Los Alamos
NATIONAL LABORATORY

Introduction to Chicoma

Peter Lamborn
Lena M. Lopatina

Chicoma Mountain

- Highest point in the Jemez Mountains (11,561’)
- Mostly within Santa Clara Pueblo
- Visible from many parts of Los Alamos and surrounding area

Agenda

- Basics
  - Getting Assistance
  - About this Training
  - Accessing the Cluster
- System Overview
- Filesystems
- Applications
  - Programming Environment
  - Building Applications
  - Running Applications
- Helpful Tools
- Idiosyncrasies

Basics

- Getting Assistance
- About this Training
- Accessing the Cluster

Getting Assistance

Help Resources Available

- Contact Consult at 665-4444 Option 3 (option 2 is Askit)
  - Monday-Friday 8am-12pm, 1pm-5pm
- Email support also available at consult@lanl.gov
  - Very quick response time
- Consult is in close contact with other teams and can get assistance when subject matter experts are needed
- Documentation available online at hpc.lanl.gov/index.php
  - https://hpc.lanl.gov/platforms/chicoma.html
- You can request help in porting your applications to Chicoma from the ic-help@lanl.gov mailing list

About this Training

About this Training

- This Training assumes you are familiar with
  - Other HPC clusters
  - Basic Linux commands
- After this training you will be able to
  - Access Chicoma
  - Start using Chicoma
- You will still need to
  - Adapt your applications to Chicoma
  - Make changes related to Cray Software
  - Learn GPU specific programming
    - GPU Programming Course
      - February 2nd
      - hpctraining.lanl.gov

Accessing the Cluster

Getting a Chicoma Account

- Part of the IC proposal process
  - https://icp.lanl.gov/
- Principal Investigators add group members
  - https://hpcaccounts.lanl.gov/project_approvers/membership_editor

Logging into Chicoma

- Located in an enclave within the Turquoise network
  - localmachine> `ssh wtrw.lanl.gov`
    - enter cryptocard password
  - wtrw> `ssh ch-fe`

How to Copy Data to Chicoma

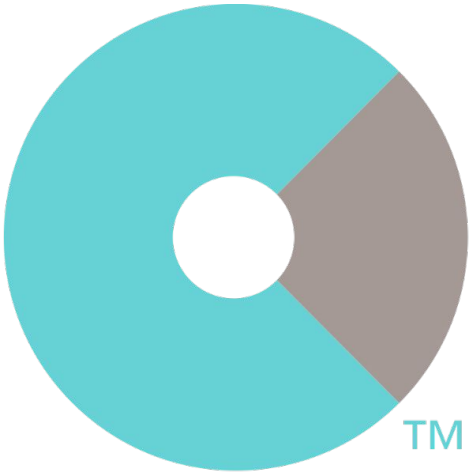
- All transfers must be initiated outside of Chicoma
  - `scp -r <dir> <user>@wtrw:ch-fe:/lustre/scratch5/<user>/.`
  - `scp <user>@wtrw:ch-fe:/lustre/scratch5/<user>/<file> .`
  - `rsync -rLpt -e 'ssh <user>@wtrw ssh' ch-fe:/lustre/scratch5/<user>/<dir> .`
  - `rsync -rLpt <dir> -e 'ssh <user>@wtrw ssh' ch-fe:/lustre/scratch5/<user>/.`

- Lustre directories planned to be cross mounted in February
- Can use ar-tn nodes to move files from scratch4 to scratch5

System Overview

Cray Shasta System

- CrayOS
  - Derived from SUSE
- Cray System Management (CSM)
  - Designed to minimize downtime
  - Fewer operations require full system reboots



Hardware at a High Level

- Two Partitions
  - Standard
    - 2 AMD EPYC 7H12 Processors
  - GPU
    - 1 AMD EPYC 7713 Processor
    - 4 NVIDIA A100 Tensor Core GPUs
- HPE/Cray Slingshot10 interconnect 200Gb/s

Standard Partition

- 2 AMD EPYC 7H12 processors
  - Rome
  - 512 GB RAM per node
  - 64 cores per chip - 128 cores total
  - 2.6 GHz clock rate
- 560 nodes

Comparison to Grizzly

Chicoma	Grizzly
128 Cores	36 Cores
512 GB RAM	128 GB RAM
2.6 GHz	2.1 GHz
L3 Cache: 256MB	45 MB
5.3 TF/s	1.2 TF/s max per node
DDR4-3200	DDR4-2400
8 memory channels	4 memory channels
per socket mem bandwidth: 204.8 GB/s	per socket mem bandwidth: 76.8 GB/s
PCI Express 4.0x128	PCI Express 3.0x(x16 per socket)

GPU Partition per Node:

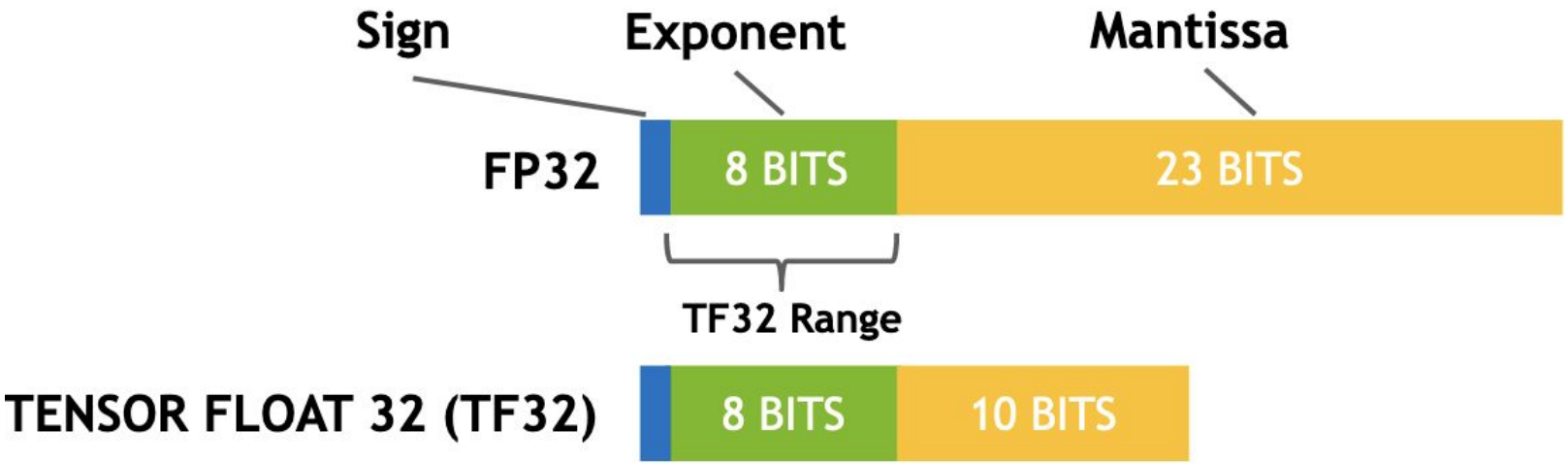
- 1 AMD EPYC 7713 processor
  - Rome
  - 512 GB RAM per node
  - 64 cores
  - 2.0 GHz clock rate
- 4 NVIDIA A100 Tensor Core GPUs
  - Peak FP64 9.7 TFLOPS per GPU
  - Peak FP32 19.5 TFLOPS per GPU
  - GPU memory Bandwidth 1,555 GB/s
- 40 GB of memory per GPU
  - 96 Nodes
- 80 GB of memory per GPU
  - 22 Nodes

Comparison to Kodiak

Chicoma (118 nodes)	Kodiak (133 nodes)
64 Cores	36 Cores
512 GB	275/550 GB
2.0 GHz	2.1 GHz
4 NVIDIA A100s	4 NVIDIA P100s
9.7 TFLOPS (FP64) per A100	5.3 TFLOPS (FP64) per P100
40/80 GB	17 GB
1,555 GB/s	720 GB/s

Tensor Core

- NVIDIA A100s have a Tensor Core
- Separate Hardware focused on Machine Learning applications
- Uses floats with less precision and better speed
- Float: <Sign>*<Mantissa>*10$^{<Exponent>}$



Using Tensor Core

- Must program specifically for the Tensor Core
- cuBLAS
  - https://docs.nvidia.com/cuda/cublas/index.html
- cuTensor
  - https://docs.nvidia.com/cuda/cutensor/index.html
- Warp-level synchronization
  - https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#wmma

Filesystems

Filesystems accessible from Chicoma

- NFS spaces
  - Home directories
  - Project directories
- Scratch (Lustre) space
  - scratch4
  - scratch5

NFS Filesystems on Chicoma

- Same NFS spaces as other Turquoise machines
  - Home directories
  - Project spaces
- Since filesystems are cross mounted, will see same files on other turquoise clusters
- Should be used for
  - Editing files
  - Source code
  - Configuration files
  - Low throughput IO
- .snapshot directories will be accessible

Lustre on Chicoma

- `/lustre/scratch5/<user>`
  - 14 PB capacity
  - Estimated to be mounted on other Turquoise clusters in February
- `/lustre/scratch4/turquoise/<user>`
  - 14 PB capacity
  - Estimated to be mounted on Chicoma in February
- Should be used for
  - Large files
  - Application output
  - Application input
  - High throughput IO
- Files eligible for purging if not accessed for 60 days

Applications

- Programming Environment
- Building Applications
- Running Applications

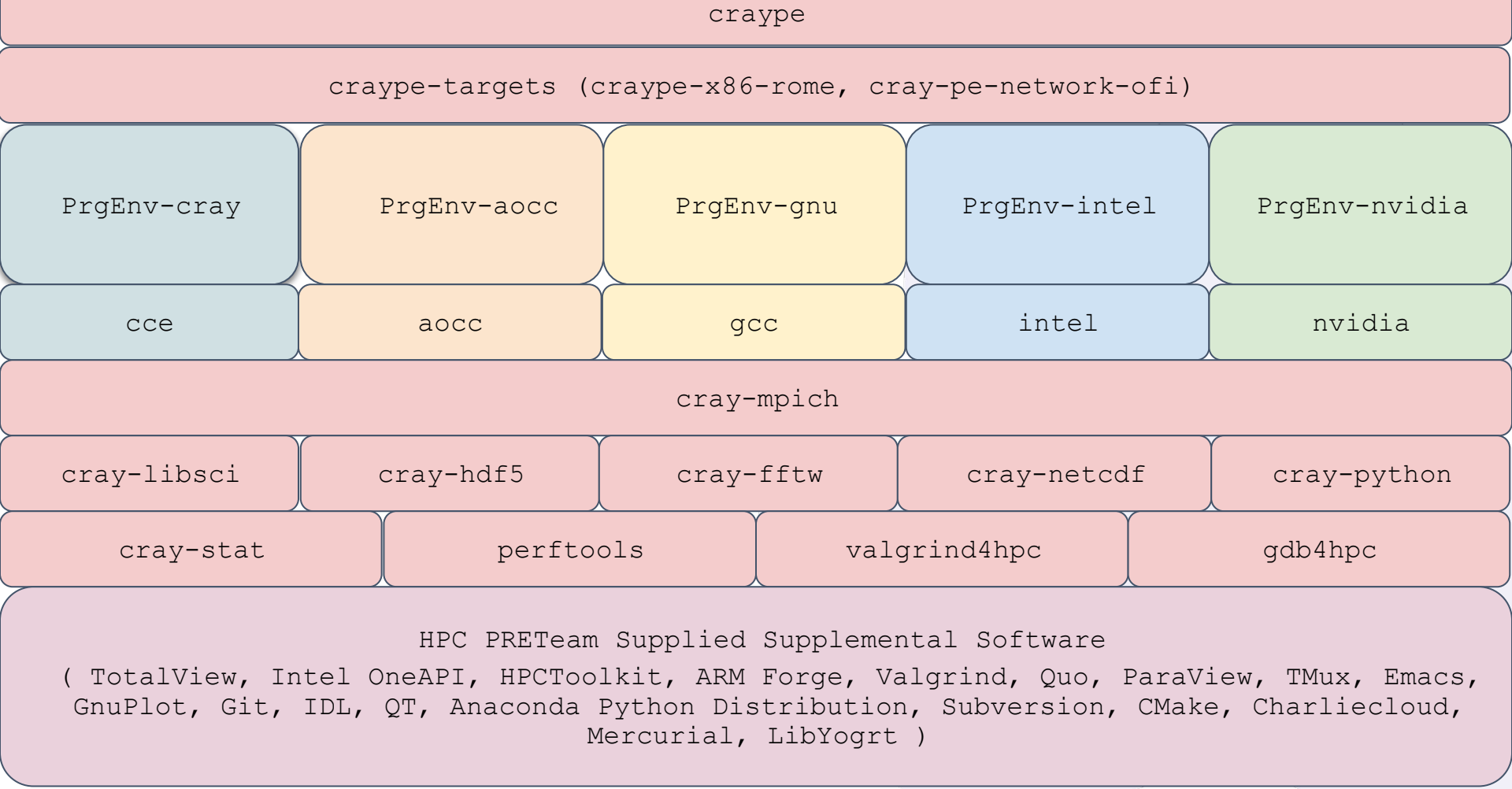
Programming Environment

Programming Environment Modulefile Guide

`module list`	Lists loaded modulefiles
`module avail`	Displays all available modulefiles, their version defaults, and categories
`module load <modulefile>`	Loads modulefile
`module unload <modulefile>`	Unloads modulefile
`module swap <modulefile1> <modulefile2>`	Unloads modulefile1 and loads modulefile2
`module purge`	Unloads all modulefiles from your environment ***(may cause trouble on a Cray, better to swap!)*
`module show <modulefile>`	Information on the modulefile

Programming Environments

- Programming Environments are provided by HPE-Cray:
  - `Cray, AMD Optimizing C/C++ Compiler, Intel OneAPI, GNU, Nvidia-SDK`
- Loading a `PrgEnv-<env>` modulefile loads the CPE environment built in support of the compiler it's based upon (`cray,aocc,intel,gnu,nvidia`)
  - `module swap PrgEnv-<current> PrgEnv-<new>`
- `PrgEnv-cray` is the environment default login environment


craype
craype-targets (craype-x86-rome, cray-pe-network-ofi)
PrgEnv-cray
PrgEnv-aocc
PrgEnv-gnu
PrgEnv-intel
PrgEnv-nvidia
cce
aocc
gcc
intel
nvidia
cray-mpich
cray-libsci
cray-hdf5
cray-fftw
cray-netcdf
cray-python
cray-stat
perftools
valgrind4hpc
gdb4hpc
HPC PRETeam Supplied Supplemental Software
(TotalView, Intel OneAPI, HPCToolkit, ARM Forge, Valgrind, Quo, ParaView, TMux, Emacs, GnuPlot, Git, IDL, QT, Anaconda Python Distribution, Subversion, CMake, Charliecloud, Mercurial, LibYogrt)

PrgEnv Command Examples

```
plamborn@ch-fe2:~> module avail PrgEnv

------------------------------------------------------------------ /opt/cray/pe/modulefiles -------------------------------------------------------------------
PrgEnv-aocc/8.0.0(default)   PrgEnv-cray/8.0.0(default)   PrgEnv-gnu/8.0.0(default)    PrgEnv-intel/8.0.0(default)  PrgEnv-nvidia/8.0.0(default)
plamborn@ch-fe2:~> module list
Currently Loaded Modulefiles:
  1) cce/11.0.4                              5) craype-network-ofi                       9) cray-mpich/8.1.5
  2) craype/2.7.7                            6) cray-dsmml/0.1.5                        10) cray-libsci/21.05.1.1
  3) craype-x86-rome                         7) perftools-base/21.05.0                  11) PrgEnv-cray/8.0.0
  4) libfabric/1.11.0.4.71                   8) xpmem/2.2.40-7.0.1.0_2.7__g1d7a24d.shasta
plamborn@ch-fe2:~> module show PrgEnv-gnu
-------------------------------------------------------------------
/opt/cray/pe/modulefiles/PrgEnv-gnu/8.0.0:

conflict        PrgEnv-amd
conflict        PrgEnv-aocc
conflict        PrgEnv-cray
conflict        PrgEnv-gnu
conflict        PrgEnv-intel
conflict        PrgEnv-nvidia
setenv          PE_ENV GNU
setenv          gcc_already_loaded 0
module          load gcc
module          switch cray-libsci cray-libsci/21.05.1.1
module          switch cray-mpich cray-mpich/8.1.5
module          load craype
module          load craype-x86-rome
module          load craype-network-ofi
module          load cray-dsmml
module          load perftools-base
module          load xpmem
module          load cray-mpich
module          load cray-libsci
setenv          CRAY_PRGENVGNU loaded
```

Building Applications

Commands to Invoke Compilers

- All of the programing environments use the same commands to compile
  - These are the convenience wrappers
- Will use the correct compiler, libraries, linking, and MPI for the current module list
- `cc`
  - C programs
- `CC`
  - C++ programs
- `ftn`
  - Fortran Programs

Compile and Run

```
(base) plamborn@ch-fe1:/lustre/scratch5/plamborn/core_affinity> salloc -N 2 --qos=debug --reservation=debug -p standard
salloc: Granted job allocation 215979
salloc: Waiting for resource configuration
salloc: Nodes nid[001012-001013] are ready for job
bash: /etc/bashrc: No such file or directory
bash: .bashrc: No such file or directory
(base) plamborn@nid001012:/lustre/scratch5/plamborn/core_affinity> cc -fopenmp xthi.c -o xthi
(base) plamborn@nid001012:/lustre/scratch5/plamborn/core_affinity> srun -N 2 -n 4 ./xthi
Hello from rank 1, thread 0, on nid001012. (core affinity = 64-127,192-255)
Hello from rank 1, thread 1, on nid001012. (core affinity = 64-127,192-255)
Hello from rank 3, thread 0, on nid001013. (core affinity = 64-127,192-255)
Hello from rank 3, thread 1, on nid001013. (core affinity = 64-127,192-255)
Hello from rank 0, thread 0, on nid001012. (core affinity = 0-63,128-191)
Hello from rank 0, thread 1, on nid001012. (core affinity = 0-63,128-191)
Hello from rank 2, thread 0, on nid001013. (core affinity = 0-63,128-191)
Hello from rank 2, thread 1, on nid001013. (core affinity = 0-63,128-191)
(base) plamborn@nid001012:/lustre/scratch5/plamborn/core_affinity>
```

Verbose Compilation

- Using the `-v' option will show you what the wrapper is doing
- Can be used for debugging

```
(base) plamborn@nid001016:/lustre/scratch5/plamborn/core_affinity> cc -v -fopenmp xthi.c -o xthi
Cray clang version 13.0.0  (24b043d62639ddb4320c86db0b131600fdbc6ec6)
Target: x86_64-unknown-linux-gnu
Thread model: posix
InstalledDir: /opt/cray/pe/cce/13.0.0/cce-clang/x86_64/share/../bin
Found candidate GCC installation: /opt/cray/pe/gcc/8.1.0/snos/lib/gcc/x86_64-suse-linux/8.1.0
Selected GCC installation: /opt/cray/pe/gcc/8.1.0/snos/lib/gcc/x86_64-suse-linux/8.1.0
Candidate multilib: .;@m64
Selected multilib: .;@m64
 "/opt/cray/pe/cce/13.0.0/cce-clang/x86_64/bin/clang-13" -cc1 -triple x86_64-unknown-linux-gnu -mllvm -cray-omp-opt-fork-
call -mllvm -cray-omp-parallel-opt-call -mllvm -cray-openmp-rename-outlined -fcray-gpu -flocal-restrict -mllvm -cray-enha
nced-asm=1 -fenhanced-asm=1 -mllvm -cray-enhanced-ir=1 -fenhanced-ir=1 -fomp-local-offload-table -ffortran -emit-obj -mre
lax-all --mrelax-relocations -disable-free -main-file-name xthi.c -mrelocation-model static -mframe-pointer=all -fmath-er
rno -fno-rounding-math -mconstructor-aliases -munwind-tables -target-cpu znver2 -debugger-tuning=gdb -v -fcoverage-compil
ation-dir=/lustre/scratch5/plamborn/core_affinity -resource-dir /opt/cray/pe/cce/13.0.0/cce-clang/x86_64/lib/clang/13.0.0
 -isystem /opt/cray/pe/cce/13.0.0/cce-clang/x86_64/lib/clang/13.0.0/include -isystem /opt/cray/pe/cce/13.0.0/cce/x86_64/i
nclude/craylibs -D __CRAY_X86_ROME -D __CRAYXT_COMPUTE_LINUX_TARGET -I /opt/cray/pe/mpich/8.1.11/ofi/cray/10.0/include -I
 /opt/cray/pe/dsmml/0.2.2/dsmml//include -I /opt/cray/pe/libsci/21.08.1.2/CRAY/9.0/x86_64/include -I /opt/cray/xpmem/2.2.
40-7.0.1.0_2.7__g1d7a24d.shasta/include -internal-isystem /opt/cray/pe/cce/13.0.0/cce-clang/x86_64/lib/clang/13.0.0/inclu
de -internal-isystem /usr/local/include -internal-isystem /opt/cray/pe/gcc/8.1.0/snos/lib/gcc/x86_64-suse-linux/8.1.0/../
../../../x86_64-suse-linux/include -internal-externc-isystem /include -internal-externc-isystem /usr/include -fdebug-comp
ilation-dir=/lustre/scratch5/plamborn/core_affinity -ferror-limit 19 -fcray-openmp -fcray-omp-opt-fork -fcray-omp-paralle
l-opt -fcray-openmp-rename-outlined-funcs -fopenmp -fgnuc-version=4.2.1 -fcolor-diagnostics -faddrsig -D__GCC_HAVE_DWARF2
_CFI_ASM=1 -o /tmp/xthi-ae7e30.o -x c xthi.c
clang -cc1 version 13.0.0 based upon LLVM 13.0.0 default target x86_64-unknown-linux-gnu
ignoring nonexistent directory "/opt/cray/pe/gcc/8.1.0/snos/lib/gcc/x86_64-suse-linux/8.1.0/../../../../x86_64-suse-linux
/include"
ignoring nonexistent directory "/include"
ignoring duplicate directory "/opt/cray/pe/cce/13.0.0/cce-clang/x86_64/lib/clang/13.0.0/include"
#include "..." search starts here:
#include <...> search starts here:
```

Running Jobs

Slurm: Partition Selection

- Jobs are Managed through Slurm
- Select Partition
  - Your project may have an allocation on just one partition
  - `-p standard`
  - `-p gpu`
    - GPU partition not yet available
      - Estimated for February
    - To select which memory size
      - `-C gpu80`
      - `-C gpu40`
    - If you do not select
      - gpu40 will be allocated first
      - may get some of both types
  - Have to use account specific to partition
    - `-A <proj_name>` for standard
    - `-A <proj_name>_g` for gpu

Scheduling Limits

- Limits similar to Grizzly
  - Standard QOS
    - no node limit
    - 16 hours
  - Debug QOS
    - 4 nodes
    - 2 hours
    - priority boost
  - Long QOS -- need special permission to use
    - 4 nodes
    - 7 days
  - Standby QOS
    - no node limit
    - 48 hours
    - preemptable
    - either partition
    - less effect on your ‘fairshare’

Slurm: Example Commands

- `salloc -p standard -N 2 -t 2:00:00 --account=<prj_name> --qos=standard`
- `salloc -p gpu -N 8 -C gpu40 -t 12:00:00 --account=<prj_name> --qos=standby`
- `salloc -p standard -N 3 -t 1:00:00 --qos=debug --reservation=debug`
- `srun -n 4 ./<exe name>`

Helpful Tools

Available Profilers

- Tools to analysis the performance of your application
- Cray Perftools
  - The Performance Tools module sets up environments for CrayPat, Apprentice2 and Reveal
  - `` `module load perftools-base; which app2; which pat_run; which reveal` ``
- Allinea MAP
  - `` `module load forge; map` ``
  - GUI or Command Line profiler

Available Debuggers

- Tools to find bugs in your application
- Totalview
  - `module load totalview; totalview`
  - GUI
- Allinea DDT
  - `module load forge; ddt`
  - GUI
- Valgrind
  - `module load valgrind` or `module load valgrind4hpc`
  - Primarily for memory leaks
  - Command line
- STAT
  - GUI or Command Line
- GDB4HPC
  - `module load gdb4hpc; gdb4hpc`
  - Command Line

Chicoma Idiosyncrasies

Cray Systems have Hugepages Modules

- Controls how much memory is read at a time
  - Virtual memory pages
- Regular pages are 4 KB
- Additional sizes available on Chicoma
  - 2 MB 4 MB 8 MB 16 MB 32 MB 64 MB
  - 128 MB 256 MB 512 MB
  - 1 GB 2 GB
- None loaded by default
- `module avail craype-hugepages`

Hugepage Usage

- If your application clusters memory
  - But in larger groups that 4 KB
  - Hugepages are a benefit
  - On previous systems hugepages had an advantage for MPI performance
    - Aries NIC used hugepages exclusively
    - Translated buffer before storing in NIC memory
    - Chicoma currently uses Mellanox chips and will eventually use Cassini
      - Slingshot interconnect
      - Cassini cards may show the same benefit
- **If you compile with any hugepages module**
  - **At runtime, application will use currently loaded size**
- For more information "`man intro_hugepages`"

Other Differences

- Module differences
  - Start with PrgEnv-<type>
  - Use of "`module purge`" is not recommended

- Two partitions
  - `-p standard`
  - `-p gpu`
    - two types of gpu nodes
      - `-C gpu40`
      - `-C gpu80`

Wrap Up

Help Resources Available

- Contact Consult at 665-4444 Option 3 (option 2 is Askit)
  - Monday-Friday 8am-12pm, 1pm-5pm
- Email support also available at consult@lanl.gov
  - Very quick Response time
- Consult is in close contact with other teams and can get assistance when subject matter experts are needed
- Documentation available online at hpc.lanl.gov/index.php
  - https://hpc.lanl.gov/platforms/chicoma.html
- You can request help in porting your applications to Chicoma from the ic-help@lanl.gov mailing list