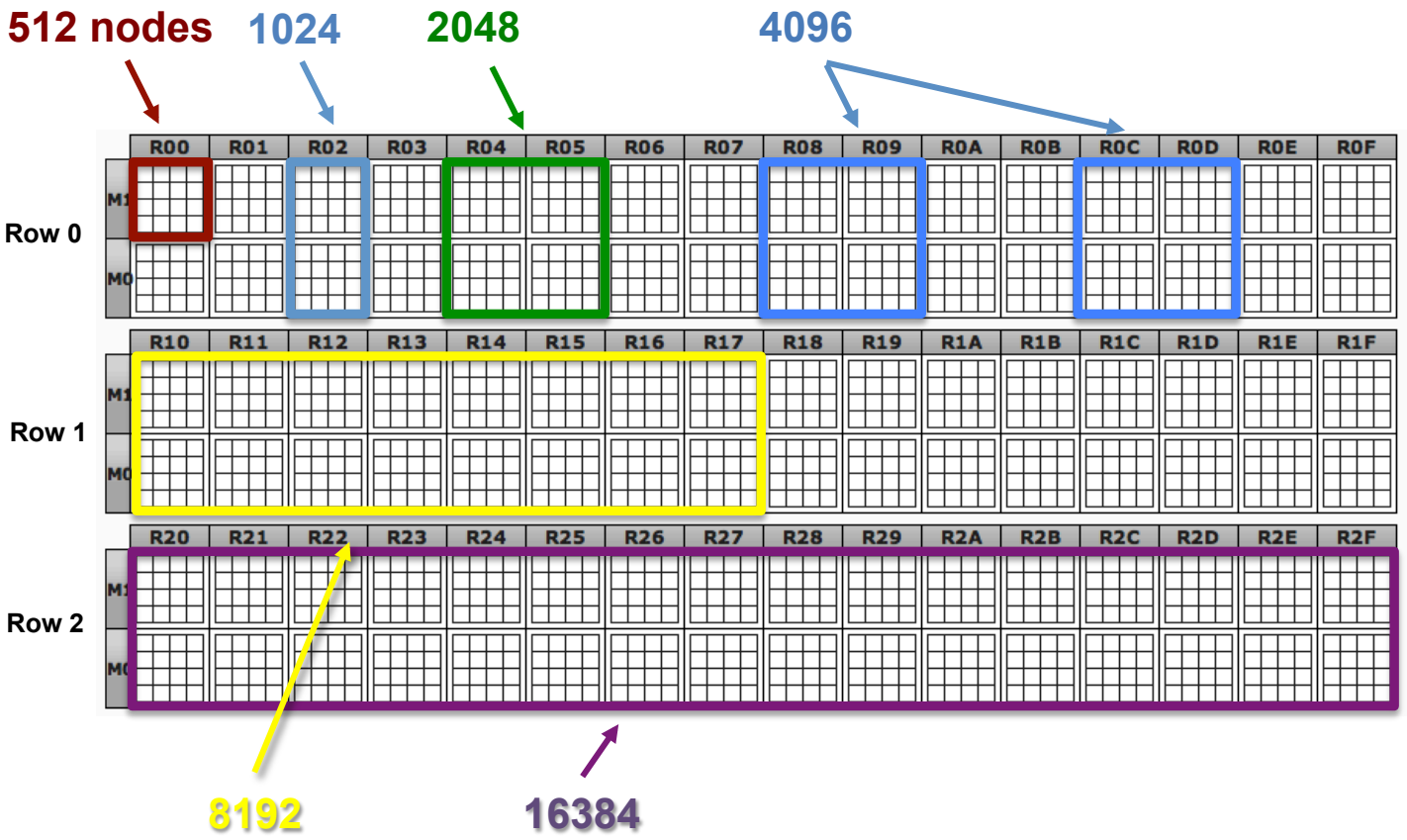
Nodes	A	B	C	D	E
128	2	2	4	4	2
256	4	2	4	4	2
512	4	4	4	4	2
1024	4	4	4	8	2
2048	4/8/8	4/4/4	8/4/8	8/8/4	2
4096(*)	8	4	8	8	2

Vesta

Nodes	A	B	C	D	E
32	2	2	2	2	2
64	2	2	4	2	2
128	2	2	4	4	2
256	4	2	4	4	2
512	4	4	4	4	2
1024	4	4	4/8	8/4	2
2048(*)	4	4	8	8	2

(*) Partition not active.

Mira multiple rack partitions ("blocks")



The number of large block sizes possible is:

# of nodes	# of blocks
49152	1
32768	3
24576	2
16384	9
12288	12
8192	6
4096	12
2048	24
1024	64
512	96

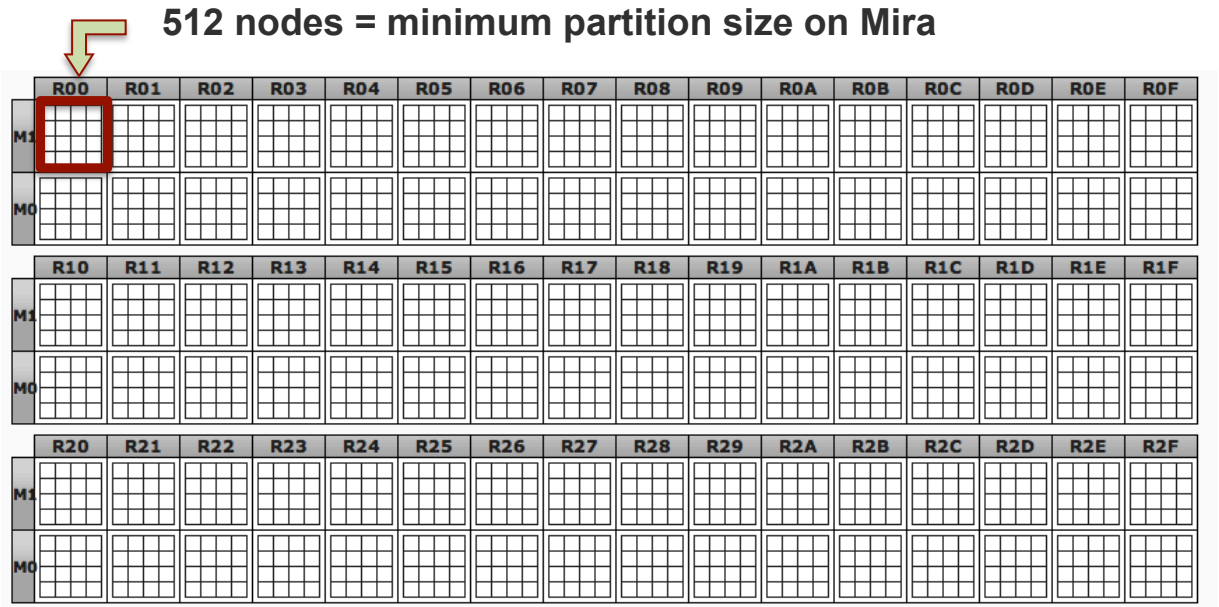
http://status.alcf.anl.gov/mira/activity (beta, a.k.a. The Gronkulator)

partlist will show you if a large free block is busy due to a wiring dependency

Minimum partition sizes on Blue Gene/Q systems

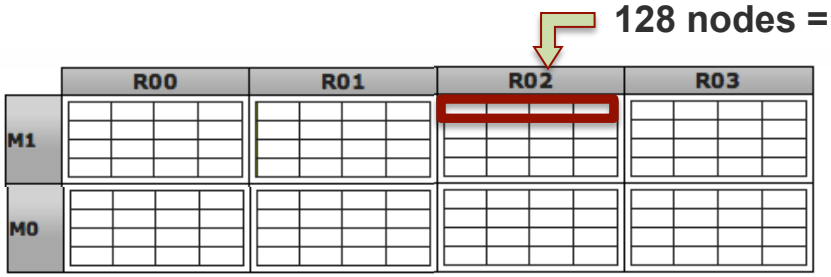
Mira

48 racks



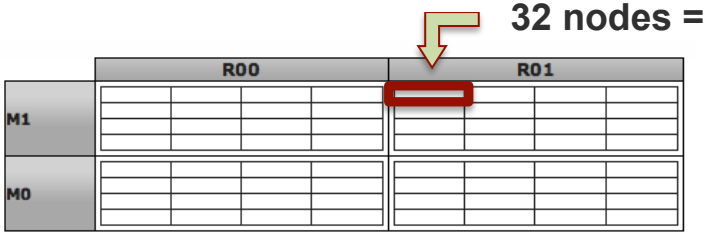
Cetus

4 racks



Vesta

2 racks



Section:

Building your code

SoftEnv

- A tool for managing a user's environment
  - Sets your PATH to access desired front-end tools
  - *Your compiler version can be changed here*
- Settings:
  - Maintained in the file ~/.soft (Mira/Cetus & Vesta) or ~/.soft.cooley (Cooley)
  - Add/remove keywords from ~/.soft or ~/.soft.tukey to change environment
  - ***Make sure @default is at the very end***
- Commands:
  - **softenv**
    - A list of all keywords defined on the systems
  - **resoft**
    - Reloads initial environment from ~/.soft or ~/.soft.cooley file
  - **soft add|remove keyword**
    - Temporarily modify environment by adding/removing keywords

http://www.mcs.anl.gov/hs/software/systems/softenv/softenv-intro.html

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.
  - "-show" option: shows complete command used to invoke compiler. E.g.:
    > mpixlc -show

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

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

http://www.alcf.anl.gov/user-guides/software-and-libraries

IBM XL Optimization Settings Options

Level	Implies	Description
-O0	-qstrict -qfloat=nofltint:norsqrt:rngchk -qstrict_induction	Preserves program semantics, minimal optimization. Best for debugging
-O2 (*or* -O)	-qstrict -qfloat=nofltint:norsqrt:rngchk -qnostrict_induction -qmaxmem=8192	Preserves program semantics, eliminates redundant code, basic loop optimization. Good for correctness check, baseline performance
-O3	-qnostrict -qfloat=fltint:rsqrt:norngchk -qnostrict_induction -qmaxmem=-1 -qhot=level=0	High order loop analysis and transformations, better loop scheduling, inlining, in depth memory access analysis. Can alter program semantics unless used with -qstrict
-O4	*All -O3 options plus* -qhot=level=1 -qhot=vector -qipa=level=1	Additional loop analysis, basic interprocedural optimization.
-O5	*All -O4 options plus* -qipa=level=2	Advanced interprocedural analysis (IPA).

IBM XL Optimization Tips

- Tips:
  - -qlistopt generates a listing with all flags used in compilation
  - -qreport produces a listing, shows how code was optimized
  - Performance can decrease at higher levels of optimization, especially at -O4 or -O5
  - May specify different optimization levels for different routines/files
  - The compiler option '-g' must be used to resolve the code line numbers in the debugger.

Threading

- **OpenMP** is supported
  - IBM XL compilers: -qsmp=omp:noauto
  - GNU: -fopenmp
  - BGCLANG: -fopenmp
- **Pthreads** is supported
  - NPTL Pthreads implementation in glibc requires no modifications
- Compiler **auto thread parallelization** is available
  - Use -qsmp=auto
  - Not always effective
- The runjob mode will determine maximum total number of threads (including the master thread)
  - runjob --ranks-per-node (or for non-script jobs, qsub --mode)
  - Maximum 4 threads per core
  - Each core needs at least 2 (possibly more) threads for peak efficiency

OpenMP

- Shared-memory parallelism is supported within a single node
- Hybrid programming model
  - MPI at outer level, across compute nodes
  - OpenMP at inner level, 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):
  - parallel, for, parallel for, sections, parallel sections, critical, single
  - #pragma omp <rest of pragma> for C/C++
  - !$OMP <rest of directive> for Fortran
- Compiler functions
  - omp_get_num_procs, omp_get_num_threads omp_get_thread_num, omp_set_num_threads
- Number of OpenMP threads
  - set using environment variable OMP_NUM_THREADS
  - must be exported to the compute nodes using runjob --envs (or for non-script jobs, qsub --env)

Software & libraries on Blue Gene/Q systems

- ALCF supports two sets of libraries:
  - IBM system and provided libraries: /bgsys/drivers/ppcfloor
    - glibc
    - mpi
    - PAMI (Parallel Active Messaging Interface)

- Site supported libraries and programs: /soft/libraries
  - ESSL, PETSc, HDF5, netCDF, Parallel netCDF, Boost
    - ESSL is IBM's optimized Engineering and Scientific Subroutine library for BG/Q: BLAS, LAPACK, FFT, sort/search, interpolation, quadrature, random numbers, BLACS
  - Additional tuned libraries in /soft/libraries/alcf subdirectory
    - BLAS, CBLAS, FFTW2, FFTW3, LAPACK, METIS, PARMETIS, PARPACK, SCALAPACK, SILO, SZIP, ZLIB

For a complete list visit:

http://www.alcf.anl.gov/user-guides/software-and-libraries

Tools: performance, profiling, debugging

- Non-system libraries and tools are under the /soft directory:
  - **/soft/applications** - applications
    - LAMMPS, NAMD, QMCPACK, etc.
  - **/soft/buildtools** - build tools
    - autotools, cmake, doxygen, etc.
  - **/soft/compilers** - IBM Compiler versions
  - **/soft/debuggers** - debuggers
    - DDT, Totalview
  - **/soft/libraries** - libraries
    - ESSL, PETSc, HDF5, NetCDF, etc.
  - **/soft/perftools** - performance tools
    - TAU, HPCToolkit, PAPI, OpenSpeedshop, Scalasca, HPM, etc.

Section:

Considerations before you run

Accounts, projects, allocations & such at ALCF

- ALCF Account
  - Login username
    - /home/username
  - Access to at least one machine
  - CRYPTOCard token for authentication
    - PIN
    - Must call ALCF help desk to activate
- Project
  - Corresponds to allocation of core-hours on at least one machine
  - User can be member of one or more projects
    - /projects/ProjectName
- Logging in
  - ssh -Y username@mira.alcf.anl.gov
    - Click button on CRYPTOCard
    - Password: PIN + CRYPTOCard display

Manage your account at

http://accounts.alcf.anl.gov

(password needed)

http://www.alcf.anl.gov/user-guides/accounts-access

Allocation Management

- Every user must be assigned to at least one Project:
  - Use '**projects**' command to query.
- Projects are then given allocations:
  - Allocations have an amount, start, and end date, and are tracked separately; Charges will cross allocations automatically. The allocation with the earliest end date will be charged first, until it runs out, then the next, and so on.
- Use '**cbank**' command to query allocation, balance:
  - cbank charges -p <projectname> # list all charges against a particular project
  - cbank allocations -p <projectname> # list all **active** allocations for a particular project
  - Other useful options:
    - -u <user> : show info for specific user(s)
    - -a <YYYY-MM-DD> : show info after date (inclusive), including **inactive** allocations
    - -b <YYYY-MM-DD> : show info before date (exclusive), including **inactive** allocations
    - --help

  Note: cbank is updated once a day, at midnight (CDT).
- Charges are based on the partition size, **NOT the number of nodes or cores used**!

http://www.alcf.anl.gov/user-guides/querying-allocations-using-cbank

HPC storage file systems at ALCF

Name	Accessible from	Type	Path	Backed Up to HPSS	*Daily Snapshots	Uses
vesta-home	Vesta	GPFS	/home or /gpfs/vesta-home	No	Yes	General use
projects	Vesta	GPFS	/projects	No	No	Intensive job output, large files
mira-home	Mira Cetus Cooley	GPFS	/home or /gpfs/mira-home	Yes	Yes	General use
projects	Mira Cetus Cooley	GPFS	/projects	No	No	Intensive job output, large files

* Daily snapshots are stored for 1 week on-disk in /gpfs/{vesta,mira}-home/.snapshots/. These snapshots do NOT persist in the event of disk failure.

http://www.alcf.anl.gov/user-guides/bgq-file-systems

Disk quota management

- **Disk storage**
  - /home directories:
    - Default of **100 GB** for Mira and **50 GB** for Vesta
    - Check your quota with the 'myquota' command
  - /projects directories:
    - Default of **1000 GB** for Mira and **500 GB** for Vesta
    - Check the quota in your projects with the 'myprojectquotas' command
  - See http://www.alcf.anl.gov/user-guides/data-policy#data-storage-systems

Backups and tape archiving

- **Backups**
  - On-disk snapshots of /home directories are done nightly
    - If you delete files accidentally, check:
      - /gpfs/mira-home/.snapshot on Mira
      - /gpfs/vesta-home/.snapshots on Vesta
  - **Only Mira/Cetus/Tukey home directories** are backed up to tape
    - The Vesta home directories are ***not*** backed up to tape (just daily snapshots)
    - Project directories are ***not*** backed up (/projects)

- **Manual data archiving to tape (HPSS)**
  - HSI is an interactive client
  - GridFTP access to HPSS is available
  - See http://www.alcf.anl.gov/user-guides/using-hpss

Data transfer to/from ALCF

The Blue Gene/Q connects to other research institutions using a total of 60 Gbits/s of public network connectivity

- Data management webinar:
  https://www.youtube.com/watch?feature=player_embedded&v=pEkgf2KnaU4

Data Transfer Utilities

- **GridFTP** (for large transfers)
  - Other site must accept our Certificate Authority (CA)
  - CRYPTOCard access available
- **sftp** and **scp** (for "small" transfers)
  - For local transfers of small files, not recommended for large data transfers due to poor performance and excess resource utilization on the login nodes.

Data Transfer Service

- **Globus** (for large transfers)
  - Globus addresses the challenges faced by researchers in moving, sharing, and archiving large volumes of data among distributed sites.
  - ALCF Blue Gene/Q endpoints: **alcf#dtn_mira, alcf#dtn_vesta**
  - Ask your laboratory or university system administrator if your institution has an endpoint.
  - Globus Connect Personal to share and transfer files to/from a local machine.

globus

http://www.alcf.anl.gov/user-guides/data-transfer

GSW15 Hands-on session (Part I)

- Log into Cetus or Vesta:

  > ssh username@cetus.alcf.anl.gov or > ssh username@vesta.alcf.anl.gov

  http://www.alcf.anl.gov/user-guides/connect-log

- GSW15 project:
  - Check if you are a member of the project used for this Hands-on session:

    > projects

    … ALCF_Getting_Started …
  - Check the allocation of core-hours available for this project:

    > cbank allocations -p ALCF_Getting_Started
  - Check the disk space in your $HOME and the projects that you are part of:

    > myquota

    > myprojectquotas

GSW15 Hands-on session (Part I)

- GSW15 reservation:
  - Check the name of the queue created for the Hands-on session:

    > showres

- Select MPI wrapper scripts:
  - The default user environment does not provide a set of MPI compiler wrappers. A wrapper may be selected by using the appropriate SoftEnv key.
  - For the following examples, ensure that you have the +mpiwrapper-xl key uncommented and that it is located before the @default line.

```
> cat ~/.soft
    +mpiwrapper-xl
    @default
> resoft
```

Note: after editing your ~/.soft file, run command 'resoft' to refresh your environment.

http://www.alcf.anl.gov/user-guides/overview-how-compile-and-link

Part II

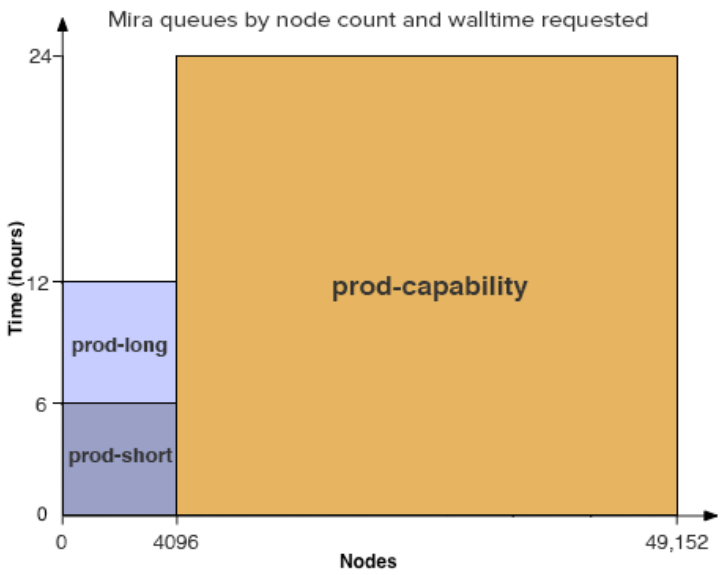
Section:

Queuing and running

Mira job scheduling

- Restrictions in queues
  - **prod-long:** restricted to the row 0.
  - **prod-short**, **prod-capability:** can run in the full machine

http://www.alcf.anl.gov/user-guides/job-scheduling-policy-bgq-systems



User Queued	Underlying Queue	Nodes	Wall-clock Time (hours)	Max. Running per User	Max. Queued per User
prod	prod-short	512 - 4096	0 - ≤6	5	20
	prod-long	512 - 4096	>6 - 12	5	20
	prod-capability	4097 - 49152	0 - 24	5	20
	backfill (*)	512 - 49152	0 - 6	5	20
prod-1024-torus	prod-1024-torus	1024	0 - 12	5	16
prod-32768-torus	prod-32768-torus	32768	0 - 24	1	20

(*) This queue is automatically selected based on the scheduling policy.

- **I/O to compute node ratio 1:128**

Cetus job scheduling

User Queue	Partition Sizes in Nodes	Wall-clock Time (hours)	Max. Running per User	Max. Queued per User
default, low	128, 256, 512, 1024, 2048	0 - 1	5	20

Cetus scheduling is designed to support application testing and debugging, not production work.

- **I/O to compute node ratio 1:128**

Vesta job scheduling

User Queue	Partition Sizes in Nodes	Wall-clock Time (hours)	Max. Running per User	Max. Queued Node-hours
default	32, 64, 128, 256, 512, 1024	0 - 2	5	1024
singles	32, 64, 128, 256, 512, 1024	0 - 2	1	1024
low	32, 64, 128, 256, 512, 1024	0 - 2	None	4096

- **I/O to compute node ratio 1:32**

http://www.alcf.anl.gov/user-guides/job-scheduling-policy-bgq-systems

Mira job boot times

- Each time a job is submitted using a standard qsub command, all nodes in a partition are rebooted.
- Boot times depend on the size of the partition:

Nodes in partition	Boot time (minutes)
≤ 2048	1
4096	1.5
8192	3
16384	4
32768	6
49152	7

The scheduler will attempt to boot the block up to three times if the boot procedure fails, so it may take as much as three times as long under rare circumstances.

Cobalt resource manager and job scheduler

- Cobalt is resource management software on all ALCF systems
  - Similar to PBS but not the same

- Job management commands:
  - **qsub**: submit a job
  - **qstat**: query a job status
  - **qdel**: delete a job
  - **qalter**: alter batched job parameters
  - **qmove**: move job to different queue
  - **qhold**: place queued (non-running) job on hold
  - **qrls**: release hold on job
  - **qavail**: list current backfill slots available for a particular partition size

- For reservations:
  - **showres**: show current and future reservations
  - **userres**: release reservation for other users

qsub Options

Syntax:

qsub [-d] [-v] -A <project name> -q <queue> --cwd <working directory>
--env envvar1=value1:envvar2=value2 --kernel <kernel profile>
-K <kernel options> -O <outputprefix> -t time <in minutes>
-e <error file path> -o <output file path> -i <input file path>
-n <number of nodes> -h --proccount <processor count>
--mode <mode> -M <email> --dependencies <jobid1>:<jobid2> <command> <args>

- Standard options:

-A project	project to charge
-q queue	queue
-t <time_in_minutes>	required runtime
-n <number_of_nodes>	number of nodes
--proccount <number_of_cores>	number of CPUs
--mode <cX | script>	running mode
--env VAR1=1:VAR2=1	environment variables
<command> <args>	command with arguments
-O project <output_file_prefix>	prefix for output files (default **jobid**)
-M <email_address>	e-mail notification of job start, end
--dependencies <jobid1>:<jobid2>	set dependencies for job being submitted
-I or --interactive	run an interactive command

Further options and details may be found in the man pages (> man qsub) or at:

http://trac.mcs.anl.gov/projects/cobalt/wiki/CommandReference

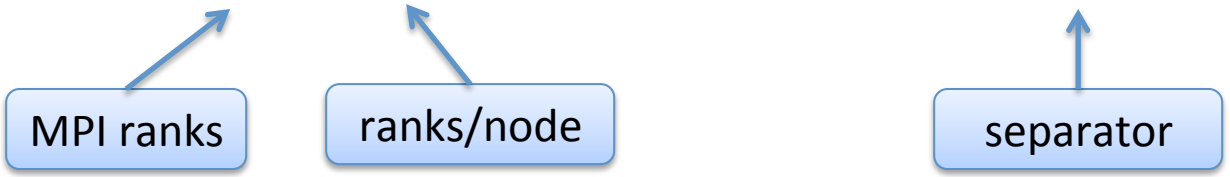
Cobalt job control: basic method

- **Basic**: submit a BG/Q executable
  qsub -n *nodes* --proccount *P* --mode c*N* … *path/executable*
  - *N* is number of processes (MPI ranks) per node
  - Node has 16 cores
    --mode c1 — 1 rank/node
    --mode c2 — 2 rank/node
    …
    --mode c16 — 1 rank/core
    --mode c32 — 2 rank/core
    --mode c64 — 4 rank/core
  - Threads
    qsub --mode c1 --env OMP_NUM_THREADS=64
    qsub --mode c2 --env OMP_NUM_THREADS=32
    …
    qsub --mode c16 --env OMP_NUM_THREADS=4
    ….

Cobalt job control: script method

- **Script**: submit a script (bash, csh, ….)
  qsub --mode script … *path/script*
  - Example:
    > qsub -t 10 -n 8192 -A myproject --mode script myscript.sh

```
#!/bin/sh
echo "Starting Cobalt job script"
runjob --np 131072 -p 16 --block $COBALT_PARTNAME : <executable> <args…>
```



  - Options may appear within script using #COBALT (similar to #PBS):
    > qsub myscript.sh

```
#!/bin/bash
#COBALT -t 10 -n 8192 -A myproject -O My_Run
runjob --np 131072 -p 16 --block $COBALT_PARTNAME --verbose=INFO :
<executable> <args…>
```

Job dependencies

- Following job in sequence depends on completion of current job

```
> qsub -A MyProject -t 12:00:00 -n 8192 --mode c32 myprogram
** Project 'MyProject'; job rerouted to queue 'prod-capability'
234439
```

```
> qsub -A MyProject -t 12:00:00 -n 8192 --mode c32 myprogram \
         --dependencies 234439
** Project 'MyProject'; job rerouted to queue 'prod-capability'
234440

> qstat -u myusername
JobID   User   WallTime  Nodes  State     Location
================================================
234439  myusername 12:00:00  8192   queued    None
234440  myusername 12:00:00  8192   dep_hold  None
```

Job dependencies (cont'd)

- Job 24439 fails (ends with error code):

```
> qstat -u myusername
JobID   User    WallTime  Nodes  State     Location
================================================
234440  myusername 12:00:00  8192   dep_fail  None
```

- Release 23440 to allow it to run:

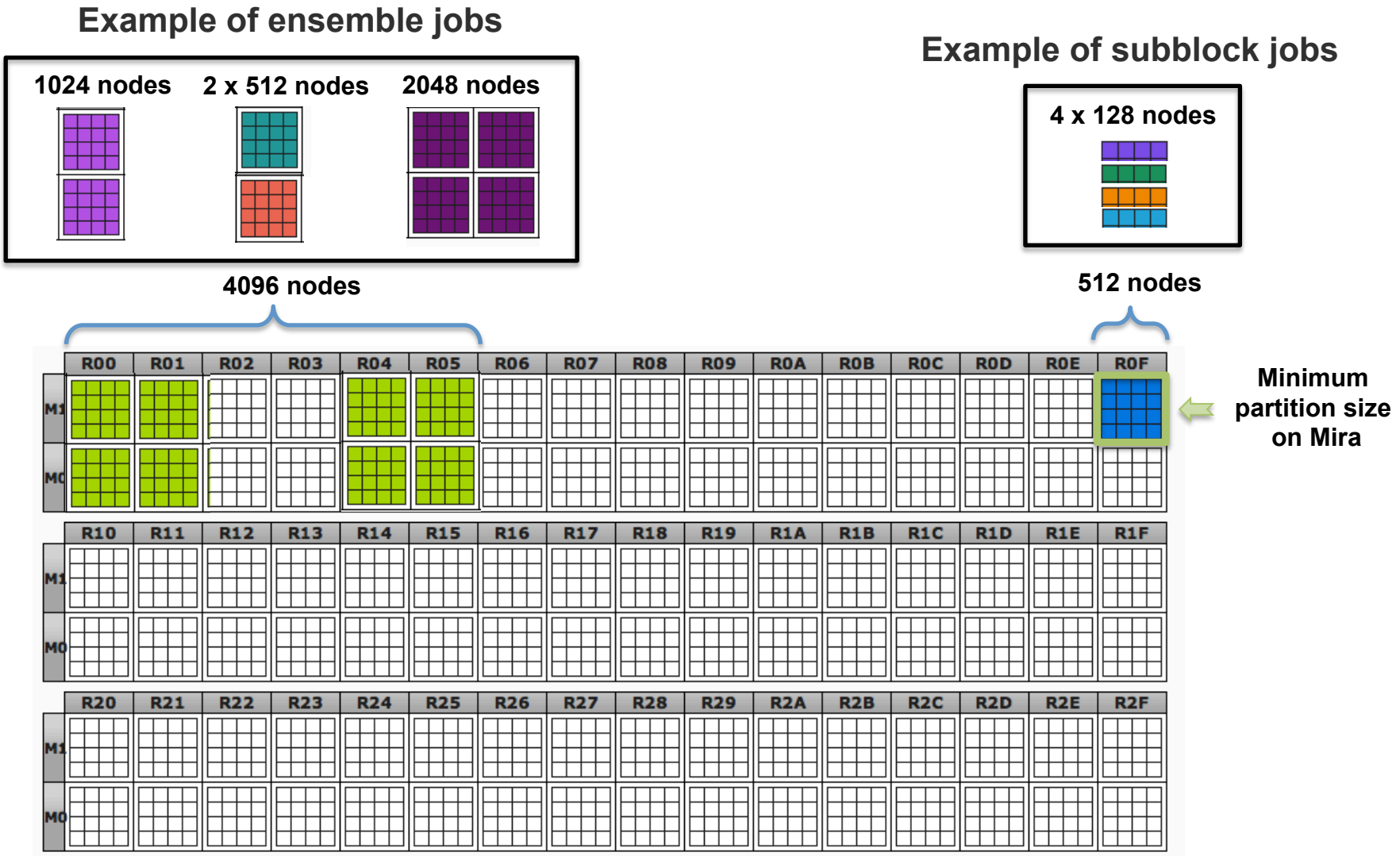
```
> qrls --dependencies 23440

> qstat -u myusername
JobID   User    WallTime  Nodes  State     Location
================================================
234440  myusername 12:00:00  8192   queued  None
```

Advanced runs using script mode

- Multiple (consecutive) runs in a single job
- Multiple simultaneous runs in a single job
- Combinations of the above
- See:
  - http://www.alcf.anl.gov/user-guides/cobalt-job-control
  - http://trac.mcs.anl.gov/projects/cobalt/wiki/BGQUserComputeBlockControl

Advanced runs using ensemble and subblock jobs



http://www.alcf.anl.gov/user-guides/cobalt-job-control

MPI mapping

- A mapping defines the assignment of MPI ranks to BG/Q processors
- Default mapping is *ABCDET*
  - (*ABCDE*) are 5D torus coordinates, *T* is a CPU number
  - Rightmost letter of the mapping increases first as processes are distributed (T then E)
- Mappings may be specified by user using the `RUNJOB_MAPPING` environment variable:
  - With a mapping string:
    - qsub --env RUNJOB_MAPPING=TEDACB --mode c32…
    - String may be any permutation of ABCDET
    - E dimension of torus is always of size 2
  - With a mapping file:
    - qsub --env RUNJOB_MAPPING=<FileName> --mode c32…
    - mapfile: each line contains 6 coordinates to place the task, first line for task 0, second line for task 1…
    - allows for use of any desired mapping
    - file must contain one line per process and not contain conflicts (no verification)
    - use high-performance toolkits to determine communication pattern

http://www.alcf.anl.gov/user-guides/machine-partitions

Reservations

- Reservations allow exclusive use of a partition for a specified group of users for a specific period of time
  - a reservation prevents other users' jobs from running on that partition
  - often used for system maintenance or debugging
  - **R.pm** (preventive maintenance), **R.hw*** or **R.sw*** (addressing HW or SW issues)
  - reservations are sometimes idle, but still block other users' jobs from running on a partition
  - should be the exception not the rule

- Requesting
  - See: http://www.alcf.anl.gov/user-guides/reservations
  - Email reservation requests to ***support@alcf.anl.gov***
  - View reservations with **showres**
  - Release reservations with **userres**

- When working with others in a reservation, these qsub options are useful:
  - --run_users <user1>:<user2>:… All users in this list can control this job
  - --run_project <projectname> All users in this project can control this job

Section:

After your job is submitted

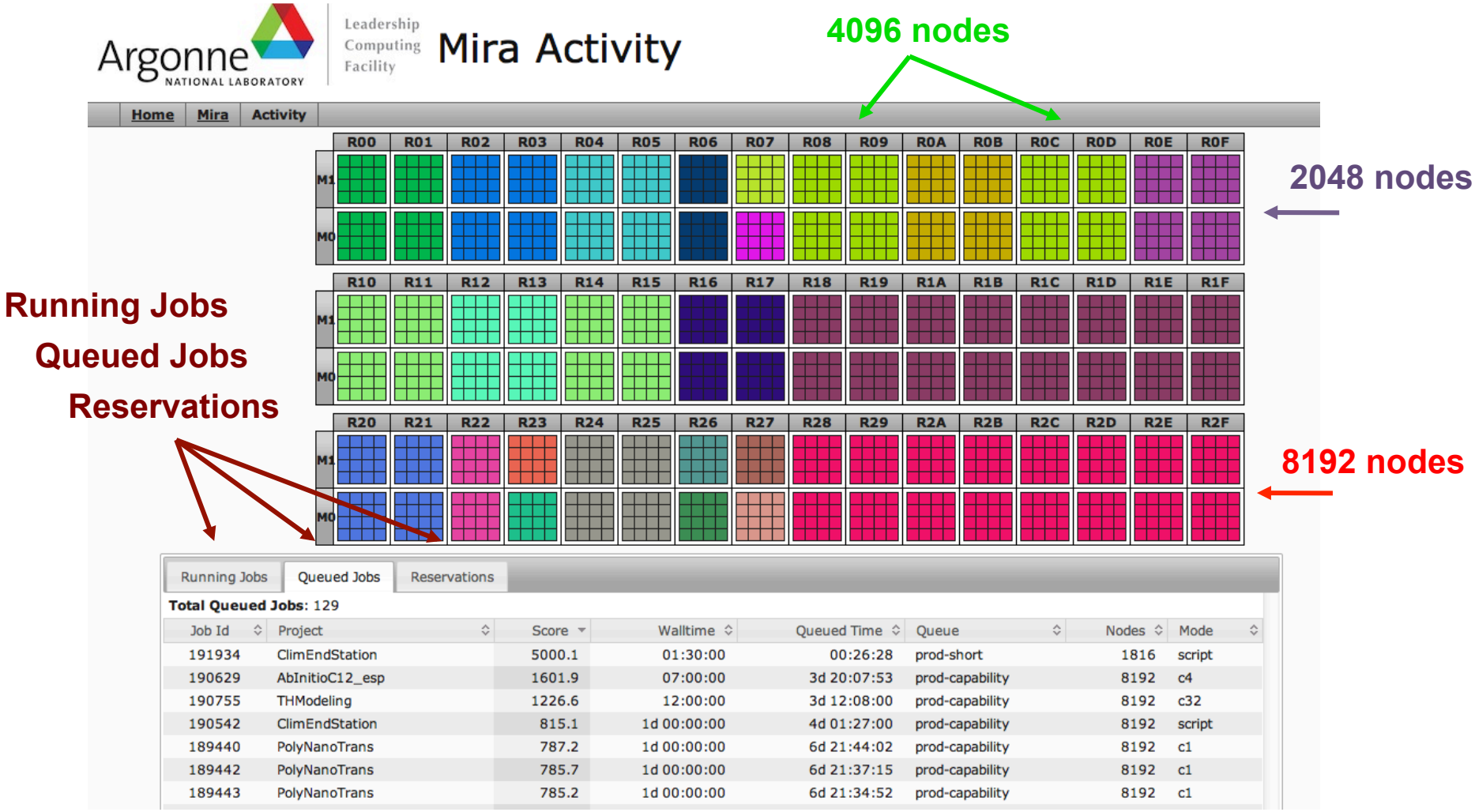
qstat: show status of a batch job(s)

- **qstat** # list all jobs

```
JobID   User     WallTime  Nodes  State     Location
==========================================================
301295  smith    00:10:00   16       queued  None
```

- About jobs
  - JobID is needed to kill the job or alter the job parameters
  - Common states: queued, running, user_hold, maxrun_hold, dep_hold, dep_fail
- **qstat -f <jobid>** # show more job details
- **qstat -fl <jobid>** # show all job details
- **qstat -u <username>** # show all jobs from <username>
- **qstat -Q**
  - Instead of jobs, this shows information about the queues
  - Will show all available queues and their limits
  - Includes special queues used to handle reservations

Machine status web page



Job Id	Project	Score	Walltime	Queued Time	Queue	Nodes	Mode
191934	ClimEndStation	5000.1	01:30:00	00:26:28	prod-short	1816	script
190629	AbInitioC12_esp	1601.9	07:00:00	3d 20:07:53	prod-capability	8192	c4
190755	THModeling	1226.6	12:00:00	3d 12:08:00	prod-capability	8192	c32
190542	ClimEndStation	815.1	1d 00:00:00	4d 01:27:00	prod-capability	8192	script
189440	PolyNanoTrans	787.2	1d 00:00:00	6d 21:44:02	prod-capability	8192	c1
189442	PolyNanoTrans	785.7	1d 00:00:00	6d 21:37:15	prod-capability	8192	c1
189443	PolyNanoTrans	785.2	1d 00:00:00	6d 21:34:52	prod-capability	8192	c1

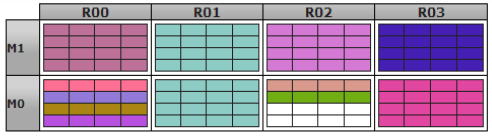
http://status.alcf.anl.gov/mira/activity (a.k.a. The Gronkulator)

Machine status web page (cont'd)

Argonne NATIONAL LABORATORY — Leadership Computing Facility

Cetus Activity

Home | Cetus | Activity



Running Jobs | Queued Jobs | Reservations

Total Running Jobs: 11

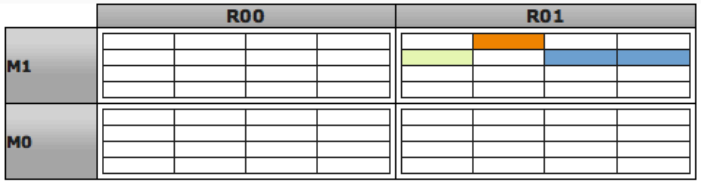
Job Id	Project	Run Time	Walltime	Location	Queue	Nodes	Mode
319157	NRSafety	00:51:20	01:00:00	CET-40040-73371-512	default	512	script
319163	HighLift	00:43:39	01:00:00	CET-40440-73771-512	default	512	script
319166	CombStab_CEF	00:40:34	01:00:00	CET-40400-73731-512	default	512	script
319171	PPI_Entropy	00:28:42	01:00:00	CET-02000-13331-128	default	128	script
319079	PEACEndStation	00:27:41	01:00:00	CET-00400-33771-1024	default	1024	interactive
319174	PPI_Entropy	00:26:21	01:00:00	CET-22000-33331-128	default	128	script
319175	PPI_Entropy	00:26:08	01:00:00	CET-20000-31331-128	default	128	script
319176	PPI_Entropy	00:25:55	01:00:00	CET-00000-11331-128	default	128	script
319177	ctrsp2014	00:11:34	01:00:00	CET-00040-33371-512	default	512	script
319213	ExaHDF5	00:01:32	00:15:00	CET-62000-73331-128	default	1	c4
319214	NucStructReact_2	00:01:06	01:00:00	CET-42000-53331-128	default	128	c16

http://status.alcf.anl.gov/cetus/activity

Argonne NATIONAL LABORATORY — Leadership Computing Facility

Vesta Activity

Home | Vesta | Activity



Running Jobs | Queued Jobs | Reservations

Total Running Jobs: 3

Job Id	Project	Run Time	Walltime	Location	Queue	Nodes	Mode
148459	Excited_States_CNTs	01:06:53	02:00:00	VST-02440-13751-64	default	40	c16
148460	Performance	01:04:27	02:00:00	VST-22460-33571-32	default	32	script
148461	Excited_States_CNTs	00:56:10	02:00:00	VST-02660-13771-32	default	32	c16

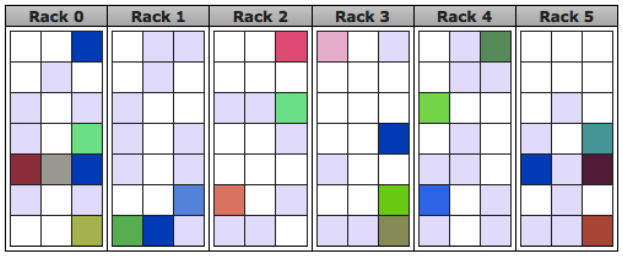
http://status.alcf.anl.gov/vesta/activity

Machine status web page (cont'd)

Argonne NATIONAL LABORATORY | Leadership Computing Facility

Cooley Activity

Home | Cooley | Activity



Running Jobs | Queued Jobs | Reservations

Total Running Jobs: 19

Job Id	Project	Run Time	Walltime	Location
493989	Vendor_Support	07:09:45	12:00:00	cc005,cc007,[cc009-010],cc016,cc018,[cc023-024],cc026,cc028,cc031,[cc033-034],c
494070	SDAV	04:52:08	10:00:00	cc084
494093	OSCon	04:38:38	11:59:00	cc021
494092	OSCon	04:38:27	11:59:00	cc039
494095	OSCon	04:38:16	11:59:00	cc045
494096	OSCon	04:38:05	11:59:00	cc091
494097	OSCon	04:37:54	11:59:00	cc117
494094	OSCon	04:37:43	11:59:00	cc126
494098	OSCon	04:37:32	11:59:00	cc040
494099	OSCon	04:37:22	11:59:00	cc081
494100	OSCon	04:37:11	11:59:00	cc064
494101	OSCon	04:37:00	11:59:00	cc120
494156	SDAV	04:00:01	07:00:00	cc012,cc051
494182	visualization	03:29:09	10:00:00	cc003,cc015,cc041,cc075,cc118
494256	OSCon	00:37:46	11:59:00	cc014
494261	OSCon	00:17:03	11:59:00	cc058
494260	OSCon	00:16:41	11:59:00	cc100
494388	visualization	00:09:54	00:10:00	cc087
494389	visualization	00:09:43	00:10:00	cc013

http://status.alcf.anl.gov/cooley/activity

Cobalt files for a job

- Cobalt will create 3 files per job, the basename <prefix> defaults to the jobid, but can be set with “qsub -O myprefix”
  - jobid can be inserted into your string e.g. "-O myprefix_$jobid"

- Cobalt log file: **<prefix>.cobaltlog**
  - created by Cobalt when job is submitted, additional info written during the job
  - contains submission information from qsub command, runjob, and environment variables
- Job stderr file: **<prefix>.error**
  - created at the start of a job
  - contains job startup information and any content sent to standard error while the user program is running
- Job stdout file: **<prefix>.output**
  - contains any content sent to standard output by user program

qdel: kill a job

- **qdel <jobid1> <jobid2>**
  - delete the job from a queue
  - terminate a running job

qalter, qmove: alter parameters of a Job

- Allows to alter the parameters of queued jobs without resubmitting
  - Most parameters may only be changed before the run starts
- Usage: **qalter** [options] <jobid1> <jobid2> …
- Example:
  > **qalter** -t 60 123 124 125
  (changes wall time of jobs 123, 124 and 125 to 60 minutes)
- Type '**qalter -help**' to see full list of options
- qalter cannot change the queue; use **qmove** instead:
  > **qmove** <destination_queue> <jobid>

qhold, qrls: holding and releasing

- **qhold** - Hold a submitted job (will not run until released)

  qhold <jobid1> <jobid2>

- To submit directly into the hold state, use qsub –h
- **qrls** - Release a held job (in the *user_hold* state)

  qrls <jobid1> <jobid2>

- Jobs in the dep_hold state released by removing the dependency

  qrls --dependencies <jobid>

  or qalter –dependencies none <jobid>

- Jobs in the *admin_hold* state may only be released by a system administrator

Possibilities why a job is not running yet

- There is a reservation, which interferes with your job
  - **showres** shows all reservations currently in place
- There are no available partitions for the requested queue
  - **partlist** shows all partitions marked as functional
  - **partlist** shows the assignment of each partition to a queue

```
Name                    Queue                State                          //
=========================================================================//
…                                                                          //
MIR-04800-37B71-1-1024  prod-short:backfill  busy                           //
MIR-04880-37BF1-1-1024  prod-short:backfill  blocked (MIR-048C0-37BF1-512)  //
MIR-04C00-37F71-1-1024  prod-short:backfill  blocked (MIR-04C00-37F31-512)  //
MIR-04C80-37FF1-1-1024  prod-short:backfill  idle                           //
…                                                                          //
```

- Job submitted to a queue which is restricted to run at this time

Optimizing for queue throughput

- Small (≤ 4K) , long (6h < time < 12h) jobs submitted to prod will be redirected to prod-long, which is restricted to row 0.
- Consider instead:
  - Small (≤ 4K) , short (≤ 6h) jobs in prod queue will be redirected to prod-short, which can run anywhere.
  - Large (> 4K) jobs in prod queue will be redirected to prod-capability, which can run anywhere.
- Shotgun approach:
  - If your code is amenable, submit a mix of job sizes and lengths.
- Check for drain windows:
  - partlist | grep idle
  - full partlist output:

```
Name                    Queue                State                          Backfill  Geometry
=================================================================================================
…
MIR-04800-37B71-1-1024  prod-short:backfill  busy                           -         4x4x4x8x2
MIR-04880-37BF1-1-1024  prod-short:backfill  blocked (MIR-048C0-37BF1-512)  -         4x4x4x8x2
MIR-04C00-37F71-1-1024  prod-short:backfill  blocked (MIR-04C00-37F31-512)  -         4x4x4x8x2
MIR-04C80-37FF1-1-1024  prod-short:backfill  idle                           0:49      4x4x4x8x2
…
```

In this case, a job submitted for 1024 nodes can run immediately if its time is < 49 minutes (might need to be a few minutes shorter to allow for scheduling delay)

Section:

Potential problems

When things go wrong… logging in

- Check to make sure it's not maintenance:
  - Login nodes on Blue Gene/Q and data analytics systems are often closed off during maintenance to allow for activities that would impact users
  - Look for reminders in the weekly maintenance announcement and the pre-login banner message
  - An all-clear email will be sent out at the close of maintenance
- Remember that CRYPTOCard passwords:
  - Require a pin at the start
  - Are all hexadecimal characters (0-9, A-F). Letters are all **UPPER CASE**.
- On failed login, try in this order:
  - Try typing PIN + password again (without generating new password)
  - Try a different ALCF host to rule out login node issues (e.g., maintenance)
  - Push CRYPTOCard button to generate new password and try that
  - Walk through the unlock and resync steps at: http://www.alcf.anl.gov/user-guides/using-cryptocards#troubleshooting-your-cryptocard
  - Still can't login?
    - Connect with **ssh -vvv** and record the output, your IP address, hostname, and the time that you attempted to connect.
    - Send this information in your e-mail to support@alcf.anl.gov

When things go wrong… running

- Cobalt jobs, by default, produce three files (*.cobaltlog, *.error, *.output)
- Only *.cobaltlog is generated at submit time, the others at runtime
- Boot status (successful or not) written to *.cobaltlog
- After booting, the *.error file will have a non-zero size:
  - *Note: If your script job redirects the stderr of cobalt-mpirun, it will not end up in the job's .error file*
- If you think there is an issue, it's best to save all three files:
  - Send the jobid, machine name and a copy of the files to support@alcf.anl.gov

When things go wrong… running

- **RAS** events appearing in your .error file not always a sign of trouble:
  - RAS stands for Reliability, Availability, and Serviceability
- Few are signs of a serious issue, most are system noise:
  - Messages have a severity associated with them:
    - INFO
    - WARN
    - ERROR
    - FATAL
  - Only **FATAL** RAS events will terminate your application
  - **ERROR may degrade performance but will NOT kill your job.**
  - Still worth watching as they may indicate an application performance issue
- If your run exits abnormally, the system will list the last RAS event encountered in the run. **This RAS event did not necessarily cause the run to die.**

Core files

- Jobs experiencing fatal errors will generally produce a core file for each process
- Examining core files:
  - Core files are in text format, readable with the 'more' command
  - **bgq_stack** command provides call stack trace from a core file:
    - Ex: bgq_stack <program_binary> core.*
    - Command line interface (CLI)
    - Can only examine one core file at a time

      http://www.alcf.anl.gov/user-guides/bgqstack
  - **coreprocessor.pl** command provides call stack trace from multiple cores:
    - Ex: coreprocessor.pl -c=<directory_with_core_files> -b=a.out
    - CLI and GUI. GUI interface requires X11 forwarding (ssh -X mira.alcf.anl.gov)
    - Provides information from multiple core files

      http://www.alcf.anl.gov/user-guides/coreprocessor
- Environment variables control core dump behavior:
  - `BG_COREDUMPONEXIT=1` : creates a core dump when the application exits
  - `BG_COREDUMPDISABLED=1` : disables creation of any core files

Getting help

Online resources (24/7):

- ALCF web pages:
  - http://www.alcf.anl.gov
  - http://www.alcf.anl.gov/user-guides
  - https://accounts.alcf.anl.gov
- Mira/Cetus/Vesta/Cooley status (a.k.a. The Gronkulator):
  - http://status.alcf.anl.gov/{mira,cetus,vesta,cooley}/activity

Contact us:

e-mail: **support@alcf.anl.gov**

ALCF Help Desk: **Hours**: Monday – Friday, 9 a.m. – 5 p.m. (Central time)
Phone: **630-252-3111** or **866-508-9181** (toll-free, US only)

Your Catalyst

News from ALCF:

- ALCF Weekly Updates, ALCF newsletters, email via {*mira,cetus,vesta,cooley*}-*notify* lists, etc.

Help Us Help You

- For better, faster results, provide ALCF these details when you contact us for help (where applicable):
  - Machine(s) involved (Mira/Cetus/Vesta/Cooley)
  - Job IDs for any jobs involved
  - Exact error message received
  - Exact command executed
  - Filesystem used when the problem was encountered with path to files
  - Account username and project name that the problem pertains to
  - For connection problems: IP address from which you are connecting
  - Application software name/information

GSW15 Hands-on session

- GSW15 **queue** and **project** name: -q GSW15 -A ALCF_Getting_Started
- **Compilation** example, using basic qsub mode:
  - Create directory, copy files, and compile program:

    > mkdir –p ~/GSW15/Hands-on
    > cp -r /soft/cobalt/examples/compilation ~/GSW15/Hands-on
    > cd ~/GSW15/Hands-on/compilation
    > ls
    > cat hello_mpi.cpp
    > cat Makefile
    > make

  - Submit job using basic mode and check output:

    > qsub -t 5 -n 16 --mode c1 -A ALCF_Getting_Started -q GSW15 -O hello_mpi.output hello_mpi_cpp
    > qstat -u <username>
    > cat hello_mpi.output

  - qsub echoes a number to the screen, which is the Cobalt job id. In the absence of a -o argument, three files are created (say job id was 227076):

    227076.cobaltlog, 227076.error, 227076.output *(replaced by hello_mpi.output with -o)*

GSW15 Hands-on session

- Example of **script job** submission:
  - Copy directory:
    > cp -r /soft/cobalt/examples/script ~/GSW15/Hands-on
    > cd ~/GSW15/Hands-on/script
  - Open the README file and follow the instructions to submit a script job.
  - Look at the runHelloMPI.sh script file. It invokes runjob to run the program.
  - Example:
    > mpixlcxx_r -g -o hello_mpi_cpp hello_mpi.cpp
    > qsub -t 5 -n 16 --mode script -A ALCF_Getting_Started -o hello_mpi.output runHelloMPI.sh

- Example of **interactive job** submission:
  - Copy directory:
    > cp -r /soft/cobalt/examples/interactive ~/GSW15/Hands-on
    > cd ~/GSW15/Hands-on/interactive
  - Open the README file and follow the instructions to submit an interactive job.

GSW15 Hands-on session

- Example of **ensemble** submission:
  - Copy directory:

    ```
    > cp -r /soft/cobalt/examples/ensemble ~/GSW15/Hands-on
    > cd ~/GSW15/Hands-on/ensemble
    ```

  - Open the README file and follow the instructions to submit a 1-rack job with two 512 blocks.

    ***NOTE**: remember to adapt the number of nodes and the block sizes provided in this example to the min./max. partition sizes available in the machine where you want to run the test (see slides 11 for reference).*

- Example of **subblock job** submission:
  - Copy directory:

    ```
    > cp -r /soft/cobalt/examples/subblock ~/GSW15/Hands-on
    > cd ~/GSW15/Hands-on/subblock
    ```

  - Open the README file and follow the instructions to submit a job on multiple 128-node jobs on a midplane on Mira.

- Example of **python job** submission:
  - Copy directory:

    ```
    > cp -r /soft/cobalt/examples/python ~/GSW15/Hands-on
    > cd ~/GSW15/Hands-on/python
    ```

  - Open the README file and follow the instructions to submit a python job.