ARC Documentation

Release 1.0

ARC @ VT

Oct 29, 2021

CONTENTS

This site provides in depth documentation of how to use our resources. For more general information about ARC, see our main site.

CHAPTER

ONE

GETTING STARTED

New to ARC? No problem. See the content below to get started, but don't hesitate to *reach out to us* if you have questions or need assistance.

1.1 The Basics

The following are the basic steps to getting started with ARC resources (and at most research computing centers):

1. Get an account
2. Get an *allocation* (if you are a faculty member/PI) or get access to one (if you are a student)
3. Decide which *hardware* you want to use
4. *Find or install your software*
5. Develop your workflow, possibly via *interactive jobs*
6. Submit your production research via *batch jobs*

In addition to the text documentation linked above, we offer *video tutorials* of most of these steps as well as *training courses* to help people get started.

1.2 Learning Curve

There can be a learning curve in using high-performance computing (HPC) resources. In particular:

- ARC systems run Linux, and traditional use is via the command line. *However*, the latter has become less true in recent years. For example, via *Open OnDemand* ARC users can now access our systems from their browser and start many popular applications such as Jupyter notebooks via the click of a button.
- To run on ARC systems, you must submit your work through the *scheduler*. This is different from running on, e.g., a lab workstation. *However*, this mostly just involves writing down a list of commands you want the system to run and how many resources you want it to use – it is not difficult once you get used to it.
- To leverage HPC resources, your program needs to be able to leverage parallel computing in some way. *However*, may third party programs or libraries exist to make this easier and ARC computational scientists are available if you need assistance.

1.3 Familiar with HPC, new to ARC

If you are an experienced HPC user who is new to ARC, you may just need to know the following:

- ARC uses the *Slurm scheduler*.
- ARC uses EasyBuild for software *modules*.
- You will need to have an *allocation* to charge your jobs to. This is free of charge unless you would like to invest in priority access.
- Descriptions of our compute and storage resources can be found *here*.

1.4 Training

To help users get started, we offer introductory training sessions throughout the year via the Professional Development Network. Our computational scientists are also available for classroom presentations on high-performance, parallel, scientific, or other research computing topics – this is a great way to get a research group up to speed.

If you prefer to do things at your own pace, we offer *video tutorials* that walk through each of the steps of getting started with ARC.

1.5 Getting Help

If you are interested in using ARC's resources for your current or future projects, or if you would just like to learn more about our computing systems and services, please request a consultation or drop by our office hours. You do not need to have any prior experience with high-performance computing — our team can assist you in determining the right system for your project.

CHAPTER TWO

INFORMATION FOR FACULTY/PROJECT PIS

The following pages provide information that might be of specific use to faculty members or other project Principal Investigators (PIs).

2.1 Citations

Recognition and documentation of the contribution that ARC's systems play in breakthrough research is essential to ensuring continued support for and availability of cutting-edge computing resources at Virginia Tech. Please cite Advanced Research Computing at Virginia Tech in any research report, journal article, or other publication that requires citation of an author's contributions.

Suggested verbiage:

> The authors acknowledge Advanced Research Computing at Virginia Tech for providing computational resources and technical support that have contributed to the results reported within this paper. URL: https://arc.vt.edu/

2.2 Cost Center

2.2.1 Intent

The Cost Center provides researchers or projects with the ability to purchase computational or storage resources beyond *what ARC provides for free*, for computational "bursts" to meet, e.g., conference deadlines, or short-term storage of large datasets. It provides:

- Compute or storage beyond the *free tier*
- Priority quality of service (QOS) for faster job execution
- PI-specified sub-account limits
- Requestable through ColdFront

The program is also intended to provide the accounting infrastructure to allow PIs to include access to resources in grant proposals and contracts.

If you would like to get access to dedicated computational resources or long-term expansion of storage, you may want to instead consider the *Investment Program*.

2.2.2 Free Tier

ARC is working to decrease HPC cost to VT, improve access, services and augment VT's research and teaching missions. As part of this, we are realigning ARC to more naturally support research groups (and class groups). Starting on *TinkerCliffs*, the Division of IT provides the following resources for each ARC user account free of charge:

Category	User	PI (project request)
Compute	240 core-hours/month	600,000 core-hours/month (*TinkerCliffs* only)
Home storage	640 GB	–
User workspace storage	1 TB	–
Project storage	–	25 TB
Archive storage	/vtarchive/home/*pid*	/vtarchive/groups/*group*

Allocations can also be submitted for class needs; these are "owned" by ARC and not billed toward a PI's account.

Note: Jobs submitted to preemptable partitions do *NOT* count against the above user/project limits.

2.2.3 Job Priority

Priority determines position in "line":

Quality of Service (QoS)	Available by/through:
priority (high)	for-fee via cost-center
normal (default)	normal

2.2.4 Current Cost Structure

TinkerCliffs

The fee structure on *TinkerCliffs* is as follows:

Queue	Cost
normal_q	$0.0023 / core-hour
largemem_q	$0.01 / core-hour
intel_q	$0.0091 / core-hour

Storage and other available resources

Temporary expansion of `/project` *storage* can be requested. This will be billed at $2.1694 per TB per month.

For server hosting, enterprise backup or other needs, please send Terry Herdman an email.

2.3 Facilities, Equipment, and Other Resources Statement

The following is a draft Facilities, Equipment and Other Resources statement that researchers can include in research proposals:

Computing resources will be provided through Advanced Research Computing (ARC) within the Division of Information Technology at Virginia Tech. ARC provides cutting-edge high-performance computing and visualization resources. Currently available high performance computing (HPC) systems include:

1. **TinkerCliffs**: a general purpose CPU cluster. This cluster has approximately 40,000 AMD Rome CPU cores, HDR Infiniband offering 100 Gbps throughput, nodes for high-memory applications, an additional 16 Intel Xeon AP nodes and four nodes with eight NVIDIA A100-80GB GPUs each
2. **Infer**: GPU-based cluster made up of 58 compute nodes with a total of 4 NVIDIA Volta V100 GPUs, 18 NVIDIA Tesla T4 GPUs, and 80 NVIDIA Tesla P100 GPUs; Infiniband interconnect
3. **Cascades**: General purpose cluster with 190 compute nodes equipped with two 16-core Intel Xeon “Broadwell” CPU and 128 GB of memory; 38 compute nodes equipped with two 12-core Intel Xeon Skylake CPU, 376 GB of memory, and two NVIDIA V100 GPU; 4 compute nodes with two NVIDIA K80 GPU, 512 GB of memory and one 2 TB NVMe flash card; 2 four-socket compute nodes with four 18-core Intel Xeon “Broadwell” CPU and 3 TB of memory; Mellanox EDR Infiniband interconnect
4. **Dragonstooth**: High-throughput cluster with 48 two-socket compute nodes equipped with two 12-core Intel Xeon “Haswell” CPU, 256 GB of memory and four 480GB SSD drives

Parallel filesystems provide over 11 Petabytes of high performance storage, and a tape archive is provided to support long term data storage.

ARC’s Visionarium Lab also provides an array of visualization resources, including the VisCube, an immersive 10 x 10 three-dimensional visualization environment. In all, the VT Visionarium provides nearly 86 million pixels, 4 billion triangles-per-second and 22 TB/s of GPU memory bandwidth. ARC resources are able to leverage Virginia Tech’s excellent network connectivity, and network. Virginia offers access to advanced national networks, including ESnet, Internet2, and Mid Atlantic Crossroads.

Upcoming Resources

In the next year, ARC plans to release additional resources supporting:

1. **Protected data**: This will be a dedicated cluster and storage supporting data needing elevated protections. This cluster will be available early in the winter of 2021-2022.
2. **AI/ML**: Additional nodes will be added to the TinkerCliffs cluster to support AI/ML applications. Scheduled to be released in late Spring 2022.
3. **Cloud**: Kubernetes resource for cloud-like applications.

2.4 Investment Program

2.4.1 Intent

The investment program is for researchers or projects who want dedicated resources from ARC over some period of time:

- For long-term (1-5 year) project needs
- Reserved compute hardware via dedicated partition (with preemptable overlay)
- Expansion of Project or Work via quota increase

- Available via MOU

If you are less interested in *dedicated* hardware and just want to use more resources than ARC provides for free – for example, in bursts before conference deadlines – you might instead consider the *Cost Center*.

2.4.2 Memorandum of Understanding (MOU)

An investment Memoradum Of Understanding (MOU) is updated and made available for each new cluster as it comes online. The current MOU covers *TinkerCliffs*.

For example investment MOUs, see below:

- Compute
- Storage

2.4.3 To Invest

If you have interest in learning more about the Investment Computing Program, please submit a consultation request. ARC can provide a brief presentation on the Investment Computing program at department meetings or to research teams if desired.

Contents:

- ColdFront: Interface for requesting compute or storage *allocations*
- *Cost Center*: Description of ARC's cost center program if you need more resources than ARC provides for free (even in short intervals for conference deadlines, etc)
- *Investment Program*: Description of ARC's investment program if you want to acquire a dedicated portion of one of ARC's systems
- *FE&R Statement*: Facilities, Equipment, and Other Resources statement for inclusion in proposals
- *Citations*: Example acknowledgement of ARC for inclusion in papers that were prepared with the help of our systems

CHAPTER THREE

RESOURCES

Contents:

3.1 Computational Resources

Contents:

3.1.1 TinkerCliffs, ARC's Flagship Resource

Overview

TinkerCliffs came online in the summer of 2020. With nearly 42,000 cores and over 93 TB of RAM, TinkerCliffs is nearly seven times the size of BlueRidge, ARC's previous flagship CPU compute system, which was retired at the end of 2019. TinkerCliffs hardware is summarized in the table below.

	Base Compute Nodes	High Memory Nodes	Intel Nodes	A100 GPU Nodes	Total
Vendor	Cray	Cray	HPE	HPE Apollo 6500	-
Chip	AMD EPYC 7702	AMD EPYC 7702	Intel Xeon Platinum 9242	AMD EPYC 7742	-
Nodes	308	8	16	4	336
Accelerators	-	-	-	8x NVIDIA A100-80G	-
Cores/Node	128	128	96	128	-
Memory (GB)/Node	256	1,024	384	2048	-
Total Cores	39,424	1,024	1,536	512	42,496
Total Memory (GB)	78,848	8,192	6,144	8192	101,376
Local Disk	480GB SSD	480GB SSD	3.2TB NVMe	11.7TB NVMe	-
Interconnect	HDR-100 IB	HDR-100 IB	HDR-100 IB	4x HDR-200 IB	-

Tinkercliffs is hosted in the Steger Hall HPC datacenter on the Virginia Tech campus, so it is physically separated from other ARC HPC systems which are hosted in the AISB Datacenter at the Corporate Research Center (CRC) in Blacksburg.

For HPC, it is important that file systems (data storage) be physically near to the compute systems, so there is not direct connectivity from Tinkercliffs to some of the legacy filesystems (eg. GPFS /groups and /work). The /home filesystem on Tinkercliffs is the same as on legacy clusters, but for the reasons stated above, should not be used for i/o intensive workloads.

A BeeGFS file system supports /projects and /work filesystems for group collaboration and high-performance input/output (I/O).

A100 GPU Nodes

Four nodes nodes equipped with GPU accelerators were added to Tinkercliffs in June 2021. Each of these nodes is designed to be a clone of NVIDIA's DGX nodes to provide a dense GPU resource for the VT research computing community. The eight NVIDIA A100-80G GPUs in each node are interconnected with NVIDIA's NVLink technology. For internode communications, each chassis is equipped with four Mellanox HDR-200 Infiniband cards distributed across the PCIe Gen4 bus to provide each GPU with a nearby, high speed, low latency, path to the Infiniband network.

Get Started

Tinkercliffs can be accessed via one of the two login nodes:

`tinkercliffs1.arc.vt.edu` `tinkercliffs2.arc.vt.edu`

For testing purposes, all users will be alloted 240 core-hours each month in the "personal" allocation. Researchers at the PI level are able to request resource allocations in the "free" tier (usage fully subsidized by VT) and can allocate 600,000 monthly billing units (normal_q core-hours) among their projects.

To do this, log in to the ARC allocation portal https://coldfront.arc.vt.edu,

- select or create a project
- click the "+ Request Resource Allocation" button
- Choose the "Compute (Free) (Cluster)" allocation type

Usage needs in excess of 600,000 monthly billing units can be purchased via the ARC Cost Center.

Policies

Limits are set on the scale and quantity of jobs at the user and allocation (Slurm account) levels to help ensure availability of resources to a broad set of researchers and applications. These are the limits applied to free tier usage (note that the terms "cpu" and "core" are used interchangably here following Slurm terminology):

	normal_q	dev_q	largemem_q	intel_q	a100_normal_q	a100_dev_q	interactive_q	preemptable_q
Node Type	Base Compute	Base Compute	High Memory	Intel	A100 GPU	A100 GPU	Base Compute	Base Compute
Billing Weight	1	1	4.045454/core	3.772727/core	155.26/GPU	155.26/GPU	0.25/core	0 (no billing)
Number of Nodes	302	307	8	16	4	4	2	307
MaxRunningJobs (User)	24	2	4	8	12	2	4	64
MaxSubmitJobs (User)	240	3	40	80	48	8	4	640
MaxRunningJobs (Allocation)	48	3	6	12	24	4	-	128
MaxSubmitJobs (Allocation)	480	6	60	120	48	8	-	1280
MaxNodes (User)	64	64	4	8	1	1	-	-
MaxNodes (Allocation)	96	96	6	12	2	2	-	-
MaxCPUs (User)	8192	8192	512	768	128	128	64	-
MaxCPUs (Allocation)	12288	12288	768	1152	256	256	128	-
MaxGPUs (User)	-	-	-	-	8	8	-	-
MaxGPUs (Allocation)	-	-	-	-	16	16	-	-
MaxWallTime	6 days	4 hours	6 days	6 days	6 days	4 hours	4 hours	-
Free allowance at Max[CPU/GPU]s (User)	3.05 days	3.05 days	12.07 days	8.63 days	20.13 days	-	-	-
Free allowance at Max[CPU/GPU]s (Alloc)	2.03 days	2.03 days	9.05 days	6.47 days	10.06 days	-	-	-

Tinkercliffs is part of the ARC cost center, which provides a substantial "free tier" of usage. Each researcher is provided 600,000 billing units (1 billing unit = 1 TC normal_q core-hour) which can be divided among all projects and allocations they own. Monthly billing is based on usage attributed to jobs which complete in that month, so jobs which start in month A and finish in month B are billed in month B.

Modules

TinkerCliffs is different from previous ARC clusters in that it uses a new application stack/module system based on EasyBuild. Our old application stack was home-grown and involved a fair amount of overhead in getting new modules - e.g., new versions of a package - installed. EasyBuild streamlines a lot of that work and should also make it trivial in some cases for users to install their own versions of packages if they so desire. Key differences from a user perspective include:

- Hierarchies are replaced by toolchains. Right now, there are two:
  - foss ("Free Open Source Software"): gcc compilers, OpenBLAS for linear algebra, OpenMPI for MPI, etc
  - intel: Intel compilers, Intel MKL for linear algebra, Intel MPI

- Instead of loading modules individually (e.g., module load intel mkl impi), a user can just load the toolchain (e.g., `module load intel/2019b`).
- Modules load their dependencies, e.g.,

```
$ module reset; module load HPL/2.3-intel-2019b; module list

  Currently Loaded Modules:
    1) shared                                 6) crayype-x86-rome           11) binutils/2.
↪32-GCCcore-8.3.0             16) intel/2019b
    2) slurm/20.02.3                          7) craype-network-infiniband  12) iccifort/
↪2019.5.281                     17) HPL/2.3-intel-2019b
    3) apps                                   8) DefaultModules             13) impi/2018.5.
↪288-iccifort-2019.5.281
    4) site/tinkercliffs/easybuild/setup      9) GCCcore/8.3.0              14) iimpi/2019b
    5) cray                                  10) zlib/1.2.11-GCCcore-8.3.0  15) imkl/2019.5.
↪281-iimpi-2019b
```

- All modules are visible with `module avail`. So in many cases it is probably better to search with `module spider` rather than printing the whole list.
- Some key system software, like the Slurm scheduler, are included in default modules. This means that `module purge` can break important functionality. Use `module reset` instead.
- Lower-level software is included in the module structure (see, e.g., `binutils` in the HPL example above), which should mean less risk of conflicts in adding new versions later.
- Environment variables (e.g., `$SOFTWARE_LIB`) available in our previous module system may not be provided. Instead, EasyBuild typically provides `$EBROOTSOFTWARE` to point to the software installation location. So for example, to link to NetCDF libraries, one might use `-L$EBROOTNETCDF/lib64` instead of the previous `-L$NETCDF_LIB`.

Architecture

- The AMD Rome architecture is similar to Cascades in that it is x86_64 but lacks the AVX-512 instruction set added to Intel processors in the last couple of years.
- Nodes are larger (128 cores) and have more memory bandwidth (~350 GB/s).
- There are eight NUMA (memory locality) domains per node and one L3 cache for every four cores.

Optimization

See also the tuning guides available at https://developer.amd.com/, especially this guide to compiler flags.

- Cache locality really matters - process pinning can make a big difference on performance.
- Hybrid programming often pays off - one MPI process per L3 cache with 4 threads is often optimal.

Intel toolchain:

- Fast, though our testing has found that v2020 is slower than v2019
- Avoid `-xhost`
- Use `-march=core-avx2` to get the optimal vectorization instruction set
- Use the following environment variables for MKL (we set these as part of the MKL module):

```
export MKL_DEBUG_CPU_TYPE=5
export MKL_ENABLE_INSTRUCTIONS=AVX2
```

Foss (GCC) toolchain:

- Use `-mtune=znver2 -march=znver2` to target the Zen2 architecture
- Use `-mavx2` to get the optimal vectorization instruction set

AOCC Compiler:

- AMD compiler. Very fast on Rome architectures. ARC is working on getting AOCC integrated into a toolchain.
- Use `-mtune=znver2 -march=znver2` to target the Zen2 architecture
- Use `-mavx2` to get the optimal vectorization instruction set

Examples

See below for a series of examples of how to compile code for a variety of compilers and for how to run optimally in a variety of configurations. These and a wide variety of simple application-specific examples can be found *in our examples repository*.

Stream

STREAM is a memory bandwidth benchmark. To maximize bandwidth, we run in parallel with one process per L3 cache (cores 0, 4, . . . , 124).

```
#Load the Intel toolchain
module reset; module load intel/2019b

#Tell OpenMP to use every 4th core
export OMP_PROC_BIND=true
export OMP_NUM_THREADS=32
export OMP_PLACES="$( seq -s },{ 0 4 127 | sed -e 's/\(.*\)/\{\1\}/' )"

#Compile
icc -o stream.intel stream.c -DSTATIC -DNTIMES=10 -DSTREAM_ARRAY_SIZE=2500000000 \
  -mcmodel=large -shared-intel -Ofast -qopenmp -ffreestanding -qopt-streaming-stores␣
↪always

#Run
./stream.intel
```

Results:

```
Function    Best Rate MB/s
Copy:             341475.1
Scale:            341770.0
Add:              336668.3
Triad:q:          336972.6
```

MT-DGEMM

mt-dgemm is a threaded matrix multiplication program that can be used to benchmark dense linear algebra libraries. Here we use it to show how to link against linear algebra libraries and run efficiently across a socket.

AOCC

```
#Load the aocc and blis modules
module reset; module load aocc/aocc-compiler-2.1.0 amd-blis/aocc/64/2.1

#Compile:
#  Build for the Rome architecture: -mtune=znver2 -march=znver2
#  Use fast vectorization: -mavx2
#  Use math libraries: -lm
#  Use OpenMP: -fopenmp -lomp
#  Other optimizations: -Ofast -ffp-contract=fast -funroll-loops
#  Link with AMD BLIS linear algebra library: -I$BLISDIR/../include $BLISDIR/libblis-mt.a
#  Macro used by the mt-dgemm program: -D USE_CBLAS
clang -mtune=znver2 -march=znver2 -mavx2 -lm -fopenmp -lomp -Ofast -ffp-contract=fast -
↪funroll-loops -I$BLISDIR/../include $BLISDIR/libblis-mt.a -D USE_CBLAS -o mt-dgemm.
↪aocc mt-dgemm.c

#Run with 64 OpenMP threads on cores 0-63 (socket 1) using NUMA memory regions 0-3␣
↪(socket 1). This keeps Linux from moving the threads away from memory.
OMP_NUM_THREADS=64 GOMP_CPU_AFFINITY=0-63:1 numactl --membind=0-3 ./mt-dgemm.aocc 16000
```

GCC

```
#Load the foss toolchain
module reset; module load foss/2020a

#Compile:
#  Build for the Rome architecture: -mtune=znver2 -march=znver2
#  Use fast vectorization: -mavx2
#  Use math libraries: -lm
#  Use OpenMP: -fopenmp
#  Other optimizations: -Ofast -ffp-contract=fast -funroll-loops
#  Link with OpenBLAS linear algebra library: -L$OPENBLAS_LIB -lopenblas
#  Macro used by the mt-dgemm program: -D USE_CBLAS
gcc -mtune=znver2 -march=znver2 -mavx2 -lm -fopenmp -Ofast -ffp-contract=fast -funroll-
↪loops -L$OPENBLAS_LIB -lopenblas -D USE_CBLAS -o mt-dgemm.gcc mt-dgemm.c

#Run with 64 OpenMP threads on the cores (0-63) and memory (regions 0-3) associated with␣
↪socket 1. This keeps Linux from moving the threads away from memory. Using GOMP_CPU_
↪AFFINITY to pin thread 0 to core 0, thread 1 to core 1, etc would be ideal but breaks␣
↪the threading in OpenBLAS for whatever reason.
OMP_NUM_THREADS=64 numactl -C 0-63 --membind=0-3 ./mt-dgemm.gcc 16000
```

Intel

Here we use intel 2019 as testing indicates that 2020 is substantially slower.

```
#Load the intel toolchain
module reset; module load intel/2019b

#Note that the module has set MKL_ENABLE_INSTRUCTIONS=AVX2 and MKL_DEBUG_CPU_TYPE=5
 to ensure that MKL uses the optimal instruction set
env | egrep "MKL_DEBUG_CPU_TYPE|MKL_ENABLE_INSTRUCTIONS"

#Compile:
#  Use fast vectorization: -march=core-avx2
#  Use OpenMP: -qopenmp
#  Other optimizations: -O3 -ffreestanding
#  Link with MKL linear algebra library: -mkl
#  Macro used by the mt-dgemm program: -D USE_MKL=1
icpc -march=core-avx2 -qopenmp -O3 -ffreestanding -mkl -D USE_MKL=1 -o mt-dgemm.intel mt-
↪dgemm.c

#Run with 64 threads on cores 0-63 (socket 1) using NUMA memory regions 0-3 (socket 1).␣
↪This keeps Linux from moving the threads away from memory.
MKL_NUM_THREADS=64 GOMP_CPU_AFFINITY=0-63:1 numactl --membind=0-3 ./mt-dgemm.intel 16000
```

Results

The results show the benefits of AMD's optimizations and of MKL's performance over OpenBLAS:

```
  aocc+blis 2.1:  1658.861832 GF/s
  foss/2020a:     1345.527671 GF/s
  intel/2019b:    1615.846327 GF/s
```

HPL

HPL is a computing benchmark. Here we use it to demonstrate how to run in the pure MPI (1 process per core) and hybrid MPI+OpenMP (1 process per L3 cache with 4 OpenMP threads working across the cache) models. To load the HPL module, we can do simply

```
  module reset; module load HPL/2.3-intel-2019b  #intel
  module reset; module load HPL/2.3-foss-2020a   #gcc
```

MPI Only (1 MPI process/core)

Here we use pure MPI and start one MPI process per core. Jobs in this case should typically be requested with –ntasks-per-node=128 (if you want full node performance).

- Intel, using mpirun. We use an environment variable to make sure that MPI processes are laid out in order and not moved around by the operating system.

```
mpirun -genv I_MPI_PIN_PROCESSOR_LIST=0-127 xhpl
```

- gcc, using mpirun. Here we use OpenMPI's mapping and binding functionality to assign the processes to consecutive cores.

```
mpirun --map-by core --bind-to core -x OMP_NUM_THREADS=1 xhpl
```

- Intel or gcc, using srun. We use srun's cpu-bind flag to bind the processes to cores.

```
srun --cpu-bind=cores xhpl
```

Hybrid MPI+OpenMP (1 MPI process/L3 cache)

Here we start one MPI process per L3 cache (every 4 cores). Jobs in this case should typically be requested with –ntasks-per-node=32 –cpus-per-task=4 so that Slurm knows how many processes you need.

- Intel, using mpirun. We use environment variables to tell mpirun to start a process on every fourth core and use 4 OpenMP (MKL) threads per process:

```
mpirun -genv I_MPI_PIN_PROCESSOR_LIST="$( seq -s , 0 4 127 )" -genv I_MPI_PIN_
↪DOMAIN=omp -genv OMP_NUM_THREADS=4 -genv OMP_PROC_BIND=TRUE -genv OMP_PLACES=cores xhpl
```

- gcc, using mpirun. Here we use OpenMPI's mapping and binding functionality to assign the processes to L3 caches.

```
mpirun --map-by ppr:1:L3cache --bind-to l3cache -x OMP_NUM_THREADS=4 xhpl
```

- Intel or gcc, using Slurm's srun launcher. We use a cpu mask to tell Slurm which cores each process should have access to. (0xF is hexadecimal for 15, or 1111 in binary, meaning access should be allowed to the first four cores. 0xF0 is 11110000 in binary, meaning access should be allowed to the second set of four cores. The list continues through 11110000…..0000, indicating that the last process should have access to cores 124-127.)

```
srun --cpu-bind=mask_cpu=0xF,0xF0,0xF00,0xF000,0xF0000,0xF00000,0xF000000,0xF0000000,
↪0xF00000000,0xF000000000,0xF0000000000,0xF00000000000,0xF000000000000,0xF0000000000000,
↪0xF00000000000000,0xF000000000000000,0xF0000000000000000,0xF00000000000000000,
↪0xF000000000000000000,0xF0000000000000000000,0xF00000000000000000000,
↪0xF000000000000000000000,0xF0000000000000000000000,0xF00000000000000000000000,
↪0xF000000000000000000000000,0xF0000000000000000000000000,0xF00000000000000000000000000,
↪0xF000000000000000000000000000,0xF0000000000000000000000000000,
↪0xF00000000000000000000000000000,0xF000000000000000000000000000000,
↪0xF0000000000000000000000000000000 xhpl
```

Results

The results show the benefit of the hybrid MPI+OpenMP model and of MKL over OpenBLAS, particularly in the hybrid model.

```
 intel |     mpi   | mpirun | 2,944 GFlops/s
 intel |     mpi   |   srun | 2,809 GFlops/s
   gcc |     mpi   | mpirun | 2,734 GFlops/s
   gcc |     mpi   |   srun | 2,659 GFlops/s
 intel | mpi+omp   | mpirun | 3,241 GFlops/s
 intel | mpi+omp   |   srun | 3,227 GFlops/s
   gcc | mpi+omp   | mpirun | 2,836 GFlops/s
   gcc | mpi+omp   |   srun | 2,845 GFlops/s
```

3.1.2 Infer, GPU Cluster

Overview

Infer came online in January of 2021 and provides 18 nodes, each with an Nvidia T4 GPU. The cluster's name "Infer" alludes to the AI/ML inference capabilities of the T4 GPUs derived from the "tensor cores" on these devices. We think they will also be a great all-purpose resource for researchers who are making their first forays into GPU-enabled computations of any type.

In the spring of 2021, 40 nodes with two Nvidia P100 GPUs each were migrated from a older ARC system which was being decommissioned.

Technical details are below:

Vendor	HPE	Dell
Chip	Intel Xeon Gold 6130	Intel Xeon E5-2680v4 2.4GHz
Nodes	18	40
Cores/Node	32	28
GPU Model	Nvidia Tesla T4	Nvidia Tesla P100
GPU/Node	1	2
Memory (GB)/Node	192	512
Total Cores	576	1120
Total Memory (GB)	3,456	20,480
Local Disk	480GB SSD	187GB SSD
Interconnect	EDR-100 IB	Ethernet

Login

ARC users can log into Infer at:

`infer1.arc.vt.edu`

Policies

Limits are set on the scale and quantity of jobs at the user and allocation (Slurm account) levels to help ensure availability of resources to a broad set of researchers and applications:

	t4_normal_q	t4_dev_q	p100_normal_q	p100_dev_q
Node Type	T4 GPU	T4 GPU	P100 GPU	P100 GPU
Billing Weight	0 (no billing)	0 (no billing)	0 (no billing)	0 (no billing)
Number of Nodes	16	2	-coming soon-	-coming soon-
MaxRunningJobs (User)	10	2		
MaxSubmitJobs (User)	100	3		
MaxRunningJobs (Allocation)	20	3		
MaxSubmitJobs (Allocation)	200	6		
MaxNodes (User)	8	2		
MaxNodes (Allocation)	12	2		
MaxCPUs (User)	256	64		
MaxCPUs (Allocation)	384	64		
MaxGPUs (User)	8	2		
MaxGPUs (Allocation)	12	2		
Max Job Duration (hours)	72	4		

Modules

Infer's module structure is similar to that of *TinkerCliffs*, but different from previous ARC clusters in that it uses a new application stack/module system based on EasyBuild. A video tutorial of module usage under this paradigm is provided here; a longer class on EasyBuild, including how you can use it to build your own modules is here.

Key differences between EasyBuild and our legacy paradigm from a user perspective include:

- Hierarchies are replaced by toolchains. Right now, there are four:
  - foss ("Free Open Source Software"): gcc compilers, OpenBLAS for linear algebra, OpenMPI for MPI, etc
  - fosscuda: `foss` with CUDA support
  - intel: Intel compilers, Intel MKL for linear algebra, Intel MPI
  - intelcuda: `intel` with CUDA support
- Instead of loading modules individually (e.g., module load intel mkl impi), a user can just load the toolchain (e.g., `module load fosscuda/2020b`).
- Modules load their dependencies, e.g.,

```
$ module reset; module load GROMACS/2020.4-fosscuda-2020b; module list
Currently Loaded Modules:
  1) shared                          8) GCCcore/10.2.0                 15) numactl/2.0.13-
↪GCCcore-10.2.0     22) GDRCopy/2.1-GCCcore-10.2.0-CUDA-11.1.1  29) FFTW/3.3.8-gompic-
↪2020b
  2) gcc/9.2.0                       9) zlib/1.2.11-GCCcore-10.2.0     16) XZ/5.2.5-GCCcore-
↪10.2.0           23) UCX/1.9.0-GCCcore-10.2.0-CUDA-11.1.1    30) ScaLAPACK/2.1.0-
↪gompic-2020b
  3) slurm/slurm/19.05.5            10) binutils/2.35-GCCcore-10.2.0  17) libxml2/2.9.10-
↪GCCcore-10.2.0     24) libfabric/1.11.0-GCCcore-10.2.0         31) fosscuda/2020b
  4) apps                           11) GCC/10.2.0                     18) libpciaccess/0.16-
↪GCCcore-10.2.0  25) PMIx/3.1.5-GCCcore-10.2.0                32) GROMACS/2020.4-
↪fosscuda-2020b
  5) site/infer/easybuild/setup  12) CUDAcore/11.1.1                19) hwloc/2.2.0-
↪GCCcore-10.2.0        26) OpenMPI/4.0.5-gcccuda-2020b
  6) useful_scripts                 13) CUDA/11.1.1-GCC-10.2.0         20) libevent/2.1.12-
↪GCCcore-10.2.0    27) OpenBLAS/0.3.12-GCC-10.2.0
  7) DefaultModules                 14) gcccuda/2020b                  21) Check/0.15.2-
↪GCCcore-10.2.0      28) gompic/2020b
```

- All modules are visible with `module avail`. So in many cases it is probably better to search with `module spider` rather than printing the whole list.
- Some key system software, like the Slurm scheduler, are included in default modules. This means that `module purge` can break important functionality. Use `module reset` instead.
- Lower-level software is included in the module structure (see, e.g., `binutils` in the GROMACS example above), which should mean less risk of conflicts in adding new versions later.
- Environment variables (e.g., `$SOFTWARE_LIB`) available in our previous module system may not be provided. Instead, EasyBuild typically provides `$EBROOTSOFTWARE` to point to the software installation location. So for example, to link to NetCDF libraries, one might use `-L$EBROOTCUDA/lib64` instead of the previous `-L$CUDA_LIB`.

3.1.3 Cascades, CPU/GPU Cluster

Overview

Cascades is a 236-node system capable of tackling the full spectrum of computational workloads, from problems requiring hundreds of compute cores to data-intensive problems requiring large amount of memory and storage resources. Cascade contains four compute engines designed for distinct workloads.

- **General** - Distributed, scalable workloads. With Intel's Broadwell processors, 2 16-core processors and 128 GB of memory on each node, this 190-node compute engine is suitable for traditional HPC jobs and large codes using MPI.
- **Very Large Memory** - Graph analytics and very large datasets. With 3TB (3072 gigabytes) of memory, four 18-core processors and 6 1.8TB direct attached SAS hard drives, 400 GB SAS SSD drive, and one 2 TB NVMe PCIe flash card , each of these two servers will enable analysis of large highly-connected datasets, in-memory database applications, and speedier solution of other large problems.
- **K80 GPU** - Data visualization and code acceleration. There are four nodes in this compute engine which have - two Nvidia K80 (Kepler) GPUs, 512 GB of memory, and one 2 TB NVMe PCIe flash card.
- **V100 GPU** - Extremely fast execution of GPU-enabled codes. There are 40 nodes in this engine, although one of these nodes is reserved for system maintenance. Each node is equipped with two Intel Skylake Xeon Gold 3

Ghz CPU's, amounting to 24 cores on each node. There is 384 GB of memory, and two NVIDIA V100 (Volta) GPU's. Each of these GPU's is capable of more than 7.8 TeraFLOPS of double precision performance.

Technical Specifications

COMPUTE ENGINE	#	HOSTS	CPU	CORES	MEMORY	LOCAL STORAGE	OTHER FEATURES
General	190	ca007-ca196	2 x E5-2683v4 2.1GHz (Broadwell)	32	128 GB, 2400 MHz	1.8TB 10K RPM SAS200 GB SSD	
Very Large Memory	2	ca001-ca002	4 x E7-8867v4 2.4 GHz (Broadwell)	72	3 TB, 2400 MHz	3.6 TB (2 x 1.8 TB) 10K RPM SAS (RAID 0)6-400 GB SSD (RAID 1) 2 TB NVMe PCIe	
K80 GPU	4	ca003-ca006	2 x E5-2683v4 2.1GHz (Broadwell)	32	512GB, 2400MHz	3.6 TB (2 x 1.8 TB) 10K RPM SAS (RAID 0)2-400 GB SSD (RAID 1) 2 TB NVMe PCIe	2-NVIDIA K80 GPU
V100 GPU	40	ca197-ca236	2 x Intel Xeon Gold 6136 3.0GHz (Skylake)	24	384GB, 2666MHz	2-400 GB SSD (RAID 1)	2-NVIDIA V100 GPU

Notes:

- K80 GPU Notes: There are 4 CUDA Devices. Although the K80s are a single physical device in 1 PCIe slot, there are 2 separate GPU chips inside. They will be shown as 4 separate devices to CUDA code. nvidia-smi will show this.
- All nodes have locally mounted SAS and SSDs. `/scratch-local` (and `$TMPDIR`) point to the SAS drive and `/scratch-ssd` points to the SSD on each node. On large memory and GPU nodes, which have multiple of each drive, the storage across the SSDs are combined in `/scratch-ssd` (RAID 0) and the SAS drives are mirrored (RAID 1) for redundancy.

Network:

- 100 Gbps Infiniband interconnect provides low latency communication between compute nodes for MPI traffic.
- 10 Gbps Ethernet interconnect provides high speed connectivity and access to storage.

Policies

Cascades is governed by an allocation manager, meaning that in order to run most jobs, you must be an authorized user of an allocation that has been submitted and approved. For more on allocations, click *here*. The Cascades partitions (queues) are:

- **normal_q** for production (research) runs.
- **largemem_q** for production (research) runs on the large memory nodes.
- **dev_q** for short testing, debugging, and interactive sessions. dev_q provides slightly elevated job priority to facilitate code development and job testing prior to production runs.
- **k80_q** for runs that require access to K80 GPU nodes

- **v100_normal_q** for production (research) runs with the V100 nodes
- **v100_dev_q** short testing, debugging, and interactive sessions with the V100 nodes

The Cascades partition (queue) settings are:

PARTITION	NORMAL_Q	LARGE-MEM_Q	DEV_Q	K80_Q	V100_NORMAL	V100_DEV
Access to	ca007-ca196	ca001-ca002	ca007-ca196	ca003-ca006	ca197-ca236	ca197-ca236
Max Jobs	24 per user, 48 per allocation	1 per user	1 per user	4 per user, 6 per allocation	8 per user, 12 per allocation	1 per user
Max Nodes	32 per user, 48 per allocation	1 per user	32 per user, 48 per allocation	4 per user	12 per user, 24 per allocation	12 per user, 24 per allocation
Max Cores	1,024 per user, 1,536 per allocation	72 per user	1,024 per user, 1536 per allocation	128 per user	288 per user, 576 per allocation	336 per user
Max Memory (calculated, not enforced)	4 TB per user, 6 TB per allocation	3 TB per user	4 TB per user, 6 TB per allocation	2 TB per user	4 TB per user, 6 TB per allocation	1 TB per user
Max Walltime	144 hr	144 hr	2 hr	144 hr	144 hr	2 hr
Max Core-Hours	73,728 per user	10,368 per user	256 per user	9,216 per user	20,736 per user	168 per user

Notes:

- *Shared* node access: more than one job can run on a node
- The micro-architecture on the V100 nodes is newer than (and distinct from) the Broadwell nodes. For best performance and compatibility, programs that are to run on V100 nodes should be compiled on a V100 node. Note that the login nodes are Broadwell nodes, so compilation on a V100 node should be done as part of the batch job, or during an interactive job on a V100 node (see below).

Access

Cascades can be accessed via one of the two login nodes:

- `cascades1.arc.vt.edu`
- `cascades2.arc.vt.edu`

Users may also use *Open OnDemand* to access the cluster.

Job Submission

Access to all compute nodes is controlled via the Slurm resource manager; see the *Slurm documentation* for additional usage information. Example resource requests on Cascades include:

```
#Request exclusive access to all resources on 2 nodes
#SBATCH --nodes=2
#SBATCH --exclusive

#Request 4 cores (on any number of nodes)
#SBATCH --ntasks=4

#Request 2 nodes with 12 tasks running on each
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=12

#Request 12 tasks with 20GB memory per core
#SBATCH --ntasks=12
#SBATCH --mem-per-cpu=20G

#Request one NVIDIA V100 GPU and 100GB memory
#SBATCH --nodes=1 #(implies --ntasks=1 unless otherwise specified)
#SBATCH --partition=v100_normal_q
#SBATCH --gres=gpu:1
#SBATCH --mem=100G
```

3.1.4 DragonsTooth, High-Throughput Computing

Overview

DragonsTooth is a 48-node system designed to support general batch HPC jobs. The table below lists the technical details of each DragonsTooth node. Nodes are connected to each other and to storage via 10 gigabit ethernet (10GbE), a communication channel with high bandwidth but higher latency than InfiniBand (IB). As a result, DragonsTooth is better suited to jobs that require less internode communication and/or less I/O intearction with non-local storage than NewRiver, which has similar nodes but a low-latency IB interconnect. To allow I/O-intensive jobs, DragonsTooth nodes are each outfitted with nearly 2 TB of solid state local disk. DragonsTooth was released to the Virginia Tech research community in August 2016. In November of 2018, DragonsTooth was reprovisioned with Slurm as its scheduler as a replacement for Moab/Torque.

Technical Specifications

Component	Specification
CPU	2 x Intel Xeon E5-2680v3 (Haswell) 2.5 GHz 12-core
Memory	256 GB 2133 MHz DDR4
Local Storage	4 x 480 GB SSD Drives
Theoretical Peak (DP)	806 GFlops/s

Policies

Note: DragonsTooth is governed by an allocation manager, meaning that in order to run most jobs on it, you must be an authorized user of an allocation that has been submitted and approved. For more on allocations, click *here*.

As described above, communications between nodes and between a node and storage will have higher latency on DragonsTooth than on other ARC clusters. For this reason the queue structure is designed to allow more jobs and longer-running jobs than on other ARC clusters. DragonsTooth has two partitions (queues):

- `normal_q` for production (research) runs.
- `dev_q` for short testing, debugging, and interactive sessions. `dev_q` provides slightly elevated job priority to facilitate code development and job testing prior to production runs.

The settings for the partitions are:

Partition	`normal_q`	`dev_q`
Access to	dt003-dt048	dt003-dt048
Max Jobs	288 per user 432 per allocation	1 per user
Max Nodes	12 per user 18 per allocation	12 per user
Max Core-Hours*	34,560 per user 51,840 per allocation	96 per user
Max Walltime	30 days	2 hr

Other notes:

- *Shared* node access: more than one job can run on a node.

*A user cannot, at any one time, have more than this many core-hours allocated across all of their running jobs. So you can run long jobs or large/many jobs, but not both. For illustration, the following table describes how many nodes a user can allocate for a given amount of time:

Walltime	Max Nodes (per user)	Max Nodes (per allocation)
72 hr (3 days)	12	18
144 hr (6 days)	10	15
360 hr (15 days)	4	6
720 hr (30 days)	2	3

Access

DragonsTooth can be accessed via one of the two login nodes:

- `dragonstooth1.arc.vt.edu`
- `dragonstooth2.arc.vt.edu`

Users may also use *Open OnDemand* to access the cluster.

Job Submission

Access to all compute nodes is controlled via the Slurm resource manager; see the *Slurm documentation* for additional usage information. Example resource requests on Cascades include:

```
#Request exclusive access to all resources on 2 nodes
#SBATCH --nodes=2
#SBATCH --exclusive

#Request 4 cores (on any number of nodes)
#SBATCH --ntasks=4

#Request 2 nodes with 12 tasks running on each
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=12

#Request 12 tasks with 20GB memory per core
#SBATCH --ntasks=12
#SBATCH --mem-per-cpu=20G
```

3.1.5 Huckleberry

Warning: Huckleberry is scheduled to be retired in Spring 2022. Please consider one of our other *GPU resources* for deep learning applications.

Overview

Huckleberry is a high performance computing system targeted at deep learning applications. Huckleberry consists of two login nodes and Fourteen IBM Minksy S822LC compute nodes. Each of the compute nodes is equipped with:

- Two IBM Power8 CPU (3.26 GHz) with 256 GB of memory
- Four NVIDIA P100 GPU with 16 GB of memory each
- NVLink interfaces connecting CPU and GPU memory spaces
- Mellanox EDR Infiniband (100 GB/s) interconnect
- CentOS 7 OS

Login

To access Huckleberry, users should login to: `ssh huckleberry1.arc.vt.edu`

Basic Job Submission and Monitoring

The huckleberry `normal_q` imposes the following limits

- maximum walltime of 3 days
- maximum of three nodes per user The huckleberry `large_q` imposes the following limits
- maximum walltime of 1 day
- maximum of four nodes per user The current configuration allows users to run jobs either through the batch scheduler or interactively. The following is a basic hello world job submission script requesting 500 GB memory and all four Pascal P100 GPU on a compute node:

```
#!/bin/bash

#SBATCH -J hello-world
#SBATCH -p normal_q
#SBATCH -p normal_q
#SBATCH -N 1  # this will not assign the node exclusively. See the note above for details
#SBATCH -t 10:00
#SBATCH --mem=500G
#SBATCH --gres=gpu:4
#SBATCH --account=(YOUR ALLOCATION ID)
echo hello world
```

NOTE: asking for `-N 1` without specifying how many cores per node will default to only 1 core (equivalent to `-n 1`). If you would like to get the full node exclusively, you should ask for all the cores on the node using the flag -n, or, you could use the `--exclusive` flag.

To learn how to submit or monitor your jobs, please see the *Slurm documentation*.

In many cases jobs will require fewer than the four GPU available on each huckleberry compute node. GPU can be requested as a generic resource (GRES) through Slurm by requesting a specific number of processor cores and GPU. To request one processor core and one GPU in an *interactive session* with 8 GB of memory per processor core,

```
interact --nodes=1 --ntasks-per-node=8  -l walltime=0:10:00 --mem-per-cpu=8G --
↪gres=gpu:1 -A yourallocation
```

Slurm will set the `$CUDA_VISIBLE_DEVICES` environment variable automatically based on your request. Multiple processor cores and/or GPU can be requested in the same manner. For example, to request two GPU and 10 CPU cores, one might run

```
interact -n10 -t 10:00 --mem-per-cpu=4G --gres=gpu:2
```

The Power8 CPU are viewed by Slurm as 20 processor cores.

Software

Software modules are available on huckleberry and function in the same manner as other ARC systems, e.g. the following syntax will load the module for cuda `module load cuda`. Additionally, IBM's PowerAI deep learning software are installed under within the Anaconda3 module. A few brief tutorials are provided below.

Python

For users that would like to customize their Python environment, we *provide online documentation for best practices to manage Python on ARC systems*. For more detailed usages, please refer to part below.

Jupyter Notebooks

Jupyter notebooks are included in the anaconda python distribution installed on huckleberry. An example script to launch a job on a compute node is here:

```
#!/bin/bash

#SBATCH -J start-jupyter
#SBATCH -n 4
##SBATCH --exclusive
#SBATCH --gres=gpu:pascal:1
#SBATCH --mem=120G
#SBATCH -t 24:00:00
#SBATCH -p normal_q

echo "starting jupyter notebook"

#PATH=/home/mcclurej/anaconda2/bin:$PATH
export PATH=/opt/apps/anaconda2/4.4.0.1/bin:$PATH

module load cuda
source /opt/DL/caffe-ibm/bin/caffe-activate
source /opt/DL/openblas/bin/openblas-activate
source /opt/DL/tensorflow/bin/tensorflow-activate
source /opt/DL/theano/bin/theano-activate
source /opt/DL/torch/bin/torch-activate
source /opt/DL/digits/bin/digits-activate

#let ipnport=($UID-6025)%65274
#echo $ipnport >> ipnport.txt

#jupyter notebook --ip=$HOSTNAME --port=5034 --no-browser > jupyter.server
unset XDG_RUNTIME_DIR

GPUID=$(echo $CUDA_VISIBLE_DEVICES | cut -c1)
port=`expr 5030 + $GPUID`

jupyter notebook --ip=$HOSTNAME --port=$port --no-browser &>  jupyter.hostname

exit
```

This will start a jupyter notebook with an appropriate hostname and port so that the session can be opened in a browser on the login node. When using firefox, it is recommended to use X-forwarding and compression when connecting to huckleberry as follows

```
ssh -X -C huckleberry1.arc.vt.edu
```

Download the juypter-server script to your home directory with Then if the script above is in the file `jupyter-server.sh`, you can start the notebook by submitting a batch job with

```
sbatch jupyter-server.sh &
```

The script will populate the file `jupyter.hostname` with the appropriate URL to interact with the remote session. This URL can be extracted from the file as follows

```
URL=$(grep -A2 URL jupyter.hostname | tail -1)
```

Then open a firefox window from the login node

```
firefox --no-remote -url $URL &
```

The jupyter notebook should open in the firefox browser, running on the compute node assigned to you job.

PowerAI

Many of the PowerAI tools depend on cuda, and your `$PATH` and `$LD_LIBRARY_PATH` variables should be set accordingly:

```
export PATH=/usr/local/cuda-8.0/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-8.0/lib64:$LD_LIBRARY_PATH
```

Theano depends on pycuda, which is not included in the centrally-provided python. It can be installed locally as follows (see our *python user guide* for additional details):

```
pip install --user pycuda
```

DIGITS wraps several of the popular deep learning tools into an easy-to-use web interface. To open the DIGITS interface, first establish an instance of the DIGITS server by submitting a batch job that launches `digits-devserver` on one of the compute nodes. The following script will start the digits server on a compute node with 2 hours of walltime:

```
#!/bin/bash

#SBATCH -J digits-devserver
#SBATCH -N 1
#SBATCH -t 24:00:00

echo "starting digits server"

module load cuda
source /opt/DL/caffe-ibm/bin/caffe-activate
source /opt/DL/openblas/bin/openblas-activate
source /opt/DL/tensorflow/bin/tensorflow-activate
source /opt/DL/theano/bin/theano-activate
source /opt/DL/torch/bin/torch-activate
source /opt/DL/digits/bin/digits-activate

digits-devserver

exit
```

The job should be launched by typing

```
sbatch digits-devserver.sh
```

Type `squeue` to identify which compute node the job is running on. Once the server is running on the compute node, you will be able to load DIGITS from a browser that runs on the login node. To start firefox from the login node, type

```
firefox --no-remote &
```

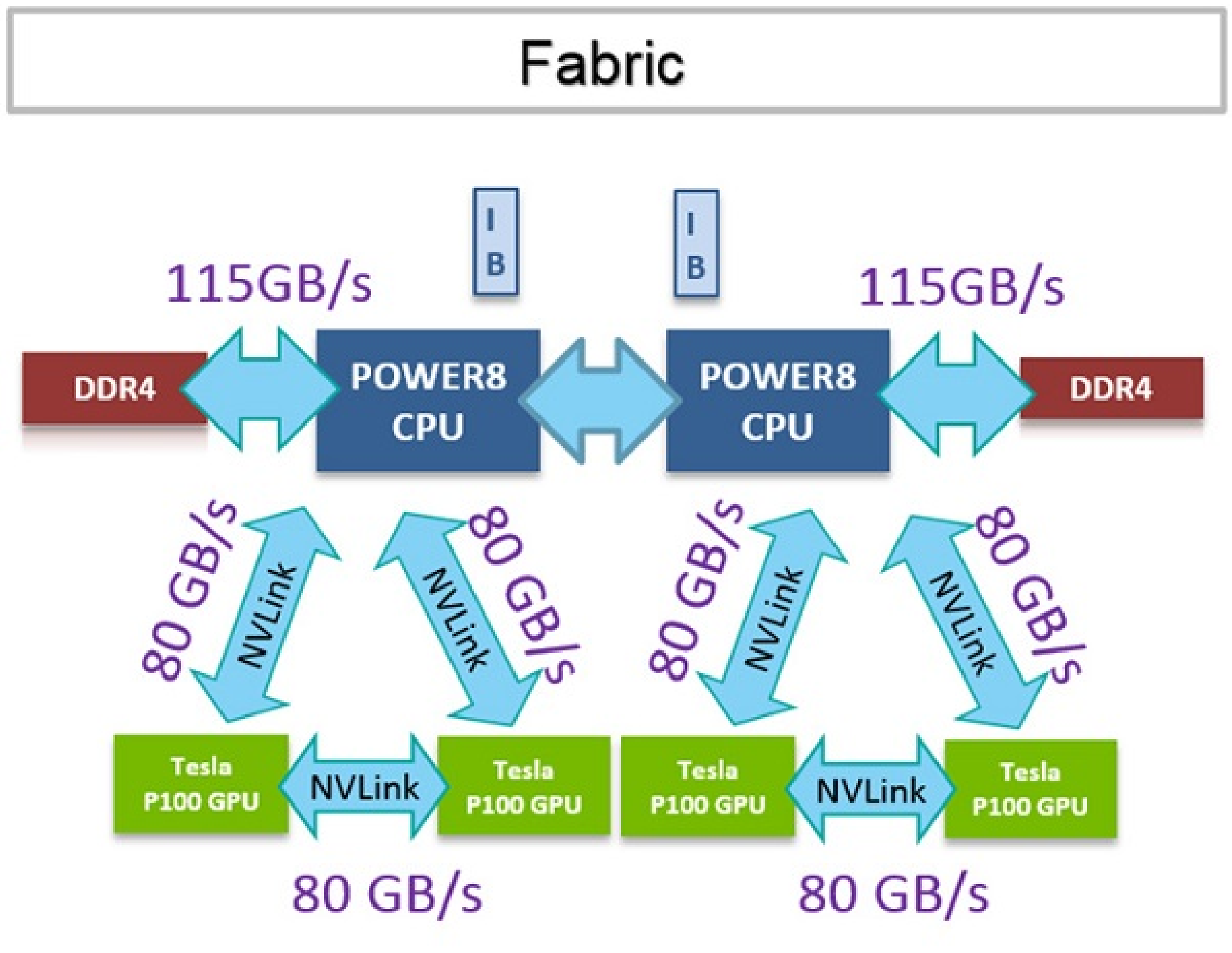
If your job is running on compute node `hu001`, you should point your browser at `http://hu001:5000` to open the digits interface (if your job is running on another compute node, you should enter it instead of `hu001`). DIGITS essentially provides a portal to control the jobs that run on the compute node. To train a basic model, a good starting point are the basic examples included in DIGITS. Input data has already been downloaded to the ARC filesystem. A local copy can be obtained by running

```
tar xvzf /home/TRAINING/mnist.tar.gz
```

Once the data has been downloaded, you can train a model by following the steps described at https://github.com/NVIDIA/DIGITS/blob/master/docs/GettingStarted.md.

NUMA

Understanding non-uniform memory access (NUMA) patterns important to get the full benefit of the S822LC compute nodes on huckleberry. The memory bandwidth associated with data movement within each compute node is summarized in the diagram below. Note that each Power8 CPU is coupled to two P100 GPU through NVLink, which supports bi-directional data transfer rates of 80 GB/s. The theoretical maximum memory bandwidth for each Power8 CPU is 115 GB/s. The theoretical maximum memory bandwidth for each NVIDIA P100 GPU is 720 GB/s.



PowerAI Installation & Usage (Updated in April 2019)

All testing(on TF, Pytorch, Keras(TF backend), Caffe) has been performed with python/3.6 on Huckleberry GPU nodes, you could see testing demonstrations and example python scripts from this shared Google Drive Folder

Part 1. PowerAI Library Usage (PREFERRED)

```
# step 1: request for GPU nodes
# salloc --partition=normal_q --nodes=1 --tasks-per-node=10 --gres=gpu:1 bash
# step 2: load all necessary modules
module load gcc cuda Anaconda3 jdk
# step 3: activate the virtual environment
source activate powerai16_ibm
# step 4: test with simple code examples, Google drive above
python test_pytorch.py
python test_TF_multiGPUs.py
python test_keras.py
# step 5: for new packages(take beautifulsoup4 for example)
```

(continues on next page)

(continued from previous page)

```
pip install --user beautifulsoup4 # on hulogin1/hulogin2
# or pip install --user --no-deps keras
```

Part 2. Installation

First make sure you are in hulogin1/hulogin2

```
module load gcc cuda Anaconda3 jdk
java -version
conda create -n powerai36 python==3.6 # create a virtual environment
source activate powerai36 # activate virtual environment
conda config --prepend channels https://public.dhe.ibm.com/ibmdl/export/pub/software/
↪server/ibm-ai/conda/
# if things don't work, add two channels and run commands showing below
conda config --add default_channels https://repo.anaconda.com/pkgs/main
conda config --add default_channels https://repo.anaconda.com/pkgs/r
# install ibm powerai meta-package via conda
conda install powerai
# keep type 'enter' and then enter 1 for license acceptance
export IBM_POWERAI_LICENSE_ACCEPT=yes
# you will need to update the jupyter package
conda install jupyter notebook
```

Please feel free to contact us if you have seen issues or have special requirements over using ML/DL/Simu/Vis packages on Huckleberry.

3.2 List of GPUs on ARC Resources

Need a GPU? Here is a list of where you can find them on ARC's clusters:

Architecture	Cluster	Partition	Number
NVIDIA A100-80G	*TinkerCliffs*	`a100_normal_q`, `a100_dev_q`	32 (4 nodes, 8 GPU/node)
NVIDIA Volta V100	*Infer*	`v100_normal_q`, `v100_dev_q`	4 (2 nodes, 2 GPU/node)
NVIDIA Volta V100	*Cascades**	`v100_normal_q`, `v100_dev_q`	76 (38 nodes, 2 GPU/node)
NVIDIA Tesla T4	*Infer*	`t4_normal_q`, `t4_dev_q`	18 (18 nodes, 1 GPU/node)
NVIDIA Tesla P100	*Infer*	`p100_normal_q`, `p100_dev_q`	80 (40 nodes, 2 GPU/node)
NVIDIA Tesla P100	*Huckleberry*	`normal_q`	56 (14 nodes, 4 GPU/node)
NVIDIA Tesla K80	*Cascades**	`k80_q`	16 (4 nodes, 4 GPU/node)

* ARC is preparing to move these nodes to *Infer*.

3.3 Open OnDemand

Open OnDemand is a web portal that provides access to ARC HPC clusters. It facilitates clusters' access and job management without the need for Linux experience or any installations on the client-side. The only requirement is an up-to-date web browser. Firefox or Chrome are preferred.

3.3.1 Features

OnDemand provides the following features

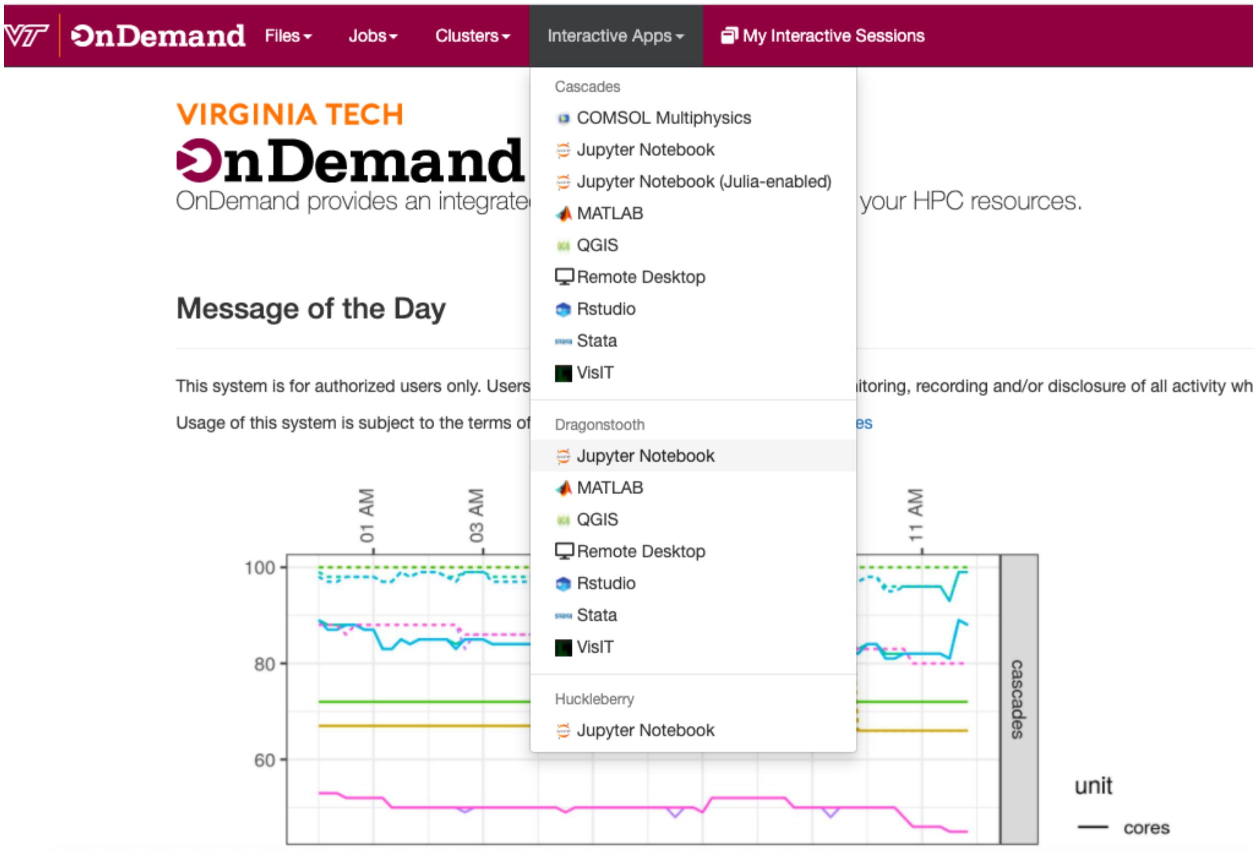
- File Management and Transfer
- Job Management
- Shell Access
- Interactive Apps

3.3.2 Usage instructions

- In order to use OnDemand, you will need to be using the university network or on VPN (VT Traffic over SSL VPN)
- Once connected, go to either:
  - https://ondemand.arc.vt.edu (Legacy site: Older version)
  - https://ood.arc.vt.edu (New site: Newer version and features, but still under development in places)
- Then, you can log in using your VT credentials (PID and password). If already logged into another VT site, you may not need to enter any credentials at all.

3.3.3 Examples

OnDemand provides interactive apps on each of the clusters, as shown in the image below.



See also our video tutorial.

For complete documentation, please visit Ohio Supercomputer Center, which develops Open OnDemand detailed documentation pages.

3.4 Storage Resources

3.4.1 Overview

ARC offers several different storage options for users' data:

Name	Intent	File System	Environment Variable	Per User Maximum	Data Lifespan	Available On
Home	Long-term storage of files	Qumulo	$HOME	640 GB 1 million files	Unlimited	Login and Compute Nodes
Group (Cascades, DragonsTooth, Huckleberry)	Long-term storage of shared, group files	GPFS	- n/a -	10 TB, 5 million files per faculty researcher (Expandable via investment)	Unlimited	Login and Compute Nodes
Project (TinkerCliffs, Infer)	Long-term storage of shared, group files	BeeGFS	- n/a -	25 TB, 5 million files per faculty researcher (Expandable via investment)	Unlimited	Login and Compute Nodes
Work (Cascades, DragonsTooth, Huckleberry)	Fast I/O, Temporary storage	GPFS	$WORK	14 TB, 3 million files	120 days	Login and Compute Nodes
Work (TinkerCliffs, Infer)	Fast I/O, Temporary storage	BeeGFS	$WORK	1 TB, 1 million files	Unlimited	Login and Compute Nodes
Archive	Long-term storage for infrequently-accessed files	GPFS	$ARCHIVE	-	Unlimited	Login Nodes
Local Scratch	Local disk (hard drives)		$TMPDIR	Size of node hard drive	Length of Job	Compute Nodes
Memory (tmpfs)	Very fast I/O	Memory (RAM)	$TMPFS	Size of node memory allocated to job	Length of Job	Compute Nodes

Each is described in the sections that follow.

3.4.2 Home

Home provides long-term storage for system-specific data or files, such as installed programs or compiled executables. Home can be reached the variable `$HOME`, so if a user wishes to navigate to their Home directory, they can simply type `cd $HOME`. Each user is provided a maximum of 640 GB in their Home directories (across all systems). When a user exceeds the soft limit, they are given a grace period after which they can no longer add any files to their Home directory until they are below the soft limit. Home directories are also subject to a 690 GB hard limit; users Home directories are not allowed to exceed this limit. Note that running jobs fail if they try to write to a Home directory after the soft limit grace period is expired or when the hard limit is reached.

3.4.3 Group and Project

Project (on *TinkerCliffs* and *Infer*) and Group (on *Cascades*, *DragonsTooth*, and *Huckleberry*) provide long-term storage for files shared among a research project or group, facilitating collaboration and data exchange within the group. Each Virginia Tech faculty member can request group storage up to the prescribed limit at no cost by requesting a storage allocation via ColdFront. Additional storage may be purchased through the investment computing or cost center programs.

Quotas on Project

The file system that provides Project and Work directories on *TinkerCliffs* and *Infer* does quotas based on the *group ID* (GID) associated with files. This means that:

- Files in your Work directory can count against your Project quota if they have that project's GID
- Files in your Project directory can count against your Work quota if they have your personal GID

You can check your Project and Work quotas with the `quota` *command*. You can check the GID associated with your files with `ll` (the same as `ls -l`) and can change the group with `chgrp` (`chgrp -R` for recursive on a directory). You can find files in a more automated fashion with `find` – see the example below. As an example, here we find some files in `/projects/myproject` that are owned by `mypid`:

```
[mypid@tinkercliffs2 ~]$ find /projects/myproject/test -group mypid
/projects/myproject/test
/projects/myproject/test/datafile
/projects/myproject/test/test.txt
[mypid@tinkercliffs2 ~]$ ls -ld /projects/myproject/test/
drwxrwxr-x 2 mypid mypid 2 Oct  4 08:43 /projects/myproject/test/
[mypid@tinkercliffs2 ~]$ ls -lh /projects/myproject/test/
total 1.1G
-rw-rw-r-- 1 mypid mypid 1.0G Oct  4 08:43 datafile
-rw-rw-r-- 1 mypid mypid    5 Jun  8 10:51 test.txt
```

These files will count against `mypid`'s Work quota. We change their ownership to the associated group with `chgrp -R`:

```
[mypid@tinkercliffs2 ~]$ chgrp -R arc.myproject /projects/myproject/test
[mypid@tinkercliffs2 ~]$ ls -ld /projects/myproject/test/
drwxrwxr-x 2 mypid arc.myproject 2 Oct  4 08:43 /projects/myproject/test/
[mypid@tinkercliffs2 ~]$ ls -lh /projects/myproject/test/
total 1.1G
-rw-rw-r-- 1 mypid arc.myproject 1.0G Oct  4 08:43 datafile
-rw-rw-r-- 1 mypid arc.myproject    5 Jun  8 10:51 test.txt
```

The files will now count against the Project quota.

A more automated example would be to have `find` both locate *and change ownership* of the files:

```
[mypid@tinkercliffs2 ~]$ ls -lh /projects/myproject/test/
total 1.1G
-rw-rw-r-- 1 mypid mypid 1.0G Oct  4 08:43 datafile
-rw-rw-r-- 1 mypid mypid    5 Jun  8 10:51 test.txt
[mypid@tinkercliffs2 ~]$ find /projects/myproject/test -group mypid -exec chgrp arc.
↪myproject {} +
[mypid@tinkercliffs2 ~]$ ls -lh /projects/myproject/test/
```

(continues on next page)

(continued from previous page)

```
total 1.1G
-rw-rw-r-- 1 mypid arc.myproject 1.0G Oct  4 08:43 datafile
-rw-rw-r-- 1 mypid arc.myproject    5 Jun  8 10:51 test.txt
```

3.4.4 Work

Work provides users with fast, user-focused storage for use during simulations or other research computing applications. However, it encompasses two paradigms depending on the cluster where it is being used:

- On *TinkerCliffs* and *Infer*, it provides 1 TB of user-focused storage that is not subject to a time limit. Note that this quota is enforced by the GID associated with files and not by directory, so files in Project storage can wind up being counted against your Work quota; see *here* for details and fixes.
- On *Cascades*, *DragonsTooth*, and *Huckleberry*, it provides up to 14 TB of space. However, ARC reserves the right to purge files older than 120 days from this file system. It is therefore aimed at temporary files, checkpoint files, and other scratch files that might be created during a run but are not needed long-term. Work for a given system can be reached via the variable `$WORK`. So if a user wishes to navigate to Work directory, they can simply type cd `$WORK`.

3.4.5 Archive

Archive provides users with long-term storage for data that does not need to be frequently accessed i.e. storing important/historical results. Archive is accessible from all ARC's systems. Archive is not mounted on compute nodes, so running jobs cannot access files on it. Archive can be reached the variable `$ARCHIVE`, so if a user wishes to navigate to their Archive directory, they can simply type `cd $ARCHIVE`.

Best Practices for archival storage

Because the ARCHIVE filesystem is backed by tape (a high capacity but very high latency medium), it is very inefficient and disruptive to do file operations (especially on lots of small files) on the archive filesystem itself. Archival systems are designed to move and replicate very large files; ideally users will tar all related files into singular, large files. Procedures are below:

To place data in `$ARCHIVE`:

1. create a tarball containing the files in your `$HOME` (or `$WORK`) directory
2. copy the tarball to the `$ARCHIVE` filesystem (use rsync in case the transfer were to fail)

To retrieve data from `$ARCHIVE`:

1. copy the tarball back to your `$HOME` (or `$WORK`) directory (use rsync in case the transfer were to fail).
2. untar the file on the login node in your `$HOME` (or `$WORK`) directory. Directories can be tarred up in parallel with, for example, gnu parallel (available via the `parallel` module). This line will create a tarball for each directory more than 180 days old:

```
find . -maxdepth 1 -type d -mtime +180 | parallel [[ -e {}.tar.gz ]] || tar -czf {}.tar.
↪gz {}
```

The resulting tarballs can then be moved to Archive and directories can then be removed. (Directories can also be removed automatically by providing the `--remove-files` flag to `tar`, but this flag should of course be used with caution.)

3.4.6 Local Scratch

Running jobs are given a workspace on the local hard drive on each compute node. The path to this space is specified in the `$TMPDIR` environment variable. This provides another option for users who would prefer to do I/O to local disk (such as for some kinds of big data tasks). Please note that any files in local scratch are removed at the end of a job, so any results or files to be kept after the job ends must be copied to Work or Home.

3.4.7 Memory

Running jobs have access to an in-memory mount on compute nodes via the `$TMPFS` environment variable. This should provide very fast read/write speeds for jobs doing I/O to files that fit in memory (see the system documentation for the amount of memory per node on each system). Please note that these files are removed at the end of a job, so any results or files to be kept after the job ends must be copied to Work or Home.

3.4.8 Checking Usage

You can check your current storage usage (in addition to your *compute allocation* usage) with the `quota` command:

```
[mypid@tinkercliffs2 ~]$ quota
USER       FILESYS/SET                         DATA (GiB)   QUOTA (GiB) FILES      QUOTA␣
↪     NOTE
mypid      /home                               584.2        596         -          -

           BEEGFS
mypid      /projects/myproject1                109.3        931
mypid      /projects/myproject2                2648.4       25600
mypid      /work/mypid                         2.7          931
```

CHAPTER

FOUR

SOFTWARE

The following pages describe the software packages installed on ARC's systems and how to use them. To access a given software install, please use the *module system*. Your are also welcome to install your own software; see *here* for details.

Contents:

4.1 Examples

ARC maintains a git repository of example submission scripts here.

To, for example, run the `stream` example on TinkerCliffs using their `personal` *allocation*, a user might log into TinkerCliffs and issue the following commands:

```
#clone the repository
git clone git@github.com:AdvancedResearchComputing/examples.git

#change to the stream directory
cd examples/stream

#submit the job (using your personal allocation)
sbatch -Apersonal stream_tinkercliffs_rome.sh
```

The output would then be in the file `slurm-XXXXXX.out` where `XXXXXX` represents the job number.

4.2 Table of Software on ARC Systems

SOFTWARE	DESCRIPTION
guppyGPU	SOFTWAREDESCRIPTION
julia	Julia technical computing language
matlab	MATLAB Technical Computing
ABAQUS	Finite Element Analysis software for modeling, visualization and best-in-class implicit and explicit dynamics
ABINIT	ABINIT is a package whose main program allows one to find the total energy, charge density and electronic
ABySS	Assembly By Short Sequences - a de novo, parallel, paired-end sequence assembler
ANSYS	ANSYS simulation software enables organizations to confidently predict how their products will operate in t
APR	Apache Portable Runtime (APR) libraries.
APR-util	Apache Portable Runtime (APR) util libraries.
ATK	ATK provides the set of accessibility interfaces that are implemented by other toolkits and applications. Usi
AUGUSTUS	AUGUSTUS is a program that predicts genes in eukaryotic genomic sequences

SOFTWARE	DESCRIPTION
AccelerateCFD_CE	Community Edition of AccelerateCFD platform for creating reduced order models from high fidelity CFD
Anaconda3	Anaconda python distribution, python version 3.5
AtomPAW	AtomPAW is a Projector-Augmented Wave Dataset Generator that can be used both as a standalone program
Autoconf	Autoconf is an extensible package of M4 macros that produce shell scripts to automatically configure softwa
Automake	Automake: GNU Standards-compliant Makefile generator
Autotools	This bundle collect the standard GNU build tools: Autoconf, Automake and libtool
BCFtools	Samtools is a suite of programs for interacting with high-throughput sequencing data. BCFtools - Reading/w
BEDTools	BEDTools: a powerful toolset for genome arithmetic.The BEDTools utilities allow one to address common g
BLAST+	Basic Local Alignment Search Tool, or BLAST, is an algorithm for comparing primary biological sequence
BUSCO	BUSCO: assessing genome assembly and annotation completeness with single-copy orthologs
BamTools	BamTools provides both a programmer's API and an end-user's toolkit for handling BAM files.
Bazel	Bazel is a build tool that builds code quickly and reliably.It is used to build the majority of Google's software
Biopython	Biopython is a set of freely available tools for biological computation written in Python by an international te
Bison	Bison is a general-purpose parser generator that converts an annotated context-free grammar into a determin
Boost	Boost provides free peer-reviewed portable C++ source libraries.
Bowtie2	Bowtie 2 is an ultrafast and memory-efficient tool for aligning sequencing reads to long reference sequences.
CGAL	The Computational Geometry Algorithms Library is a C++ library that aims to provide easy access to efficie
CMake	CMake, the cross-platform, open-source build system. CMake is a family of tools designed to build, test and
CP2K	CP2K is a freely available (GPL) program, written in Fortran 95, to perform atomistic and molecular simulat
DB	Berkeley DB enables the development of custom data management solutions, without the overhead traditiona
DBus	D-Bus is a message bus system, a simple way for applications to talk to one another. In addition to interproc
Dalton	The Dalton suite consists of two separate executables, Dalton and LSDalton. The Dalton code is a powerful t
DendroPy	A Python library for phylogenetics and phylogenetic computing: reading, writing, simulation, processing and
Doxygen	Doxygen is a documentation system for C++, C, Java, Objective-C, Python, IDL (Corba and Microsoft flavor
ELPA	Eigenvalue SoLvers for Petaflop-Applications .
EasyBuild	EasyBuild is a software build and installation framework written in Python that allows you to install software
Eigen	Eigen is a C++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithm
FDS	Fire Dynamics Simulator (FDS) is a large-eddy simulation (LES) code for low-speed flows, with an emphasi
FFTW	FFTW is a C subroutine library for computing the discrete Fourier transform (DFT
FFmpeg	A complete, cross-platform solution to record, convert and stream audio and video.
FLAC	FLAC stands for Free Lossless Audio Codec, an audio format similar to MP3, but lossless, meaningthat audi
FriBidi	The Free Implementation of the Unicode Bidirectional Algorithm.
GCC	The GNU Compiler Collection includes front ends for C, C++, Objective-C, Fortran, Java, and Ada, as well a
GCCcore	The GNU Compiler Collection includes front ends for C, C++, Objective-C, Fortran, Java, and Ada, as well a
GDAL	GDAL is a translator library for raster geospatial data formats that is released under an X/MIT style Open So
GEOS	GEOS (Geometry Engine - Open Source) is a C++ port of the Java Topology Suite (JTS)
GLPK	The GLPK (GNU Linear Programming Kit) package is intended for solving large-scale linear programming
GLib	GLib is one of the base libraries of the GTK+ project
GMP	GMP is a free library for arbitrary precision arithmetic, operating on signed integers, rational numbers, and f
GMT	GMT is an open source collection of about 80 command-line tools for manipulating geographic and Cartesia
GObject-Introspection	GObject introspection is a middleware layer between C libraries (using GObject) and language bindings. Th
GROMACS	GROMACS is a versatile package to perform molecular dynamics, i.e. simulate theNewtonian equations of n
GSL	The GNU Scientific Library (GSL) is a numerical library for C and C++ programmers. The library provides
GTK+	GTK+ is the primary library used to construct user interfaces in GNOME. It provides all the user interface c
Gdk-Pixbuf	The Gdk Pixbuf is a toolkit for image loading and pixel buffer manipulation. It is used by GTK+ 2 and GTK
Ghostscript	Ghostscript is a versatile processor for PostScript data with the ability to render PostScript to different targets
GlobalArrays	Global Arrays (GA) is a Partitioned Global Address Space (PGAS) programming model
Go	Go is an open source programming language that makes it easy to build simple, reliable, and efficient softwa
Guile	Guile is a programming language, designed to help programmers create flexible applications that can be exte

SOFTWARE	DESCRIPTION
HDF5	HDF5 is a data model, library, and file format for storing and managing data. It supports an unlimited variety
HMMER	HMMER is used for searching sequence databases for homologs of protein sequences, and for making protei
HMMER2	HMMER is used for searching sequence databases for sequence homologs, and for making sequence alignme
HPL	HPL is a software package that solves a (random) dense linear system in double precision (64 bits
HTSlib	A C library for reading/writing high-throughput sequencing data. This package includes the utilities bgzip ar
HarfBuzz	HarfBuzz is an OpenType text shaping engine.
Hypre	Hypre is a library for solving large, sparse linear systems of equations on massively parallel computers. The
ICU	ICU is a mature, widely used set of C/C++ and Java libraries providing Unicode and Globalization support fo
ImageMagick	ImageMagick
JasPer	The JasPer Project is an open-source initiative to provide a free software-based reference implementation of
Java	Java Platform, Standard Edition (Java SE) lets you develop and deploy Java applications on desktops and ser
Jellyfish	Jellyfish is a tool for fast, memory-efficient counting of k-mers in DNA.
JsonCpp	JsonCpp is a C++ library that allows manipulating JSON values, including serialization and deserialization t
Julia	Julia is a high-level, high-performance dynamic programming language for numerical computing
LAME	LAME is a high quality MPEG Audio Layer III (MP3) encoder licensed under the LGPL.
LAMMPS	LAMMPS is a classical molecular dynamics code, and an acronymfor Large-scale Atomic/Molecular Massiv
LLVM	The LLVM Core libraries provide a modern source- and target-independent optimizer, along with code gener
LMDB	LMDB is a fast, memory-efficient database. With memory-mapped files, it has the read performance of a pur
LibTIFF	tiff: Library and tools for reading and writing TIFF data files
Libint	Libint library is used to evaluate the traditional (electron repulsion) and certain novel two-body matrix eleme
LittleCMS	Little CMS intends to be an OPEN SOURCE small-footprint color management engine, with special focus o
Lua	Lua is a powerful, fast, lightweight, embeddable scripting language. Lua combines simple procedural syntax
M4	GNU M4 is an implementation of the traditional Unix macro processor. It is mostly SVR4 compatible althou
MATLAB	MATLAB is a high-level language and interactive environment that enables you to perform computationally
METIS	METIS is a set of serial programs for partitioning graphs, partitioning finite element meshes, and producing
MPFR	The MPFR library is a C library for multiple-precision floating-point computations with correct rounding.
MUMPS	A parallel sparse direct solver
Mako	A super-fast templating language that borrows the best ideas from the existing templating languages
MariaDB-connector-c	MariaDB Connector/C is used to connect applications developed in C/C++ to MariaDB and MySQL databas
Mathematica	Mathematica is a computational software program used in many scientific, engineering, mathematicaland co
Mesa	Mesa is an open-source implementation of the OpenGL specification - a system for rendering interactive 3D
Meson	Meson is a cross-platform build system designed to be both as fast and as user friendly as possible.
MetaEuk	MetaEuk is a modular toolkit designed for large-scale gene discovery and annotation in eukaryotic metageno
Miniconda3	Miniconda is a free minimal installer for conda. It is a small, bootstrap version of Anaconda that includes on
NAMD	NAMD is a parallel molecular dynamics code designed for high-performance simulation of large biomolecul
NASM	NASM: General-purpose x86 assembler
NLopt	NLopt is a free/open-source library for nonlinear optimization, providing a common interface for a number o
NSPR	Netscape Portable Runtime (NSPR) provides a platform-neutral API for system level and libc-like functions.
NSS	Network Security Services (NSS) is a set of libraries designed to support cross-platform development of secu
NVHPC	C, C++ and Fortran compilers included with the NVIDIA HPC SDK (previously: PGI)
Nastran	NASTRANDESCRIPTION
Ninja	Ninja is a small build system with a focus on speed.
OpenBLAS	OpenBLAS is an optimized BLAS library based on GotoBLAS2 1.13 BSD version.
OpenFOAM	Open source CFD package
OpenMM	OpenMM is a toolkit for molecular simulation.
OpenMPI	The Open MPI Project is an open source MPI-3 implementation.
OpenMolcas	OpenMolcas is a quantum chemistry software package
OpenSSL	The OpenSSL Project is a collaborative effort to develop a robust, commercial-grade, full-featured, and Oper
PCRE	The PCRE library is a set of functions that implement regular expression pattern matching using the same sy

SOFTWARE	DESCRIPTION
PCRE2	The PCRE library is a set of functions that implement regular expression pattern matching using the same sy
PETSc	PETSc, pronounced PET-see (the S is silent), is a suite of data structures and routines for the scalable (parall
PLUMED	PLUMED is an open source library for free energy calculations in molecular systems which works together v
PMIx	Process Management for Exascale EnvironmentsPMI Exascale (PMIx) represents an attempt toprovide an ex
PROJ	Program proj is a standard Unix filter function which convertsgeographic longitude and latitude coordinates
Pango	Pango is a library for laying out and rendering of text, with an emphasis on internationalization.Pango can be
ParaView	ParaView Scientific Visualization
Patran	PATRANDESCRIPTION
Perl	Larry Wall's Practical Extraction and Report Language
Pillow	Pillow is the 'friendly PIL fork' by Alex Clark and Contributors. PIL is the Python Imaging Library by Fred
PyCharm	PyCharm Community Edition: Python IDE for Professional Developers
PyTorch	Tensors and Dynamic neural networks in Python with strong GPU acceleration.PyTorch is a deep learning fr
PyYAML	PyYAML is a YAML parser and emitter for the Python programming language.
Pysam	Pysam is a python module for reading and manipulating Samfiles. It's a lightweight wrapper of the samtools
Python	Python is a programming language that lets you work more quickly and integrate your systems more effectiv
QIIME2	QIIME is an open-source bioinformatics pipeline for performing microbiome analysis from raw DNA seque
Qt5	Qt is a comprehensive cross-platform C++ application framework.
QuantumESPRESSO	Quantum ESPRESSO is an integrated suite of computer codes for electronic-structure calculations and mate
R	R For Statistical Computing
Ruby	Ruby is a dynamic, open source programming language with a focus on simplicity and productivity. It has a
SAMtools	SAM Tools provide various utilities for manipulating alignments in the SAM format, including sorting, merg
SCOTCH	Software package and libraries for sequential and parallel graph partitioning,static mapping, and sparse matr
SCons	SCons is a software construction tool.
SEPP	SATe-enabled Phylogenetic Placement - addresses the problem of phylogeneticplacement of short reads into
SLEPc	SLEPc (Scalable Library for Eigenvalue Problem Computations) is a software library for the solution of larg
SQLite	SQLite: SQL Database Engine in a C Library
SWIG	SWIG is a software development tool that connects programs written in C and C++ with a variety of high-le
ScaFaCoS	ScaFaCoS is a library of scalable fast coulomb solvers.
ScaLAPACK	The ScaLAPACK (or Scalable LAPACK) library includes a subset of LAPACK routines redesigned for distr
SciPy-bundle	Bundle of Python packages for scientific software
Serf	The serf library is a high performance C-based HTTP client library built upon the Apache Portable Runtime
SoX	SoX is the Swiss Army Knife of sound processing utilities. It can convert audio files to other popular audio f
SpaceRanger	Space Ranger is a set of analysis pipelines that process Visium spatial RNA-seq outputand brightfield micros
Subversion	Subversion is an open source version control system.
SuiteSparse	SuiteSparse is a collection of libraries manipulate sparse matrices.
Szip	Szip compression software, providing lossless compression of scientific data
TINKER	The TINKER molecular modeling software is a complete and general package for molecular mechanics and
Tcl	Tcl (Tool Command Language) is a very powerful but easy to learn dynamic programming language, suitabl
TensorFlow	An open-source software library for Machine Intelligence
Tk	Tk is an open source, cross-platform widget toolchain that provides a library of basic elements for building a
Tkinter	Tkinter module, built with the Python buildsystem
TopHat	TopHat is a fast splice junction mapper for RNA-Seq reads.
UCX	Unified Communication XAn open-source production grade communication framework for data centricand h
UDUNITS	UDUNITS supports conversion of unit specifications between formatted and binary forms, arithmetic manip
UnZip	UnZip is an extraction utility for archives compressedin .zip format (also called "zipfiles"). Although highly
VASP	The Vienna Ab initio Simulation Package (VASP) is a computer program for atomic scalematerials modellin
VTK	The Visualization Toolkit (VTK) is an open-source, freely available software system for 3D computer graphi
Valgrind	Valgrind: Debugging and profiling tools
Voro++	Voro++ is a software library for carrying out three-dimensional computations of the Voronoitessellation. A c

SOFTWARE	DESCRIPTION
WRF	The Weather Research and Forecasting (WRF) Model is a next-generation mesoscale numerical weather pred
Wannier90	A tool for obtaining maximally-localised Wannier functions
X11	The X Window System (X11) is a windowing system for bitmap displays
XML-LibXML	Perl binding for libxml2
XZ	xz: XZ utilities
Xvfb	Xvfb is an X server that can run on machines with no display hardware and no physical input devices. It emu
Yasm	Yasm: Complete rewrite of the NASM assembler with BSD license
Zip	Zip is a compression and file packaging/archive utility.Although highly compatible both with PKWARE's PK
amd-uprof	AMD Prof for performance analysis
archspec	A library for detecting, labeling, and reasoning about microarchitectures
aspect-2.2.0	AspectDESCRIPTION
aspect-2.3.0	AspectDESCRIPTION
at-spi2-atk	AT-SPI 2 toolkit bridge
at-spi2-core	Assistive Technology Service Provider Interface.
bcl2fastq2	bcl2fastq Conversion Software both demultiplexes data and converts BCL files generated by Illumina sequen
binutils	binutils: GNU binary utilities
bokeh	Statistical and novel interactive HTML plots for Python
boost-1.58.0	BOOSTDESCRIPTION
bzip2	bzip2 is a freely available, patent free, high-quality data compressor. It typically compresses files to within 1
cURL	libcurl is a free and easy-to-use client-side URL transfer library, supporting DICT, FILE, FTP, FTPS, Gophe
cairo	Cairo is a 2D graphics library with support for multiple output devices. Currently supported output targets in
canu	Canu is a fork of the Celera Assembler designed for high-noise single-molecule sequencing
dask	Dask natively scales Python. Dask provides advanced parallelism for analytics, enabling performance at scal
dealii-9.2.0	SOFTWAREDESCRIPTION
dealii-9.3.1	DEALIIDESCRIPTION
double-conversion	Efficient binary-decimal and decimal-binary conversion routines for IEEE doubles.
dxa	DXA Dislocation Analysis
ea-utils	EA-utils are command line tools for processing biological sequencing data.
expat	Expat is an XML parser library written in C. It is a stream-oriented parser in which an application registers h
flatbuffers	FlatBuffers: Memory Efficient Serialization Library
flatbuffers-python	Python Flatbuffers runtime library.
flex	Github Repository
fontconfig	Fontconfig is a library designed to provide system-wide font configuration, customization and application ac
foss	GNU Compiler Collection (GCC) based compiler toolchain, including OpenMPI for MPI support, OpenBLA
freetype	FreeType 2 is a software font engine that is designed to be small, efficient, highly customizable, and portable
gaussian	Gaussian is a computer program for computational quantum chemistrythat includes various methods for elec
gc	The Boehm-Demers-Weiser conservative garbage collector can be used as a garbage collecting replacement f
gettext	GNU 'gettext' is an important step for the GNU Translation Project, as it is an asset on which we maybuild m
gflags	Google command line flags
giflib	giflib is a library for reading and writing gif images.It is API and ABI compatible with libungif which was in
git	Git is a free and open source distributed version control system designedto handle everything from small to v
glm-0.9.8.5	SOFTWAREDESCRIPTION
glog	Google Logging Library
gmsh	3D finite element grid generator with a build-in CAD engine and post-processor
gnuplot	GNU Plot
gomkl	GNU Compiler Collection (GCC) based compiler toolchain with OpenMPI and MKL
gompi	GNU Compiler Collection (GCC) based compiler toolchain, including OpenMPI for MPI support.
gperf	GNU gperf is a perfect hash function generator. For a given list of strings, it produces a hash function and ha
gperftools	gperftools is a collection of a high-performance multi-threaded malloc(

SOFTWARE	DESCRIPTION
guppyCPU	SOFTWAREDESCRIPTION
gzip	gzip (GNU zip) is a popular data compression program as a replacement for compress
h5py	HDF5 for Python (h5py) is a general-purpose Python interface to the Hierarchical Data Format library, versi
help2man	help2man produces simple manual pages from the '–help' and '–version' output of other commands.
hwloc	The Portable Hardware Locality (hwloc) software package provides a portable abstraction (across OS, versio
hypothesis	Hypothesis is an advanced testing library for Python. It lets you write tests which are parametrized by a sour
iccifort	Intel C, C++ & Fortran compilers
iimpi	Intel C/C++ and Fortran compilers, alongside Intel MPI.
imkl	Intel Math Kernel Library is a library of highly optimized, extensively threaded math routines for science, en
impi	Intel MPI Library, compatible with MPICH ABI
intel	Intel Compilers (C, C++, Fortran)
intltool	intltool is a set of tools to centralize translation of many different file formats using GNU gettext-compatible
iomkl	Intel Cluster Toolchain Compiler Edition provides Intel C/C++ and Fortran compilers, Intel MKL & OpenM
iompi	Intel C/C++ and Fortran compilers, alongside Open MPI.
kaldi	Kaldi is a toolkit for speech recognition
kim-api	Open Knowledgebase of Interatomic Models.KIM is an API and OpenKIM is a collection of interatomic mo
libGLU	The OpenGL Utility Library (GLU) is a computer graphics library for OpenGL.
libarchive	Multi-format archive and compression library
libcerf	libcerf is a self-contained numeric library that provides an efficient and accurate implementation of complex
libdrm	Direct Rendering Manager runtime library.
libepoxy	Epoxy is a library for handling OpenGL function pointer management for you
libevent	The libevent API provides a mechanism to execute a callback function when a specific event occurs on a file
libfabric	Libfabric is a core component of OFI. It is the library that defines and exportsthe user-space API of OFI, and
libffi	The libffi library provides a portable, high level programming interface to various calling conventions. This a
libgd	GD is an open source code library for the dynamic creation of images by programmers.
libgeotiff	Library for reading and writing coordinate system information from/to GeoTIFF files
libglvnd	libglvnd is a vendor-neutral dispatch layer for arbitrating OpenGL API calls between multiple vendors.
libiconv	Libiconv converts from one character encoding to another through Unicode conversion
libjpeg-turbo	libjpeg-turbo is a JPEG image codec that uses SIMD instructions to accelerate baseline JPEG compression a
libmatheval	GNU libmatheval is a library (callable from C and Fortran) to parse and evaluate symbolic expressions input
libogg	Ogg is a multimedia container format, and the native file and stream format for the Xiph.orgmultimedia code
libpciaccess	Generic PCI access library.
libpng	libpng is the official PNG reference library
libreadline	The GNU Readline library provides a set of functions for use by applications that allow users to edit comma
libsndfile	Libsndfile is a C library for reading and writing files containing sampled sound (such as MS Windows WAV
libtool	GNU libtool is a generic library support script. Libtool hides the complexity of using shared libraries behind
libunistring	This library provides functions for manipulating Unicode strings and for manipulating C strings according to
libunwind	The primary goal of libunwind is to define a portable and efficient C programming interface (API) to determ
libvorbis	Ogg Vorbis is a fully open, non-proprietary, patent-and-royalty-free, general-purpose compressedaudio form
libxc	Library of exchange-correlation functionals for density-functional theory
libxml2	Libxml2 is the XML C parser and toolchain developed for the Gnome project (but usable outside of the Gno
libxsmm	LIBXSMM is a library for small dense and small sparse matrix-matrix multiplicationstargeting Intel Archite
libyaml	LibYAML is a YAML parser and emitter written in C.
lpsolve	Mixed Integer Linear Programming (MILP) solver
ls-dyna	LS-DYNA general-purpose finite element simulation software.
ls-prepost	LS-PrePost
lz4	LZ4 is lossless compression algorithm, providing compression speed at 400 MB/s per core. It features an ext
matplotlib	matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy
metis-5.1.0	METISDESCRIPTION

SOFTWARE	DESCRIPTION
minimap2	Minimap2 is a fast sequence mapping and alignmentprogram that can find overlaps between long noisy reads
molmod	MolMod is a Python library with many compoments that are useful to write molecular modeling programs.
mpi4py	MPI for Python (mpi4py) provides bindings of the Message Passing Interface (MPI) standard for the Python
nanopolish	Software package for signal-level analysis of Oxford Nanopore sequencing data.
ncurses	The Ncurses (new curses) library is a free software emulation of curses in System V Release 4.0, and more.
netCDF	NetCDF (network Common Data Form) is a set of software libraries and machine-independent data formats
netCDF-Fortran	NetCDF (network Common Data Form) is a set of software libraries and machine-independent data formats
nettle	Nettle is a cryptographic library that is designed to fit easily in more or less any context: In crypto toolkits fo
networkx	NetworkX is a Python package for the creation, manipulation,and study of the structure, dynamics, and funct
nodejs	Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applicat
nsync	nsync is a C library that exports various synchronization primitives, such as mutexes
numactl	The numactl program allows you to run your application program on specific cpu's and memory nodes. It do
p4est	Parallel adaptive mesh refinement on forests of octrees
p4est-2.2	P4ESTDESCRIPTION
parallel	GNU Parallel
parmetis-4.0.3	PARMETIS4.0.3DESCRIPTION
picard	Picard: A set of Java command line tools for manipulating high-throughput sequencing (HTS) data and form
pixman	Pixman is a low-level software library for pixel manipulation, providing features such as image compositing
pkg-config	pkg-config is a helper tool used when compiling applications and libraries. It helps you insert the correct co
pkgconfig	pkgconfig is a Python module to interface with the pkg-config command line tool
prodigal	Prodigal (Prokaryotic Dynamic Programming Genefinding Algorithm
protobuf	Google Protocol Buffers
protobuf-python	Python Protocol Buffers runtime library.
pybind11	pybind11 is a lightweight header-only library that exposes C++ types in Python and vice versa, mainly to cre
rclone	Rclone is a command line program to sync files and directories to and from a variety of online storage servic
re2c	re2c is a free and open-source lexer generator for C and C++. Its main goal is generatingfast lexers: at least a
scikit-build	Scikit-Build, or skbuild, is an improved build system generatorfor CPython C/C++/Fortran/Cython extension
snappy	Snappy is a compression/decompression library. It does not aimfor maximum compression, or compatibility
sparsehash	An extremely memory-efficient hash_map implementation
starccm+	Engineering simulation software
tbb	Intel Threading Building Blocks
tcsh	Tcsh is an enhanced, but completely compatible version of the Berkeley UNIX C shell (csh). It is a comman
time	The `time' command runs another program, then displays information about the resources used by that progr
tpl-4.4.18	SOFTWAREDESCRIPTION
trilinos-12.18.1	TRILINOS12.18.1DESCRIPTION
typing-extensions	Typing Extensions – Backported and Experimental Type Hints for Python
utf8proc	utf8proc is a small, clean C library that provides Unicode normalization, case-folding, and other operations f
util-linux	Set of Linux utilities
x264	x264
x265	x265 is a free software library and application for encoding video streams into the H.265 AVC compression f
xorg-macros	X.org macros utilities.
yaff	Yaff stands for 'Yet another force field'. It is a pythonic force-field code.
zlib	zlib
zstd	Zstandard is a real-time compression algorithm, providing high compression ratios. It offers a very wide ran
Anaconda	Anaconda python distribution with TensorFlow 1.7 and Pytorch
Anaconda-boost	BOOST template libraries
Anaconda2	Anaconda python distribution, python version 2.7
CLAPACK	LAPACK linear algebra routines in C
JAGS	Just Another Gibbs Sampler (JAGS)

SOFTWARE	DESCRIPTION
LBPM	Lattice Boltzmann simulator
R-gpu	GPU packages for R
R-parallel	Parallel packages for R
RSEM	RSEM is a RNASeq utility
STAR	STAR 2.5
ViennaCL	Set of header files for GPU support
abaqus	Abaqus Research Edition
abinit	ABINIT is a package whose main program allows one to find the total energy, charge density and electronic
abyss	Assembly By Short Sequences - a de novo, parallel, paired-end sequence assembler
allpathslg	Assembly By Short Sequences - a de novo, parallel, paired-end sequence assembler
amber	APBS Molecular Dynamics Simulations Software Package
ansys	Engineering simulation software
ansysEM	Electromagnetics simulation software
apbs-static	Adaptive Poisson Boltzmann Solver
atlas	ATLAS Tuned Linear Algebra (includes LAPACK)
autodocksuite	Automated molecule docking tool
automake	automake tools
bamtools	BAMTools provides both a programmer's API and an end-user's toolkit for handling BAM files.
bcftools	Bcftools
beagle-lib	BEAGLE-lib
bedtools	Bedtools is a fast, flexible toolset for genome arithmetic.
bison	GNU Software
boost	BOOST template libraries
boost-mpi	BOOST template libraries
bowtie	Short read aligner
bowtie2	Bowtie 2 Fast and Sensitive Read Alignment
bwa	Burrows-Wheeler Aligner
cddlib	CDD library for fundamental polyhedral computations
charm	Charm++
cmake	Cross Platform Makefile Generator
comsol	COMSOL
cora	CORA (CORrelation and Analysis) provides an objective evaluation of time-history signal
cp2k	CP2K Open Source Molecular Dynamics
cplex	Optimization
cuda	NVIDIA CUDA Toolkit (C, C++, cuda-gdb, cuda-memcheck, gpu accel)
cudnn	cuDNN
cufflinks	Transcriptome assembly and differential expression analysis for RNA-Seq.
cytoscape	Cytoscape - Network Data Integration, Analysis, and Visualization in a Box
dcw	Geography Database
diamond	Sequence Alignment Tool
discovar	new genome assembler
discovardenovo	new genome assembler
eigen	Eigen is a C++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithm
ensight	CSM and CFD Post processing
espresso	nanoscale material modeling
evolver	evolver
examl	Exascale Maximum Likelihood (ExaML)
extrae	Instrumentation framework to generate execution traces of the most used parallel runtimes.
fastp	An ultra-fast all-in-one FASTQ preprocessor

SOFTWARE	DESCRIPTION
fastqc	FastQC - A quality control tool for high throughput sequence data.
fdk-aac	fdk-aac
fds	Fire Dynamics Simulator and Smokeview
ffmpeg	ffmpeg
fftw	Fastest Fourier Transform in the West
flann	FLANN is a library for performing fast approximate nearest neighbor searches in high dimensional spaces.
flint	Fast Library for Number Theory
gamit-globk	GPS measurement analysis software
gatb	Genome Assembly and Analysis Tool Box
gatk	The Genome Analysis Toolkit or GATK is a software package developed to analyze high-throughput sequen
gcc	GNU Compiler Collection (C, C++, Fortran, Go)
gdal	Geospatial Data Abstraction Library
gdk	NVIDIA GPU Deployment Kit
geos	GEOS (Geometry Engine - Open Source) is a C++ port of the Java Topology Suite (JTS).
glm	OpenGL Mathematics
gmap-gsnap	GMAP/GSNAP 2017-01-10
go	The Go programming language is an open source project to make programmers more productive.
graphviz	Graphviz is open source graph visualization software.
gromacs	GROMACS
gromacs.c9	GROMACS
gshhg	Geography Database
gsl	GNU Scientific Library
guile	GNU Ubiquitous Intelligent Language for Extensions
guppy-cpu	guppy-cpu
harminv	Harminv is a free program (and accompanying library) to solve the problem of harmonic inversion — given
hdf5	Tool suite for managing very large and complex data collections
hisat2	HiSat2 Fast and Sensitive Read Alignment
hmmer	Sequence Homology Search
hpctoolkit	HPCToolkit
hpgl	High Performance Geostatistics Library
hpl	High-Performance Linpack
htk	Hidden Markov Model Toolkit (HTK)
hypre	Hypre is a library for solving large, sparse linear systems of equations on massively parallel computers
igv	The Integrative Genomics Viewer (IGV) is a high-performance visualization tool for interactive exploration
ioapi	Input/Output Applications Programming Interface
iperf	iPerf - The ultimate speed test tool for TCP, UDP and SCTP
jdk	Java Development Kit
lame	lame
lammps	Large-scale Atomic/Molecular Massively Parallel Simulator
lbpm	Lattice Boltzmann simulator
libctl	A Guile-based library implementing flexible control files for scientific simulations.
libmsym	Molecular point group symmetry lib
ls-dyna-mpp	LS-DYNA general-purpose finite element simulation software.
ls-dyna-smp	LS-DYNA general-purpose finite element simulation software.
lua	Lua Scripting Language
luaJIT	Lua Just-in-time compiler
mathematica	Mathematica Technical Computing
meep	Meep is a free finite-difference time-domain (FDTD) simulation software package to model electromagnetic
megan	MEGAN Community Edition - Interactive exploration and analysis of large-scale microbiome sequencing da

SOFTWARE	DESCRIPTION
metis	Multilevel Partitioning Algorithms
minia	genome assembler
mira	MIRA is a multi-pass DNA sequence data assembler/mapper for whole genome and EST/RNASeq projects
mkl	Intel MKL Library
mpe2	MPI Parallel Environment
mpiP	mpiP: Lightweight, Scalable MPI Profiling
mpiblast	mpiBLAST: Open-Source Parallel BLAST
mrbayes	MrBayes is a program for Bayesian inference
mvapich2	OSU MPI
mysql	MySQL: a very fast and reliable SQL database server
namd	NAMD Scalable Molecular Dynamics
namd-gpu	NAMD Scalable Molecular Dynamics
nasm	nasm
nastran	Multidisciplinary Structural Analysis
ncbi-blast+	BLAST sequence alignment
ncl	NCAR Command Language
nco	netCDF Operators
ncview	Ncview
netcdf	Network Common Data Form
netcdf-c	Network Common Data Form
netcdf-c-par	Network Common Data Form
netcdf-cxx	Network Common Data Form
netcdf-fortran	Network Common Data Form
nose	Nose is open-source testing software for python.
numpy	NumPy is the fundamental package for scientific computing in Python
openblas	OpenBLAS is an optimized BLAS library based on GotoBLAS2 1.13 BSD version.
opencv	opencv
openmpi	Open MPI
opensees	OpenSees Earthquake Engineering Simulation
p2fa	Penn Phonetics Lab Forced Aligner
p7zip	p7zip is a quick port of 7z
papi	Performance Application Programming Interface
parallel-netcdf	Parallel Network Common Data Form
parmetis	Multilevel Partitioning Algorithms
patran	FEA Modeling Solution
pcre2	Perl Compatible Regular Expressions
pdtoolkit	Program Database Toolkit (PDT)
perl	The Perl Programming Language
pgi	PGI Compilers (C, C++, Fortran)
phdf5	Tool suite for managing very large and complex data collections
pigz	Parallel implimentation of gzip
prinseq	PRINSEQ for sequence data manipulation
proj	RPROJ.4 (or proj) is a library for performing conversions between cartographic projections.
proovread	proovread
pv	pipe viewer
python	Python
qt	Qt GUI SDK
rstudio	Engineering simulation software
samtools	Samtools

SOFTWARE	DESCRIPTION
scalapack	Scalable Linear Algebra PACKage
scipy	SciPy is open-source software for mathematics, science, and engineering.
scons	Open source software construction / build tool
seqtk	Toolkit for processing sequences in FASTA/Q formats
shrimp	SHRiMP - SHort Read Mapping Package
silo	Silo is a library for IO
singularity	Singularity is an open source container platform designed to be simple, fast, and secure.
sox	Sound eXchange (SoX)
spades	genome assembler
sqlite	SQLite is a relational database management system contained in a C library.
stata	Statistical software
swig	SWIG is a software development tool that connects programs written in C and C++ with a variety of high-lev
szip	Szip compression software
tau	Tuning and Analysis Utilities
tcl	Tool command language
tcltk	Tool Command Language and Toolkit
tecplot	Scientific Visualization
tophat	TopHat A spliced read mapper for RNA-Seq
tpl	Collection of commonly used third party libraries
trilinos	Object-oriented software framework for multiphysics applications
trimmomatic	Trimmomatic: A flexible read trimming tool for Illumina NGS data
trinityrnaseq	RNA-Seq De novo Assembly Using Trinity
udunits	UDUNITS units package
valgrind	Valgrind Tool Suite
vasp	The Vienna Ab initio Simulation Package
vasp-wannier	The Vienna Ab initio Simulation Package
velvet	De Novo Genomic Assembler
vtk	Visualization ToolKit
vtune	Intel VTune Performance Profiler
wannier90	Wannier90
yasm	yasm
apache-ant	Apache Ant for building Java applications
lsopt	LS-OPT is a standalone package with an interface to LS-DYNA.
mpich	OSU MPI
petsc	Portable, Extensible Toolkit for Scientific Computation (PETSc)
singular	Computer algebra system for polynomial computations
nvhpc	NVidia HPC SDK (PGI Compilers, libraries, CUDA, NCCL, NVSHEM, debugger, profiler, HPC container r

4.3 Lists of Software Installed on ARC Systems

Contents:

4.3.1 List of Software Modules on Infer P100 Nodes

We realize this list is long, but we provide it here for users who want to peruse and/or search for what they need. For a more cleanly-formatted option, see *this table*.

```
---------------------------- /cm/local/modulefiles ----------------------------
   apps                  (L)    gcc/9.2.0          openldap
   cluster-tools/9.0            ipmitool/1.8.18    python3
   cmd                          lua/5.3.5          python37
   cmjob                        luajit             shared              (L)
   cuda-dcgm/1.7.1.1            module-git         slurm/slurm/19.05.5 (L)
   dot                          module-info
   freeipmi/1.6.4               null

---------------------------- /usr/share/modulefiles ----------------------------
   DefaultModules (L)

---------------------------- /cm/shared/modulefiles ----------------------------
   bazel/0.26.1
   blacs/openmpi/gcc/64/1.1patch03
   blas/gcc/64/3.8.0
   bonnie++/1.98
   chainer-py37-cuda10.1-gcc/7.1.0
   chainer-py37-cuda10.2-gcc/7.7.0
   cm-eigen3/3.3.7
   cm-pmix3/3.1.4
   cub-cuda10.1/1.8.0
   cub-cuda10.2/1.8.0
   cuda10.1/blas/10.1.243
   cuda10.1/fft/10.1.243
   cuda10.1/nsight/10.1.243
   cuda10.1/profiler/10.1.243
   cuda10.1/toolkit/10.1.243
   cuda10.2/blas/10.2.89
   cuda10.2/fft/10.2.89
   cuda10.2/nsight/10.2.89
   cuda10.2/profiler/10.2.89
   cuda10.2/toolkit/10.2.89
   cuda11.1/blas/11.1.0
   cuda11.1/fft/11.1.0
   cuda11.1/nsight/11.1.0
   cuda11.1/profiler/11.1.0
   cuda11.1/toolkit/11.1.0
   cudnn7.6-cuda10.1/7.6.5.32
   cudnn7.6-cuda10.2/7.6.5.32
   default-environment
   dynet-py37-cuda10.1-gcc/2.1
   dynet-py37-cuda10.2-gcc/2.1
   fastai-py37-cuda10.1-gcc/1.0.60
   fastai-py37-cuda10.2-gcc/1.0.63
   fftw2/openmpi/gcc/64/double/2.1.5
   fftw2/openmpi/gcc/64/float/2.1.5
```

(continues on next page)

(continued from previous page)

```
   fftw3/openmpi/gcc/64/3.3.8
   gcc5/5.5.0
   gdb/8.3.1
   globalarrays/openmpi/gcc/64/5.7
   gpytorch-py37-cuda10.1-gcc/1.0.1
   gpytorch-py37-cuda10.2-gcc/1.2.0
   hdf5/1.10.1
   hdf5_18/1.8.21
   horovod-mxnet-py37-cuda10.1-gcc/0.19.0
   horovod-mxnet-py37-cuda10.2-gcc/0.20.2
   horovod-pytorch-py37-cuda10.1-gcc/0.19.0
   horovod-pytorch-py37-cuda10.2-gcc/0.20.2
   horovod-tensorflow-py37-cuda10.1-gcc/0.19.0
   horovod-tensorflow-py37-cuda10.2-gcc/0.20.2
   hpcx/2.4.0
   hpl/2.3
   hwloc/1.11.11
   intel-tbb-oss/ia32/2020.1
   intel-tbb-oss/intel64/2020.1
   intel/compiler/32/2019/19.0.5
   intel/compiler/64/2019/19.0.5               (D)
   intel/daal/32/2019/5.281
   intel/daal/64/2019/5.281
   intel/gdb/64/2019/4.281
   intel/ipp/32/2019/5.281
   intel/ipp/64/2019/5.281
   intel/itac/2019/5.041
   intel/mkl/32/2019/5.281
   intel/mkl/64/2019/5.281                     (D)
   intel/mpi/32/2019/5.281
   intel/mpi/64/2019/5.281                     (D)
   intel/tbb/32/2019/5.281
   intel/tbb/64/2019/5.281                     (D)
   iozone/3_487
   keras-py37-cuda10.1-gcc/2.3.1
   keras-py37-cuda10.2-gcc/2.3.1
   lapack/gcc/64/3.8.0
   ml-pythondeps-py37-cuda10.1-gcc/3.2.3
   ml-pythondeps-py37-cuda10.2-gcc/4.1.2
   mpich/ge/gcc/64/3.3.2
   mvapich2/gcc/64/2.3.2
   mxnet-py37-cuda10.1-gcc/1.5.1
   mxnet-py37-cuda10.2-gcc/1.7.0
   nccl2-cuda10.1-gcc/2.5.6
   nccl2-cuda10.2-gcc/2.7.8
   netcdf/gcc/64/gcc/64/4.7.3
   netperf/2.7.0
   openblas/dynamic/0.2.20
   opencv3-py37-cuda10.1-gcc/3.4.9
   opencv3-py37-cuda10.2-gcc/3.4.11
   openmpi-geib-cuda10.1-gcc/3.1.4
   openmpi-geib-cuda10.2-gcc/3.1.4
```

(continues on next page)

(continued from previous page)

```
   openmpi/gcc/64/1.10.7
   protobuf3-gcc/3.8.0
   pytorch-py37-cuda10.1-gcc/1.4.0
   pytorch-py37-cuda10.2-gcc/1.6.0
   scalapack/openmpi/gcc/2.1.0
   tensorflow-py37-cuda10.1-gcc/1.15.2
   tensorflow-py37-cuda10.2-gcc/1.15.4
   tensorflow2-py37-cuda10.1-gcc/2.0.0
   tensorflow2-py37-cuda10.2-gcc/2.2.0
   tensorrt-cuda10.1-gcc/6.0.1.5
   tensorrt-cuda10.2-gcc/7.0.0.11
   theano-py37-cuda10.1-gcc/1.0.4
   theano-py37-cuda10.2-gcc/1.0.5
   ucx/1.6.1
   xgboost-py37-cuda10.1-gcc/0.90
   xgboost-py37-cuda10.2-gcc/1.2.0

------------------------------ /apps/modulefiles ------------------------------
   containers/singularity/3.7.2
   infer-broadwell/guppyGPU/4.5.2
   infer-broadwell/matlab/R2021a
   site/infer-broadwell/easybuild/arc.arcadm
   site/infer-broadwell/easybuild/setup      (D)
   site/infer/easybuild/arc.arcadm
   site/infer/easybuild/setup                (L,D)
   useful_scripts                            (L)

----------------- /apps/easybuild/modules/infer-broadwell/all ------------------
   Anaconda3/2020.11
   Autoconf/2.69-GCCcore-8.3.0
   Autoconf/2.69-GCCcore-10.2.0                       (D)
   Automake/1.16.1-GCCcore-8.3.0
   Automake/1.16.2-GCCcore-10.2.0                     (D)
   Autotools/20180311-GCCcore-8.3.0
   Autotools/20200321-GCCcore-10.2.0                  (D)
   Bazel/3.7.2-GCCcore-10.2.0
   Bison/3.3.2-GCCcore-8.3.0
   Bison/3.3.2
   Bison/3.5.3-GCCcore-9.3.0
   Bison/3.5.3
   Bison/3.7.1-GCCcore-10.2.0
   Bison/3.7.1                                        (D)
   Boost.Python/1.71.0-gompic-2019b
   Boost/1.71.0-gompic-2019b
   Boost/1.74.0-GCC-10.2.0                            (D)
   CMake/3.15.3-GCCcore-8.3.0
   CMake/3.16.4-GCCcore-9.3.0
   CMake/3.18.4-GCCcore-10.2.0                        (D)
   CUDA/10.1.243-GCC-8.3.0
   CUDA/10.2.89-GCC-8.3.0
   CUDA/11.1.1-GCC-10.2.0
   CUDA/11.1.1-iccifort-2020.4.304                    (D)
```

(continues on next page)

(continued from previous page)

```
   CUDAcore/11.1.1
   Check/0.15.2-GCCcore-10.2.0
   DB/18.1.32-GCCcore-8.3.0
   DB/18.1.40-GCCcore-10.2.0                          (D)
   Doxygen/1.8.16-GCCcore-8.3.0
   Doxygen/1.8.20-GCCcore-10.2.0                      (D)
   EasyBuild/4.3.3
   EasyBuild/4.3.4
   EasyBuild/4.4.0
   EasyBuild/4.4.2                                    (D)
   Eigen/3.3.7-GCCcore-9.3.0
   Eigen/3.3.7
   Eigen/3.3.8-GCCcore-10.2.0                         (D)
   FFTW/3.3.8-gompi-2020b
   FFTW/3.3.8-gompic-2019b
   FFTW/3.3.8-gompic-2020b                            (D)
   FFmpeg/4.2.1-GCCcore-8.3.0
   FFmpeg/4.3.1-GCCcore-10.2.0                        (D)
   FriBidi/1.0.5-GCCcore-8.3.0
   FriBidi/1.0.10-GCCcore-10.2.0                      (D)
   GCC/8.3.0
   GCC/10.2.0                                         (D)
   GCCcore/8.3.0
   GCCcore/9.3.0
   GCCcore/10.2.0                                     (D)
   GDRCopy/2.1-GCCcore-10.2.0-CUDA-11.1.1
   GMP/6.1.2-GCCcore-8.3.0
   GMP/6.2.0-GCCcore-10.2.0                           (D)
   GROMACS/2020.4-fosscuda-2020b
   GSL/2.6-GCC-8.3.0
   GSL/2.6-GCC-10.2.0                                 (D)
   Guile/1.8.8-GCCcore-8.3.0
   Guile/2.2.7-GCCcore-10.2.0
   Guile/3.0.7-GCCcore-10.2.0                         (D)
   HDF5/1.10.5-gompic-2019b
   HDF5/1.10.6-gompic-2020b
   HDF5/1.10.7-gompic-2020b                           (D)
   ICU/67.1-GCCcore-10.2.0
   Java/11.0.2                                        (11)
   JsonCpp/1.9.4-GCCcore-10.2.0
   LAME/3.100-GCCcore-8.3.0
   LAME/3.100-GCCcore-10.2.0                          (D)
   LAMMPS/3Mar2020-fosscuda-2019b-Python-3.7.4-kokkos
   LMDB/0.9.24-GCCcore-10.2.0
   LibTIFF/4.1.0-GCCcore-10.2.0
   M4/1.4.18-GCCcore-8.3.0
   M4/1.4.18-GCCcore-9.3.0
   M4/1.4.18-GCCcore-10.2.0
   M4/1.4.18                                          (D)
   MPFR/4.1.0-GCCcore-10.2.0
   Meson/0.55.3-GCCcore-10.2.0
   NASM/2.14.02-GCCcore-8.3.0
```

(continues on next page)

(continued from previous page)

```
   NASM/2.15.05-GCCcore-10.2.0                        (D)
   NCCL/2.8.3-CUDA-11.1.1
   Ninja/1.10.1-GCCcore-10.2.0
   OpenBLAS/0.3.7-GCC-8.3.0
   OpenBLAS/0.3.12-GCC-10.2.0                         (D)
   OpenMM/7.5.0-fosscuda-2020b-Python-3.8.6
   OpenMPI/3.1.4-gcccuda-2019b
   OpenMPI/4.0.5-GCC-10.2.0
   OpenMPI/4.0.5-gcccuda-2020b                        (D)
   PCRE/8.43-GCCcore-8.3.0
   PCRE/8.44-GCCcore-10.2.0                           (D)
   PLUMED/2.5.3-fosscuda-2019b-Python-3.7.4
   PMIx/3.1.5-GCCcore-10.2.0
   Perl/5.30.0-GCCcore-8.3.0
   Perl/5.32.0-GCCcore-10.2.0                         (D)
   Pillow/8.0.1-GCCcore-10.2.0
   PyTorch/1.7.1-fosscuda-2020b
   PyYAML/5.3.1-GCCcore-10.2.0
   Python/2.7.16-GCCcore-8.3.0
   Python/2.7.18-GCCcore-10.2.0
   Python/3.7.4-GCCcore-8.3.0
   Python/3.8.6-GCCcore-10.2.0                        (D)
   SQLite/3.29.0-GCCcore-8.3.0
   SQLite/3.33.0-GCCcore-10.2.0                       (D)
   SWIG/4.0.2-GCCcore-10.2.0
   ScaFaCoS/1.0.1-fosscuda-2020b
   ScaLAPACK/2.0.2-gompic-2019b
   ScaLAPACK/2.1.0-gompi-2020b
   ScaLAPACK/2.1.0-gompic-2020b                       (D)
   SciPy-bundle/2019.10-fosscuda-2019b-Python-2.7.16
   SciPy-bundle/2019.10-fosscuda-2019b-Python-3.7.4
   SciPy-bundle/2020.11-fosscuda-2020b                (D)
   Szip/2.1.1-GCCcore-8.3.0
   Szip/2.1.1-GCCcore-9.3.0
   Szip/2.1.1-GCCcore-10.2.0                          (D)
   Tcl/8.6.9-GCCcore-8.3.0
   Tcl/8.6.10-GCCcore-10.2.0                          (D)
   TensorFlow/2.4.1-fosscuda-2020b
   Tk/8.6.9-GCCcore-8.3.0
   Tk/8.6.10-GCCcore-10.2.0                           (D)
   Tkinter/2.7.16-GCCcore-8.3.0
   Tkinter/3.7.4-GCCcore-8.3.0
   Tkinter/3.8.6-GCCcore-10.2.0                       (D)
   UCX/1.9.0-GCCcore-10.2.0-CUDA-11.1.1
   UCX/1.9.0-GCCcore-10.2.0                           (D)
   UnZip/6.0-GCCcore-9.3.0
   UnZip/6.0-GCCcore-10.2.0                           (D)
   Voro++/0.4.6-fosscuda-2019b
   X11/20190717-GCCcore-8.3.0
   X11/20201008-GCCcore-10.2.0                        (D)
   XZ/5.2.4-GCCcore-8.3.0
   XZ/5.2.5-GCCcore-10.2.0                            (D)
```

(continues on next page)

(continued from previous page)

```
   Yasm/1.3.0-GCCcore-8.3.0
   Yasm/1.3.0-GCCcore-10.2.0                          (D)
   Zip/3.0-GCCcore-10.2.0
   archspec/0.1.0-GCCcore-8.3.0-Python-3.7.4
   binutils/2.32-GCCcore-8.3.0
   binutils/2.32
   binutils/2.34-GCCcore-9.3.0
   binutils/2.34
   binutils/2.35-GCCcore-10.2.0
   binutils/2.35                                      (D)
   bzip2/1.0.8-GCCcore-8.3.0
   bzip2/1.0.8-GCCcore-9.3.0
   bzip2/1.0.8-GCCcore-10.2.0                         (D)
   cURL/7.66.0-GCCcore-8.3.0
   cURL/7.69.1-GCCcore-9.3.0
   cURL/7.72.0-GCCcore-10.2.0                         (D)
   cuDNN/8.0.4.30-CUDA-11.1.1
   double-conversion/3.1.5-GCCcore-10.2.0
   expat/2.2.7-GCCcore-8.3.0
   expat/2.2.9-GCCcore-10.2.0                         (D)
   flatbuffers-python/1.12-GCCcore-10.2.0
   flatbuffers/1.12.0-GCCcore-10.2.0
   flex/2.6.4-GCCcore-8.3.0
   flex/2.6.4-GCCcore-9.3.0
   flex/2.6.4-GCCcore-10.2.0
   flex/2.6.4                                         (D)
   fontconfig/2.13.1-GCCcore-8.3.0
   fontconfig/2.13.92-GCCcore-10.2.0                  (D)
   foss/2020b
   fosscuda/2019b
   fosscuda/2020b                                     (D)
   freetype/2.10.1-GCCcore-8.3.0
   freetype/2.10.3-GCCcore-10.2.0                     (D)
   gc/7.6.12-GCCcore-8.3.0
   gc/7.6.12-GCCcore-10.2.0                           (D)
   gcccuda/2019b
   gcccuda/2020b                                      (D)
   gettext/0.19.8.1
   gettext/0.20.1-GCCcore-8.3.0
   gettext/0.21-GCCcore-10.2.0
   gettext/0.21                                       (D)
   giflib/5.2.1-GCCcore-10.2.0
   git/2.28.0-GCCcore-10.2.0-nodocs
   gompi/2020b
   gompic/2019b
   gompic/2020b                                       (D)
   gperf/3.1-GCCcore-8.3.0
   gperf/3.1-GCCcore-10.2.0                           (D)
   groff/1.22.4-GCCcore-8.3.0
   gzip/1.10-GCCcore-8.3.0
   h5py/2.10.0-fosscuda-2019b-Python-3.7.4
   h5py/2.10.0-fosscuda-2020b                         (D)
```

(continues on next page)

(continued from previous page)

```
   help2man/1.47.4
   help2man/1.47.8-GCCcore-8.3.0
   help2man/1.47.12-GCCcore-9.3.0
   help2man/1.47.16-GCCcore-10.2.0                    (D)
   hwloc/1.11.12-GCCcore-8.3.0
   hwloc/2.2.0-GCCcore-10.2.0
   hypothesis/5.41.2-GCCcore-10.2.0
   hypothesis/5.41.5-GCCcore-10.2.0                   (D)
   iccifort/2020.4.304
   iccifortcuda/2020b
   iimpi/2020b
   iimpic/2020b
   imkl/2020.4.304-iimpi-2020b
   imkl/2020.4.304-iimpic-2020b                       (D)
   impi/2019.9.304-iccifort-2020.4.304
   impi/2019.9.304-iccifortcuda-2020b                 (D)
   intel/2020b
   intelcuda/2020b
   intltool/0.51.0-GCCcore-8.3.0
   intltool/0.51.0-GCCcore-10.2.0                     (D)
   kim-api/2.1.3-fosscuda-2019b
   kim-api/2.1.3-fosscuda-2020b                       (D)
   libarchive/3.4.3-GCCcore-10.2.0
   libevent/2.1.12-GCCcore-10.2.0
   libfabric/1.11.0-GCCcore-10.2.0
   libffi/3.2.1-GCCcore-8.3.0
   libffi/3.3-GCCcore-10.2.0                          (D)
   libiconv/1.16-GCCcore-8.3.0
   libiconv/1.16-GCCcore-10.2.0                       (D)
   libjpeg-turbo/2.0.3-GCCcore-8.3.0
   libjpeg-turbo/2.0.5-GCCcore-10.2.0                 (D)
   libmatheval/1.1.11-GCCcore-8.3.0
   libpciaccess/0.14-GCCcore-8.3.0
   libpciaccess/0.16-GCCcore-10.2.0                   (D)
   libpng/1.6.37-GCCcore-8.3.0
   libpng/1.6.37-GCCcore-10.2.0                       (D)
   libreadline/8.0-GCCcore-8.3.0
   libreadline/8.0-GCCcore-9.3.0
   libreadline/8.0-GCCcore-10.2.0                     (D)
   libtool/2.4.6-GCCcore-8.3.0
   libtool/2.4.6-GCCcore-10.2.0                       (D)
   libunistring/0.9.10-GCCcore-8.3.0
   libunistring/0.9.10-GCCcore-10.2.0                 (D)
   libxml2/2.9.9-GCCcore-8.3.0
   libxml2/2.9.10-GCCcore-10.2.0                      (D)
   libyaml/0.2.5-GCCcore-10.2.0
   magma/2.5.4-fosscuda-2020b
   makeinfo/6.7-GCCcore-8.3.0
   matplotlib/2.2.4-fosscuda-2019b-Python-2.7.16
   matplotlib/3.1.1-fosscuda-2019b-Python-3.7.4
   matplotlib/3.3.3-fosscuda-2020b                    (D)
   molmod/1.4.5-fosscuda-2019b-Python-3.7.4
```

(continues on next page)

(continued from previous page)

```
   molmod/1.4.5-fosscuda-2020b                        (D)
   mpi4py/3.0.2-gompi-2020b-timed-pingpong
   mpi4py/3.1.1-gompi-2020b-timed-pingpong            (D)
   ncurses/6.0
   ncurses/6.1-GCCcore-8.3.0
   ncurses/6.2-GCCcore-9.3.0
   ncurses/6.2-GCCcore-10.2.0
   ncurses/6.2                                        (D)
   netCDF-Fortran/4.5.2-gompic-2019b
   netCDF/4.7.1-gompic-2019b
   networkx/2.5-fosscuda-2020b
   nsync/1.24.0-GCCcore-10.2.0
   numactl/2.0.12-GCCcore-8.3.0
   numactl/2.0.13-GCCcore-10.2.0                      (D)
   pkg-config/0.29.2-GCCcore-8.3.0
   pkg-config/0.29.2-GCCcore-10.2.0                   (D)
   pkgconfig/1.5.1-GCCcore-8.3.0-Python-3.7.4
   pkgconfig/1.5.1-GCCcore-10.2.0-python              (D)
   protobuf-python/3.14.0-GCCcore-10.2.0
   protobuf/3.14.0-GCCcore-10.2.0
   pybind11/2.6.0-GCCcore-10.2.0
   scikit-build/0.11.1-fosscuda-2020b
   snappy/1.1.8-GCCcore-10.2.0
   tbb/2019_U9-GCCcore-8.3.0
   typing-extensions/3.7.4.3-GCCcore-10.2.0
   util-linux/2.34-GCCcore-8.3.0
   util-linux/2.36-GCCcore-10.2.0                     (D)
   x264/20190925-GCCcore-8.3.0
   x264/20201026-GCCcore-10.2.0                       (D)
   x265/3.2-GCCcore-8.3.0
   x265/3.3-GCCcore-10.2.0                            (D)
   xorg-macros/1.19.2-GCCcore-8.3.0
   xorg-macros/1.19.2-GCCcore-10.2.0                  (D)
   yaff/1.6.0-fosscuda-2019b-Python-3.7.4
   zlib/1.2.11-GCCcore-8.3.0
   zlib/1.2.11-GCCcore-9.3.0
   zlib/1.2.11-GCCcore-10.2.0
   zlib/1.2.11                                        (D)

  Where:
   Aliases:  Aliases exist: foo/1.2.3 (1.2) means that "module load foo/1.2" will load␣
↪foo/1.2.3
   D:        Default Module
   L:        Module is loaded

Module defaults are chosen based on Find First Rules due to Name/Version/Version modules␣
↪found in the module tree.
See https://lmod.readthedocs.io/en/latest/060_locating.html for details.

Use "module spider" to find all possible modules and extensions.
Use "module keyword key1 key2 ..." to search for all possible modules matching
any of the "keys".
```

(continues on next page)

(continued from previous page)

```

```

4.3.2 List of Software Modules on Infer T4 Nodes

We realize this list is long, but we provide it here for users who want to peruse and/or search for what they need. For a more cleanly-formatted option, see *this table*.

```

---------------------------- /cm/local/modulefiles -----------------------------
   apps                 (L)    gcc/9.2.0         openldap
   cluster-tools/9.0           ipmitool/1.8.18   python3
   cmd                         lua/5.3.5         python37
   cmjob                       luajit            shared               (L)
   cuda-dcgm/1.7.1.1           module-git        slurm/slurm/19.05.5  (L)
   dot                         module-info
   freeipmi/1.6.4              null

---------------------------- /usr/share/modulefiles ----------------------------
   DefaultModules (L)

---------------------------- /cm/shared/modulefiles ----------------------------
   bazel/0.26.1
   blacs/openmpi/gcc/64/1.1patch03
   blas/gcc/64/3.8.0
   bonnie++/1.98
   chainer-py37-cuda10.1-gcc/7.1.0
   chainer-py37-cuda10.2-gcc/7.7.0
   cm-eigen3/3.3.7
   cm-pmix3/3.1.4
   cub-cuda10.1/1.8.0
   cub-cuda10.2/1.8.0
   cuda10.1/blas/10.1.243
   cuda10.1/fft/10.1.243
   cuda10.1/nsight/10.1.243
   cuda10.1/profiler/10.1.243
   cuda10.1/toolkit/10.1.243
   cuda10.2/blas/10.2.89
   cuda10.2/fft/10.2.89
   cuda10.2/nsight/10.2.89
   cuda10.2/profiler/10.2.89
   cuda10.2/toolkit/10.2.89
   cuda11.1/blas/11.1.0
   cuda11.1/fft/11.1.0
   cuda11.1/nsight/11.1.0
   cuda11.1/profiler/11.1.0
   cuda11.1/toolkit/11.1.0
   cudnn7.6-cuda10.1/7.6.5.32
   cudnn7.6-cuda10.2/7.6.5.32
   default-environment
   dynet-py37-cuda10.1-gcc/2.1
```

(continues on next page)

(continued from previous page)

```
    dynet-py37-cuda10.2-gcc/2.1
    fastai-py37-cuda10.1-gcc/1.0.60
    fastai-py37-cuda10.2-gcc/1.0.63
    fftw2/openmpi/gcc/64/double/2.1.5
    fftw2/openmpi/gcc/64/float/2.1.5
    fftw3/openmpi/gcc/64/3.3.8
    gcc5/5.5.0
    gdb/8.3.1
    globalarrays/openmpi/gcc/64/5.7
    gpytorch-py37-cuda10.1-gcc/1.0.1
    gpytorch-py37-cuda10.2-gcc/1.2.0
    hdf5/1.10.1
    hdf5_18/1.8.21
    horovod-mxnet-py37-cuda10.1-gcc/0.19.0
    horovod-mxnet-py37-cuda10.2-gcc/0.20.2
    horovod-pytorch-py37-cuda10.1-gcc/0.19.0
    horovod-pytorch-py37-cuda10.2-gcc/0.20.2
    horovod-tensorflow-py37-cuda10.1-gcc/0.19.0
    horovod-tensorflow-py37-cuda10.2-gcc/0.20.2
    hpcx/2.4.0
    hpl/2.3
    hwloc/1.11.11
    intel-tbb-oss/ia32/2020.1
    intel-tbb-oss/intel64/2020.1
    intel/compiler/32/2019/19.0.5
    intel/compiler/64/2019/19.0.5                (D)
    intel/daal/32/2019/5.281
    intel/daal/64/2019/5.281
    intel/gdb/64/2019/4.281
    intel/ipp/32/2019/5.281
    intel/ipp/64/2019/5.281
    intel/itac/2019/5.041
    intel/mkl/32/2019/5.281
    intel/mkl/64/2019/5.281                      (D)
    intel/mpi/32/2019/5.281
    intel/mpi/64/2019/5.281                      (D)
    intel/tbb/32/2019/5.281
    intel/tbb/64/2019/5.281                      (D)
    iozone/3_487
    keras-py37-cuda10.1-gcc/2.3.1
    keras-py37-cuda10.2-gcc/2.3.1
    lapack/gcc/64/3.8.0
    ml-pythondeps-py37-cuda10.1-gcc/3.2.3
    ml-pythondeps-py37-cuda10.2-gcc/4.1.2
    mpich/ge/gcc/64/3.3.2
    mvapich2/gcc/64/2.3.2
    mxnet-py37-cuda10.1-gcc/1.5.1
    mxnet-py37-cuda10.2-gcc/1.7.0
    nccl2-cuda10.1-gcc/2.5.6
    nccl2-cuda10.2-gcc/2.7.8
    netcdf/gcc/64/gcc/64/4.7.3
    netperf/2.7.0
```

(continues on next page)

(continued from previous page)

```
   openblas/dynamic/0.2.20
   opencv3-py37-cuda10.1-gcc/3.4.9
   opencv3-py37-cuda10.2-gcc/3.4.11
   openmpi-geib-cuda10.1-gcc/3.1.4
   openmpi-geib-cuda10.2-gcc/3.1.4
   openmpi/gcc/64/1.10.7
   protobuf3-gcc/3.8.0
   pytorch-py37-cuda10.1-gcc/1.4.0
   pytorch-py37-cuda10.2-gcc/1.6.0
   scalapack/openmpi/gcc/2.1.0
   tensorflow-py37-cuda10.1-gcc/1.15.2
   tensorflow-py37-cuda10.2-gcc/1.15.4
   tensorflow2-py37-cuda10.1-gcc/2.0.0
   tensorflow2-py37-cuda10.2-gcc/2.2.0
   tensorrt-cuda10.1-gcc/6.0.1.5
   tensorrt-cuda10.2-gcc/7.0.0.11
   theano-py37-cuda10.1-gcc/1.0.4
   theano-py37-cuda10.2-gcc/1.0.5
   ucx/1.6.1
   xgboost-py37-cuda10.1-gcc/0.90
   xgboost-py37-cuda10.2-gcc/1.2.0

------------------------------ /apps/modulefiles ------------------------------
   containers/singularity/3.5.3
   containers/singularity/3.7.1              (D)
   infer-skylake/guppyGPU/4.5.2
   infer-skylake/julia/1.6.1-foss-2020b
   infer-skylake/julia/1.6.1-fosscuda-2020b (D)
   infer-skylake/matlab/R2021a
   site/infer-skylake/easybuild/arc.arcadm
   site/infer-skylake/easybuild/setup       (D)
   site/infer/easybuild/arc.arcadm
   site/infer/easybuild/setup                (L,D)
   useful_scripts                            (L)

------------------ /apps/easybuild/modules/infer-skylake/all -------------------
   Anaconda3/2020.11
   Autoconf/2.69-GCC-5.4.0-2.26
   Autoconf/2.69-GCCcore-7.3.0
   Autoconf/2.69-GCCcore-8.2.0
   Autoconf/2.69-GCCcore-8.3.0
   Autoconf/2.69-GCCcore-9.2.0
   Autoconf/2.69-GCCcore-9.3.0
   Autoconf/2.69-GCCcore-10.2.0                              (D)
   Automake/1.15-GCC-5.4.0-2.26
   Automake/1.16.1-GCCcore-7.3.0
   Automake/1.16.1-GCCcore-8.2.0
   Automake/1.16.1-GCCcore-8.3.0
   Automake/1.16.1-GCCcore-9.2.0
   Automake/1.16.1-GCCcore-9.3.0
   Automake/1.16.2-GCCcore-10.2.0                            (D)
   Autotools/20150215-GCC-5.4.0-2.26
```

(continues on next page)

(continued from previous page)

```
   Autotools/20180311-GCCcore-7.3.0
   Autotools/20180311-GCCcore-8.2.0
   Autotools/20180311-GCCcore-8.3.0
   Autotools/20180311-GCCcore-9.2.0
   Autotools/20180311-GCCcore-9.3.0
   Autotools/20200321-GCCcore-10.2.0                         (D)
   Bazel/3.7.2-GCCcore-10.2.0
   BirdNET/20201214-fosscuda-2019b-Python-3.7.4
   Bison/3.0.4-GCCcore-5.4.0
   Bison/3.0.4-GCCcore-6.4.0
   Bison/3.0.4-GCCcore-7.3.0
   Bison/3.0.4
   Bison/3.0.5-GCCcore-6.4.0
   Bison/3.0.5-GCCcore-7.3.0
   Bison/3.0.5-GCCcore-8.2.0
   Bison/3.0.5
   Bison/3.3.2-GCCcore-8.3.0
   Bison/3.3.2-GCCcore-9.2.0
   Bison/3.3.2
   Bison/3.5.3-GCCcore-9.3.0
   Bison/3.5.3
   Bison/3.7.1-GCCcore-10.2.0
   Bison/3.7.1                                               (D)
   CMake/3.11.4-GCCcore-7.3.0
   CMake/3.12.1-GCCcore-7.3.0
   CMake/3.15.3-GCCcore-8.3.0
   CMake/3.16.4-GCCcore-9.3.0
   CMake/3.18.4-GCCcore-10.2.0                               (D)
   CUDA/8.0.61_375.26-GCC-5.4.0-2.26
   CUDA/9.0.176-GCC-6.4.0-2.28
   CUDA/10.0.130-GCC-6.4.0-2.28
   CUDA/10.1.243-GCC-8.3.0
   CUDA/11.1.1-GCC-10.2.0
   CUDA/11.1.1-iccifort-2020.4.304                           (D)
   CUDAcore/11.1.1
   Check/0.15.2-GCCcore-10.2.0
   Clang/9.0.1-GCC-8.3.0-CUDA-10.1.243
   DB/18.1.32-GCCcore-9.3.0
   DB/18.1.40-GCCcore-10.2.0                                 (D)
   DBus/1.13.12-GCCcore-9.3.0
   DBus/1.13.18-GCCcore-10.2.0                               (D)
   Doxygen/1.8.16-GCCcore-8.3.0
   Doxygen/1.8.17-GCCcore-9.3.0
   Doxygen/1.8.20-GCCcore-10.2.0                             (D)
   EasyBuild/4.1.2
   EasyBuild/4.3.2
   EasyBuild/4.3.3
   EasyBuild/4.3.4
   EasyBuild/4.4.0
   EasyBuild/4.4.2                                           (D)
   Eigen/3.3.7-GCCcore-9.3.0
   Eigen/3.3.8-GCCcore-10.2.0                                (D)
```

(continues on next page)

(continued from previous page)

```
   FFTW/3.3.4-gompi-2016b
   FFTW/3.3.8-gompi-2018b
   FFTW/3.3.8-gompi-2019b
   FFTW/3.3.8-gompi-2020a
   FFTW/3.3.8-gompi-2020b
   FFTW/3.3.8-gompic-2019b
   FFTW/3.3.8-gompic-2020b                            (D)
   FFmpeg/4.2.1-GCCcore-8.3.0
   FFmpeg/4.2.2-GCCcore-9.3.0
   FFmpeg/4.3.1-GCCcore-10.2.0                        (D)
   FriBidi/1.0.5-GCCcore-8.3.0
   FriBidi/1.0.9-GCCcore-9.3.0
   FriBidi/1.0.10-GCCcore-10.2.0                      (D)
   GCC/5.4.0-2.26
   GCC/6.4.0-2.28
   GCC/7.3.0-2.30
   GCC/8.2.0-2.31.1
   GCC/8.3.0
   GCC/9.2.0-2.32
   GCC/9.3.0
   GCC/10.2.0                                         (D)
   GCCcore/5.4.0
   GCCcore/6.4.0
   GCCcore/7.3.0
   GCCcore/8.2.0
   GCCcore/8.3.0
   GCCcore/9.2.0
   GCCcore/9.3.0
   GCCcore/10.2.0                                     (D)
   GDRCopy/2.1-GCCcore-10.2.0-CUDA-11.1.1
   GLib/2.60.1-GCCcore-8.2.0
   GLib/2.62.0-GCCcore-8.3.0
   GLib/2.64.1-GCCcore-9.3.0
   GLib/2.66.1-GCCcore-10.2.0                         (D)
   GMP/6.1.2-GCCcore-7.3.0
   GMP/6.1.2-GCCcore-8.2.0
   GMP/6.1.2-GCCcore-8.3.0
   GMP/6.2.0-GCCcore-9.3.0
   GMP/6.2.0-GCCcore-10.2.0                           (D)
   GROMACS/2020.4-fosscuda-2020b
   GSL/2.1-foss-2016b
   GSL/2.6-foss-2019b
   GSL/2.6-GCC-8.3.0                                  (D)
   Ghostscript/9.50-GCCcore-8.3.0
   HDF5/1.10.5-gompi-2019b
   HDF5/1.10.5-gompic-2019b
   HDF5/1.10.6-gompi-2020a
   HDF5/1.10.7-gompi-2020b
   HDF5/1.10.7-gompic-2020b                           (D)
   ICU/64.2-GCCcore-8.3.0
   ICU/67.1-GCCcore-10.2.0                            (D)
   ImageMagick/7.0.9-5-GCCcore-8.3.0
```

(continues on next page)

(continued from previous page)

```
   JasPer/2.0.14-GCCcore-8.3.0
   JasPer/2.0.14-GCCcore-9.3.0
   JasPer/2.0.24-GCCcore-10.2.0                              (D)
   Java/11.0.2                                               (11)
   JsonCpp/1.9.4-GCCcore-10.2.0
   Julia/1.3.1-linux-x86_64
   LAME/3.100-GCCcore-8.3.0
   LAME/3.100-GCCcore-9.3.0
   LAME/3.100-GCCcore-10.2.0                                 (D)
   LLVM/6.0.0-GCCcore-7.3.0
   LLVM/8.0.1-GCCcore-8.3.0
   LLVM/9.0.0-GCCcore-8.3.0
   LLVM/9.0.1-GCCcore-9.3.0
   LLVM/11.0.0-GCCcore-10.2.0                                (D)
   LMDB/0.9.24-GCCcore-10.2.0
   LibTIFF/4.0.10-GCCcore-8.3.0
   LibTIFF/4.1.0-GCCcore-10.2.0                              (D)
   LittleCMS/2.9-GCCcore-8.3.0
   M4/1.4.17-GCC-5.4.0-2.26
   M4/1.4.17-GCCcore-5.4.0
   M4/1.4.17
   M4/1.4.18-GCCcore-6.4.0
   M4/1.4.18-GCCcore-7.3.0
   M4/1.4.18-GCCcore-8.2.0
   M4/1.4.18-GCCcore-8.3.0
   M4/1.4.18-GCCcore-9.2.0
   M4/1.4.18-GCCcore-9.3.0
   M4/1.4.18-GCCcore-10.2.0
   M4/1.4.18                                                 (D)
   MPFR/4.1.0-GCCcore-10.2.0
   Mako/1.0.7-foss-2018b-Python-2.7.15
   Mako/1.1.0-GCCcore-8.3.0
   Mako/1.1.2-GCCcore-9.3.0
   Mako/1.1.3-GCCcore-10.2.0                                 (D)
   Mesa/18.1.1-foss-2018b
   Mesa/19.1.7-GCCcore-8.3.0
   Mesa/20.0.2-GCCcore-9.3.0
   Mesa/20.2.1-GCCcore-10.2.0                                (D)
   Meson/0.50.0-GCCcore-8.2.0-Python-3.7.2
   Meson/0.51.2-GCCcore-8.3.0-Python-3.7.4
   Meson/0.55.1-GCCcore-9.3.0-Python-3.8.2
   Meson/0.55.3-GCCcore-10.2.0                               (D)
   NASM/2.13.03-GCCcore-7.3.0
   NASM/2.14.02-GCCcore-8.3.0
   NASM/2.14.02-GCCcore-9.3.0
   NASM/2.15.05-GCCcore-10.2.0                               (D)
   NCCL/2.4.8-gcccuda-2019b
   NCCL/2.8.3-CUDA-11.1.1                                    (D)
   NLopt/2.6.1-GCCcore-8.3.0
   NSPR/4.25-GCCcore-9.3.0
   NSPR/4.29-GCCcore-10.2.0                                  (D)
   NSS/3.51-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
   NSS/3.57-GCCcore-10.2.0                                    (D)
   Ninja/1.9.0-GCCcore-8.2.0
   Ninja/1.9.0-GCCcore-8.3.0
   Ninja/1.10.0-GCCcore-9.3.0
   Ninja/1.10.1-GCCcore-10.2.0                                (D)
   OpenBLAS/0.2.18-GCC-5.4.0-2.26-LAPACK-3.6.1
   OpenBLAS/0.3.1-GCC-7.3.0-2.30
   OpenBLAS/0.3.7-GCC-8.3.0
   OpenBLAS/0.3.9-GCC-9.3.0
   OpenBLAS/0.3.12-GCC-10.2.0                                 (D)
   OpenMM/7.4.1-fosscuda-2019b-Python-3.7.4
   OpenMM/7.5.0-fosscuda-2019b-Python-3.7.4
   OpenMM/7.5.0-fosscuda-2020b-Python-3.8.6                   (D)
   OpenMPI/1.10.3-GCC-5.4.0-2.26
   OpenMPI/3.1.1-GCC-7.3.0-2.30
   OpenMPI/3.1.4-GCC-8.3.0
   OpenMPI/3.1.4-gcccuda-2019b
   OpenMPI/4.0.3-GCC-9.2.0-2.32
   OpenMPI/4.0.3-GCC-9.3.0
   OpenMPI/4.0.5-GCC-10.2.0
   OpenMPI/4.0.5-gcccuda-2020b                                (D)
   PCRE/8.43-GCCcore-8.2.0
   PCRE/8.43-GCCcore-8.3.0
   PCRE/8.44-GCCcore-9.3.0
   PCRE/8.44-GCCcore-10.2.0                                   (D)
   PCRE2/10.34-GCCcore-9.3.0
   PCRE2/10.35-GCCcore-10.2.0                                 (D)
   PMIx/3.1.5-GCCcore-9.3.0
   PMIx/3.1.5-GCCcore-10.2.0                                  (D)
   ParaView/5.8.0-foss-2020a-Python-3.8.2-mpi
   ParaView/5.8.1-foss-2020b-mpi                              (D)
   Perl/5.22.1-foss-2016b
   Perl/5.26.1-foss-2019b
   Perl/5.28.0-GCCcore-7.3.0
   Perl/5.28.1-GCCcore-8.2.0
   Perl/5.30.0-GCCcore-8.3.0
   Perl/5.30.2-GCCcore-9.3.0
   Perl/5.32.0-GCCcore-10.2.0                                 (D)
   Pillow/8.0.1-GCCcore-10.2.0
   PyTorch/1.7.1-fosscuda-2020b
   PyYAML/5.3.1-GCCcore-10.2.0
   Python/2.7.15-foss-2018b
   Python/2.7.15-GCCcore-7.3.0-bare
   Python/2.7.16-GCCcore-8.3.0
   Python/2.7.18-GCCcore-9.3.0
   Python/2.7.18-GCCcore-10.2.0
   Python/3.7.2-GCCcore-8.2.0
   Python/3.7.4-GCCcore-8.3.0
   Python/3.8.2-GCCcore-9.3.0
   Python/3.8.6-GCCcore-10.2.0                                (D)
   Qt5/5.14.1-GCCcore-9.3.0
   Qt5/5.14.2-GCCcore-10.2.0                                  (D)
```

(continues on next page)

(continued from previous page)

```
   R/3.6.2-foss-2019b
   SQLite/3.24.0-GCCcore-7.3.0
   SQLite/3.27.2-GCCcore-8.2.0
   SQLite/3.29.0-GCCcore-8.3.0
   SQLite/3.31.1-GCCcore-9.3.0
   SQLite/3.33.0-GCCcore-10.2.0                             (D)
   SWIG/4.0.1-GCCcore-8.3.0
   SWIG/4.0.2-GCCcore-10.2.0                                (D)
   ScaLAPACK/2.0.2-gompi-2016b-OpenBLAS-0.2.18-LAPACK-3.6.1
   ScaLAPACK/2.0.2-gompi-2018b-OpenBLAS-0.3.1
   ScaLAPACK/2.0.2-gompi-2019b
   ScaLAPACK/2.0.2-gompic-2019b
   ScaLAPACK/2.1.0-gompi-2020a
   ScaLAPACK/2.1.0-gompi-2020b
   ScaLAPACK/2.1.0-gompic-2020b                             (D)
   SciPy-bundle/2019.10-fosscuda-2019b-Python-3.7.4
   SciPy-bundle/2020.03-foss-2020a-Python-3.8.2
   SciPy-bundle/2020.11-foss-2020b
   SciPy-bundle/2020.11-fosscuda-2020b                      (D)
   Szip/2.1.1-GCCcore-8.3.0
   Szip/2.1.1-GCCcore-9.3.0
   Szip/2.1.1-GCCcore-10.2.0                                (D)
   Tcl/8.6.8-GCCcore-7.3.0
   Tcl/8.6.9-GCCcore-8.2.0
   Tcl/8.6.9-GCCcore-8.3.0
   Tcl/8.6.10-GCCcore-9.3.0
   Tcl/8.6.10-GCCcore-10.2.0                                (D)
   TensorFlow/2.4.1-fosscuda-2020b
   Theano/1.0.4-fosscuda-2019b-Python-3.7.4
   Tk/8.6.9-GCCcore-8.3.0
   Tk/8.6.10-GCCcore-10.2.0                                 (D)
   Tkinter/3.7.4-GCCcore-8.3.0
   Tkinter/3.8.6-GCCcore-10.2.0                             (D)
   UCX/1.8.0-GCCcore-9.3.0
   UCX/1.9.0-GCCcore-10.2.0-CUDA-11.1.1
   UCX/1.9.0-GCCcore-10.2.0                                 (D)
   UDUNITS/2.2.26-GCCcore-8.3.0
   UnZip/6.0-GCCcore-9.3.0
   UnZip/6.0-GCCcore-10.2.0                                 (D)
   VirtualGL/2.6.1-foss-2018b
   VirtualGL/2.6.2-GCCcore-9.3.0                            (D)
   X11/20180604-GCCcore-7.3.0
   X11/20190311-GCCcore-8.2.0
   X11/20190717-GCCcore-8.3.0
   X11/20200222-GCCcore-9.3.0
   X11/20201008-GCCcore-10.2.0                              (D)
   XML-Parser/2.44_01-GCCcore-7.3.0-Perl-5.28.0
   XZ/5.2.4-GCCcore-7.3.0
   XZ/5.2.4-GCCcore-8.2.0
   XZ/5.2.4-GCCcore-8.3.0
   XZ/5.2.4-GCCcore-9.2.0
   XZ/5.2.5-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
   XZ/5.2.5-GCCcore-10.2.0                                         (D)
   Yasm/1.3.0-GCCcore-8.3.0
   Yasm/1.3.0-GCCcore-9.3.0
   Yasm/1.3.0-GCCcore-10.2.0                                       (D)
   Zip/3.0-GCCcore-10.2.0
   binutils/2.26-GCCcore-5.4.0
   binutils/2.26
   binutils/2.28-GCCcore-6.4.0
   binutils/2.28
   binutils/2.30-GCCcore-7.3.0
   binutils/2.30
   binutils/2.31.1-GCCcore-8.2.0
   binutils/2.31.1
   binutils/2.32-GCCcore-8.3.0
   binutils/2.32-GCCcore-9.2.0
   binutils/2.32
   binutils/2.34-GCCcore-9.3.0
   binutils/2.34
   binutils/2.35-GCCcore-10.2.0
   binutils/2.35                                                   (D)
   bzip2/1.0.6-GCCcore-7.3.0
   bzip2/1.0.6-GCCcore-8.2.0
   bzip2/1.0.8-GCCcore-8.3.0
   bzip2/1.0.8-GCCcore-9.3.0
   bzip2/1.0.8-GCCcore-10.2.0                                      (D)
   cURL/7.60.0-GCCcore-7.3.0
   cURL/7.66.0-GCCcore-8.3.0
   cURL/7.69.1-GCCcore-9.3.0
   cURL/7.72.0-GCCcore-10.2.0                                      (D)
   cairo/1.16.0-GCCcore-8.2.0
   cairo/1.16.0-GCCcore-8.3.0                                      (D)
   cuDNN/7.6.4.38-gcccuda-2019b
   cuDNN/8.0.4.30-CUDA-11.1.1                                      (D)
   double-conversion/3.1.5-GCCcore-9.3.0
   double-conversion/3.1.5-GCCcore-10.2.0                          (D)
   ea-utils/1.04.807-foss-2016b
   ea-utils/1.04.807-foss-2019b                                    (D)
   expat/2.2.5-foss-2019b
   expat/2.2.5-GCCcore-7.3.0
   expat/2.2.6-GCCcore-8.2.0
   expat/2.2.7-GCCcore-8.3.0
   expat/2.2.9-foss-2019b
   expat/2.2.9-GCCcore-9.3.0
   expat/2.2.9-GCCcore-10.2.0                                      (D)
   flatbuffers-python/1.12-GCCcore-10.2.0
   flatbuffers/1.12.0-GCCcore-10.2.0
   flex/2.6.0-GCCcore-5.4.0
   flex/2.6.0
   flex/2.6.3
   flex/2.6.4-GCCcore-6.4.0
   flex/2.6.4-GCCcore-7.3.0
   flex/2.6.4-GCCcore-8.2.0
```

(continues on next page)

(continued from previous page)

```
   flex/2.6.4-GCCcore-8.3.0
   flex/2.6.4-GCCcore-9.2.0
   flex/2.6.4-GCCcore-9.3.0
   flex/2.6.4-GCCcore-10.2.0
   flex/2.6.4                                                  (D)
   fontconfig/2.13.0-GCCcore-7.3.0
   fontconfig/2.13.1-GCCcore-8.2.0
   fontconfig/2.13.1-GCCcore-8.3.0
   fontconfig/2.13.92-GCCcore-9.3.0
   fontconfig/2.13.92-GCCcore-10.2.0                           (D)
   foss/2016b
   foss/2018b
   foss/2019b
   foss/2020a
   foss/2020b                                                  (D)
   fosscuda/2019b
   fosscuda/2020b                                              (D)
   freetype/2.9.1-GCCcore-7.3.0
   freetype/2.9.1-GCCcore-8.2.0
   freetype/2.10.1-GCCcore-8.3.0
   freetype/2.10.1-GCCcore-9.3.0
   freetype/2.10.3-GCCcore-10.2.0                              (D)
   gcccuda/2019b
   gcccuda/2020b                                               (D)
   gettext/0.19.8.1-GCCcore-7.3.0
   gettext/0.19.8.1-GCCcore-8.2.0
   gettext/0.19.8.1
   gettext/0.20.1-GCCcore-8.3.0
   gettext/0.20.1-GCCcore-9.3.0
   gettext/0.20.1
   gettext/0.21-GCCcore-10.2.0
   gettext/0.21                                                (D)
   giflib/5.2.1-GCCcore-10.2.0
   git/2.23.0-GCCcore-8.3.0
   git/2.28.0-GCCcore-10.2.0-nodocs                            (D)
   gompi/2016b
   gompi/2018b
   gompi/2019b
   gompi/2020a
   gompi/2020b                                                 (D)
   gompic/2019b
   gompic/2020b                                                (D)
   gperf/3.1-GCCcore-7.3.0
   gperf/3.1-GCCcore-8.2.0
   gperf/3.1-GCCcore-8.3.0
   gperf/3.1-GCCcore-9.3.0
   gperf/3.1-GCCcore-10.2.0                                    (D)
   groff/1.22.4-GCCcore-9.3.0
   gzip/1.10-GCCcore-9.3.0
   gzip/1.10-GCCcore-10.2.0                                    (D)
   help2man/1.47.4-GCCcore-6.4.0
   help2man/1.47.4-GCCcore-7.3.0
```

(continues on next page)

(continued from previous page)

```
   help2man/1.47.4
   help2man/1.47.7-GCCcore-8.2.0
   help2man/1.47.8-GCCcore-8.3.0
   help2man/1.47.10-GCCcore-9.2.0
   help2man/1.47.12-GCCcore-9.3.0
   help2man/1.47.16-GCCcore-10.2.0                          (D)
   hwloc/1.11.3-GCC-5.4.0-2.26
   hwloc/1.11.10-GCCcore-7.3.0
   hwloc/1.11.12-GCCcore-8.3.0
   hwloc/2.1.0-GCCcore-9.2.0
   hwloc/2.2.0-GCCcore-9.3.0
   hwloc/2.2.0-GCCcore-10.2.0
   hypothesis/5.41.2-GCCcore-10.2.0
   hypothesis/5.41.5-GCCcore-10.2.0                         (D)
   iccifort/2019.5.281
   iccifort/2020.4.304                                      (D)
   iccifortcuda/2020b
   iimpi/2019b
   iimpi/2020b                                              (D)
   iimpic/2020b
   imkl/2019.5.281-iimpi-2019b
   imkl/2020.4.304-iimpi-2020b
   imkl/2020.4.304-iimpic-2020b                             (D)
   impi/2018.5.288-iccifort-2019.5.281
   impi/2019.9.304-iccifort-2020.4.304
   impi/2019.9.304-iccifortcuda-2020b                       (D)
   intel/2019b
   intel/2020b                                              (D)
   intelcuda/2020b
   intltool/0.51.0-GCCcore-7.3.0-Perl-5.28.0
   intltool/0.51.0-GCCcore-8.2.0
   intltool/0.51.0-GCCcore-8.3.0
   intltool/0.51.0-GCCcore-9.3.0
   intltool/0.51.0-GCCcore-10.2.0                           (D)
   libGLU/9.0.0-foss-2018b
   libGLU/9.0.1-GCCcore-8.3.0
   libGLU/9.0.1-GCCcore-9.3.0
   libGLU/9.0.1-GCCcore-10.2.0                              (D)
   libarchive/3.4.3-GCCcore-10.2.0
   libdrm/2.4.92-GCCcore-7.3.0
   libdrm/2.4.99-GCCcore-8.3.0
   libdrm/2.4.100-GCCcore-9.3.0
   libdrm/2.4.102-GCCcore-10.2.0                            (D)
   libevent/2.1.11-GCCcore-9.3.0
   libevent/2.1.12-GCCcore-10.2.0                           (D)
   libfabric/1.11.0-GCCcore-9.3.0
   libfabric/1.11.0-GCCcore-10.2.0                          (D)
   libffi/3.2.1-GCCcore-7.3.0
   libffi/3.2.1-GCCcore-8.2.0
   libffi/3.2.1-GCCcore-8.3.0
   libffi/3.3-GCCcore-9.3.0
   libffi/3.3-GCCcore-10.2.0                                (D)
```

(continues on next page)

(continued from previous page)

```
   libglvnd/1.2.0-GCCcore-9.3.0
   libglvnd/1.3.2-GCCcore-10.2.0                             (D)
   libgpuarray/0.7.6-fosscuda-2019b-Python-3.7.4
   libiconv/1.16-GCCcore-8.3.0
   libiconv/1.16-GCCcore-9.3.0
   libiconv/1.16-GCCcore-10.2.0                              (D)
   libjpeg-turbo/2.0.0-GCCcore-7.3.0
   libjpeg-turbo/2.0.3-GCCcore-8.3.0
   libjpeg-turbo/2.0.4-GCCcore-9.3.0
   libjpeg-turbo/2.0.5-GCCcore-10.2.0                        (D)
   libpciaccess/0.14-GCCcore-7.3.0
   libpciaccess/0.14-GCCcore-8.3.0
   libpciaccess/0.16-GCCcore-9.2.0
   libpciaccess/0.16-GCCcore-9.3.0
   libpciaccess/0.16-GCCcore-10.2.0                          (D)
   libpng/1.6.34-GCCcore-7.3.0
   libpng/1.6.36-GCCcore-8.2.0
   libpng/1.6.37-GCCcore-8.3.0
   libpng/1.6.37-GCCcore-9.3.0
   libpng/1.6.37-GCCcore-10.2.0                              (D)
   libreadline/7.0-GCCcore-7.3.0
   libreadline/8.0-GCCcore-8.2.0
   libreadline/8.0-GCCcore-8.3.0
   libreadline/8.0-GCCcore-9.3.0
   libreadline/8.0-GCCcore-10.2.0                            (D)
   librosa/0.7.2-fosscuda-2019b-Python-3.7.4
   libsndfile/1.0.28-GCCcore-8.3.0
   libtool/2.4.6-GCC-5.4.0-2.26
   libtool/2.4.6-GCCcore-7.3.0
   libtool/2.4.6-GCCcore-8.2.0
   libtool/2.4.6-GCCcore-8.3.0
   libtool/2.4.6-GCCcore-9.2.0
   libtool/2.4.6-GCCcore-9.3.0
   libtool/2.4.6-GCCcore-10.2.0                              (D)
   libunwind/1.2.1-GCCcore-7.3.0
   libunwind/1.3.1-GCCcore-8.3.0
   libunwind/1.3.1-GCCcore-9.3.0
   libunwind/1.4.0-GCCcore-10.2.0                            (D)
   libxml2/2.9.8-GCCcore-7.3.0
   libxml2/2.9.8-GCCcore-8.2.0
   libxml2/2.9.9-GCCcore-8.3.0
   libxml2/2.9.10-GCCcore-9.2.0
   libxml2/2.9.10-GCCcore-9.3.0
   libxml2/2.9.10-GCCcore-10.2.0                             (D)
   libyaml/0.2.5-GCCcore-10.2.0
   lz4/1.9.2-GCCcore-9.3.0
   lz4/1.9.2-GCCcore-10.2.0                                  (D)
   magma/2.5.4-fosscuda-2020b
   makeinfo/6.7-GCCcore-9.3.0
   matplotlib/3.1.1-fosscuda-2019b-Python-3.7.4
   matplotlib/3.3.3-fosscuda-2020b                           (D)
   mpi4py/3.0.2-gompi-2020b-timed-pingpong
```

(continues on next page)

(continued from previous page)

```
   mpi4py/3.1.1-gompi-2020b-timed-pingpong                  (D)
   ncurses/6.0
   ncurses/6.1-GCCcore-7.3.0
   ncurses/6.1-GCCcore-8.2.0
   ncurses/6.1-GCCcore-8.3.0
   ncurses/6.1
   ncurses/6.2-GCCcore-9.3.0
   ncurses/6.2-GCCcore-10.2.0
   ncurses/6.2                                              (D)
   netCDF/4.7.4-gompi-2020a
   netCDF/4.7.4-gompi-2020b                                 (D)
   nettle/3.4-foss-2018b
   nettle/3.5.1-GCCcore-8.3.0                               (D)
   nsync/1.24.0-GCCcore-10.2.0
   numactl/2.0.11-GCC-5.4.0-2.26
   numactl/2.0.11-GCCcore-7.3.0
   numactl/2.0.12-GCCcore-8.3.0
   numactl/2.0.13-GCCcore-9.2.0
   numactl/2.0.13-GCCcore-9.3.0
   numactl/2.0.13-GCCcore-10.2.0                            (D)
   numba/0.47.0-fosscuda-2019b-Python-3.7.4
   pixman/0.38.0-GCCcore-8.2.0
   pixman/0.38.4-GCCcore-8.3.0                              (D)
   pkg-config/0.29.2-GCCcore-7.3.0
   pkg-config/0.29.2-GCCcore-8.2.0
   pkg-config/0.29.2-GCCcore-8.3.0
   pkg-config/0.29.2-GCCcore-9.3.0
   pkg-config/0.29.2-GCCcore-10.2.0                         (D)
   pkgconfig/1.5.1-GCCcore-10.2.0-python
   pocl/1.4-gcccuda-2019b
   protobuf-python/3.14.0-GCCcore-10.2.0
   protobuf/3.14.0-GCCcore-10.2.0
   pybind11/2.4.3-GCCcore-9.3.0-Python-3.8.2
   pybind11/2.6.0-GCCcore-10.2.0                            (D)
   re2c/1.3-GCCcore-9.3.0
   re2c/2.0.3-GCCcore-10.2.0                                (D)
   scikit-learn/0.21.3-fosscuda-2019b-Python-3.7.4
   snappy/1.1.8-GCCcore-9.3.0
   snappy/1.1.8-GCCcore-10.2.0                              (D)
   typing-extensions/3.7.4.3-GCCcore-10.2.0
   util-linux/2.32-GCCcore-7.3.0
   util-linux/2.33-GCCcore-8.2.0
   util-linux/2.34-GCCcore-8.3.0
   util-linux/2.35-GCCcore-9.3.0
   util-linux/2.36-GCCcore-10.2.0                           (D)
   x264/20190925-GCCcore-8.3.0
   x264/20191217-GCCcore-9.3.0
   x264/20201026-GCCcore-10.2.0                             (D)
   x265/3.2-GCCcore-8.3.0
   x265/3.3-GCCcore-9.3.0
   x265/3.3-GCCcore-10.2.0                                  (D)
   xorg-macros/1.19.2-GCCcore-7.3.0
```

(continues on next page)

(continued from previous page)

```
   xorg-macros/1.19.2-GCCcore-8.2.0
   xorg-macros/1.19.2-GCCcore-8.3.0
   xorg-macros/1.19.2-GCCcore-9.2.0
   xorg-macros/1.19.2-GCCcore-9.3.0
   xorg-macros/1.19.2-GCCcore-10.2.0                        (D)
   zlib/1.2.8-GCCcore-5.4.0
   zlib/1.2.8
   zlib/1.2.11-GCCcore-6.4.0
   zlib/1.2.11-GCCcore-7.3.0
   zlib/1.2.11-GCCcore-8.2.0
   zlib/1.2.11-GCCcore-8.3.0
   zlib/1.2.11-GCCcore-9.2.0
   zlib/1.2.11-GCCcore-9.3.0
   zlib/1.2.11-GCCcore-10.2.0
   zlib/1.2.11                                              (D)
   zstd/1.4.4-GCCcore-9.3.0
   zstd/1.4.5-GCCcore-10.2.0                                (D)

  Where:
   Aliases:  Aliases exist: foo/1.2.3 (1.2) means that "module load foo/1.2" will load␣
↪foo/1.2.3
   D:        Default Module
   L:        Module is loaded

Module defaults are chosen based on Find First Rules due to Name/Version/Version modules␣
↪found in the module tree.
See https://lmod.readthedocs.io/en/latest/060_locating.html for details.

Use "module spider" to find all possible modules and extensions.
Use "module keyword key1 key2 ..." to search for all possible modules matching
any of the "keys".
```

4.3.3 List of Software Modules on Infer V100 Nodes

We realize this list is long, but we provide it here for users who want to peruse and/or search for what they need. For a more cleanly-formatted option, see *this table*.

```

---------------------------- /cm/local/modulefiles -----------------------------
   apps                 (L)    gcc/9.2.0         openldap
   cluster-tools/9.0           ipmitool/1.8.18   python3
   cmd                         lua/5.3.5         python37
   cmjob                       luajit            shared                (L)
   cuda-dcgm/1.7.1.1           module-git        slurm/slurm/19.05.5   (L)
   dot                         module-info
   freeipmi/1.6.4              null

---------------------------- /usr/share/modulefiles ----------------------------
   DefaultModules (L)
```

(continues on next page)

(continued from previous page)

```
---------------------------- /cm/shared/modulefiles ----------------------------
   bazel/0.26.1
   blacs/openmpi/gcc/64/1.1patch03
   blas/gcc/64/3.8.0
   bonnie++/1.98
   chainer-py37-cuda10.1-gcc/7.1.0
   chainer-py37-cuda10.2-gcc/7.7.0
   cm-eigen3/3.3.7
   cm-pmix3/3.1.4
   cub-cuda10.1/1.8.0
   cub-cuda10.2/1.8.0
   cuda10.1/blas/10.1.243
   cuda10.1/fft/10.1.243
   cuda10.1/nsight/10.1.243
   cuda10.1/profiler/10.1.243
   cuda10.1/toolkit/10.1.243
   cuda10.2/blas/10.2.89
   cuda10.2/fft/10.2.89
   cuda10.2/nsight/10.2.89
   cuda10.2/profiler/10.2.89
   cuda10.2/toolkit/10.2.89
   cuda11.1/blas/11.1.0
   cuda11.1/fft/11.1.0
   cuda11.1/nsight/11.1.0
   cuda11.1/profiler/11.1.0
   cuda11.1/toolkit/11.1.0
   cudnn7.6-cuda10.1/7.6.5.32
   cudnn7.6-cuda10.2/7.6.5.32
   default-environment
   dynet-py37-cuda10.1-gcc/2.1
   dynet-py37-cuda10.2-gcc/2.1
   fastai-py37-cuda10.1-gcc/1.0.60
   fastai-py37-cuda10.2-gcc/1.0.63
   fftw2/openmpi/gcc/64/double/2.1.5
   fftw2/openmpi/gcc/64/float/2.1.5
   fftw3/openmpi/gcc/64/3.3.8
   gcc5/5.5.0
   gdb/8.3.1
   globalarrays/openmpi/gcc/64/5.7
   gpytorch-py37-cuda10.1-gcc/1.0.1
   gpytorch-py37-cuda10.2-gcc/1.2.0
   hdf5/1.10.1
   hdf5_18/1.8.21
   horovod-mxnet-py37-cuda10.1-gcc/0.19.0
   horovod-mxnet-py37-cuda10.2-gcc/0.20.2
   horovod-pytorch-py37-cuda10.1-gcc/0.19.0
   horovod-pytorch-py37-cuda10.2-gcc/0.20.2
   horovod-tensorflow-py37-cuda10.1-gcc/0.19.0
   horovod-tensorflow-py37-cuda10.2-gcc/0.20.2
   hpcx/2.4.0
   hpl/2.3
```

(continues on next page)

(continued from previous page)

```
    hwloc/1.11.11
    intel-tbb-oss/ia32/2020.1
    intel-tbb-oss/intel64/2020.1
    intel/compiler/32/2019/19.0.5
    intel/compiler/64/2019/19.0.5                (D)
    intel/daal/32/2019/5.281
    intel/daal/64/2019/5.281
    intel/gdb/64/2019/4.281
    intel/ipp/32/2019/5.281
    intel/ipp/64/2019/5.281
    intel/itac/2019/5.041
    intel/mkl/32/2019/5.281
    intel/mkl/64/2019/5.281                      (D)
    intel/mpi/32/2019/5.281
    intel/mpi/64/2019/5.281                      (D)
    intel/tbb/32/2019/5.281
    intel/tbb/64/2019/5.281                      (D)
    iozone/3_487
    keras-py37-cuda10.1-gcc/2.3.1
    keras-py37-cuda10.2-gcc/2.3.1
    lapack/gcc/64/3.8.0
    ml-pythondeps-py37-cuda10.1-gcc/3.2.3
    ml-pythondeps-py37-cuda10.2-gcc/4.1.2
    mpich/ge/gcc/64/3.3.2
    mvapich2/gcc/64/2.3.2
    mxnet-py37-cuda10.1-gcc/1.5.1
    mxnet-py37-cuda10.2-gcc/1.7.0
    nccl2-cuda10.1-gcc/2.5.6
    nccl2-cuda10.2-gcc/2.7.8
    netcdf/gcc/64/gcc/64/4.7.3
    netperf/2.7.0
    openblas/dynamic/0.2.20
    opencv3-py37-cuda10.1-gcc/3.4.9
    opencv3-py37-cuda10.2-gcc/3.4.11
    openmpi-geib-cuda10.1-gcc/3.1.4
    openmpi-geib-cuda10.2-gcc/3.1.4
    openmpi/gcc/64/1.10.7
    protobuf3-gcc/3.8.0
    pytorch-py37-cuda10.1-gcc/1.4.0
    pytorch-py37-cuda10.2-gcc/1.6.0
    scalapack/openmpi/gcc/2.1.0
    tensorflow-py37-cuda10.1-gcc/1.15.2
    tensorflow-py37-cuda10.2-gcc/1.15.4
    tensorflow2-py37-cuda10.1-gcc/2.0.0
    tensorflow2-py37-cuda10.2-gcc/2.2.0
    tensorrt-cuda10.1-gcc/6.0.1.5
    tensorrt-cuda10.2-gcc/7.0.0.11
    theano-py37-cuda10.1-gcc/1.0.4
    theano-py37-cuda10.2-gcc/1.0.5
    ucx/1.6.1
    xgboost-py37-cuda10.1-gcc/0.90
    xgboost-py37-cuda10.2-gcc/1.2.0
```

(continues on next page)

(continued from previous page)

```
------------------------------ /apps/modulefiles ------------------------------
   containers/singularity/3.7.2
   infer-skylake_v100/matlab/R2021a
   site/infer-skylake_v100/easybuild/arc.arcadm
   site/infer-skylake_v100/easybuild/setup      (D)
   site/infer/easybuild/arc.arcadm
   site/infer/easybuild/setup                   (L,D)
   useful_scripts                               (L)

---------------- /apps/easybuild/modules/infer-skylake_v100/all ----------------
   Anaconda3/2020.07
   Anaconda3/2020.11                      (D)
   Autoconf/2.69-GCCcore-8.3.0
   Autoconf/2.69-GCCcore-10.2.0           (D)
   Automake/1.16.1-GCCcore-8.3.0
   Automake/1.16.2-GCCcore-10.2.0         (D)
   Autotools/20180311-GCCcore-8.3.0
   Autotools/20200321-GCCcore-10.2.0      (D)
   Bison/3.3.2-GCCcore-8.3.0
   Bison/3.3.2
   Bison/3.5.3
   Bison/3.7.1-GCCcore-10.2.0
   Bison/3.7.1                            (D)
   CMake/3.15.3-GCCcore-8.3.0
   CMake/3.18.4-GCCcore-10.2.0            (D)
   CUDA/10.1.243-GCC-8.3.0
   CUDA/10.1.243-iccifort-2019.5.281
   CUDA/11.1.1-GCC-10.2.0
   CUDA/11.1.1-iccifort-2020.4.304        (D)
   CUDAcore/11.1.1
   Check/0.15.2-GCCcore-10.2.0
   DB/18.1.32-GCCcore-8.3.0
   DB/18.1.40-GCCcore-10.2.0              (D)
   EasyBuild/4.3.4
   EasyBuild/4.4.0
   EasyBuild/4.4.2                        (D)
   FFTW/3.3.8-gompic-2019b
   FFTW/3.3.8-gompic-2020b                (D)
   GCC/8.3.0
   GCC/10.2.0                             (D)
   GCCcore/8.3.0
   GCCcore/10.2.0                         (D)
   GDRCopy/2.1-GCCcore-10.2.0-CUDA-11.1.1
   M4/1.4.18-GCCcore-8.3.0
   M4/1.4.18-GCCcore-10.2.0
   M4/1.4.18                              (D)
   OpenBLAS/0.3.7-GCC-8.3.0
   OpenBLAS/0.3.12-GCC-10.2.0             (D)
   OpenMPI/3.1.4-gcccuda-2019b
   OpenMPI/4.0.5-gcccuda-2020b            (D)
   PMIx/3.1.5-GCCcore-10.2.0
```

(continues on next page)

(continued from previous page)

```
   Perl/5.30.0-GCCcore-8.3.0
   Perl/5.32.0-GCCcore-10.2.0             (D)
   ScaLAPACK/2.0.2-gompic-2019b
   ScaLAPACK/2.1.0-gompic-2020b           (D)
   UCX/1.9.0-GCCcore-10.2.0-CUDA-11.1.1
   XZ/5.2.4-GCCcore-8.3.0
   XZ/5.2.5-GCCcore-10.2.0                (D)
   binutils/2.32-GCCcore-8.3.0
   binutils/2.32
   binutils/2.35-GCCcore-10.2.0
   binutils/2.35                          (D)
   bzip2/1.0.8-GCCcore-8.3.0
   bzip2/1.0.8-GCCcore-10.2.0             (D)
   cURL/7.66.0-GCCcore-8.3.0
   cURL/7.72.0-GCCcore-10.2.0             (D)
   expat/2.2.7-GCCcore-8.3.0
   expat/2.2.9-GCCcore-10.2.0             (D)
   flex/2.6.4-GCCcore-8.3.0
   flex/2.6.4-GCCcore-10.2.0
   flex/2.6.4                             (D)
   fosscuda/2019b
   fosscuda/2020b                         (D)
   gcccuda/2019b
   gcccuda/2020b                          (D)
   gettext/0.19.8.1
   gettext/0.21                           (D)
   gompic/2019b
   gompic/2020b                           (D)
   groff/1.22.4-GCCcore-8.3.0
   groff/1.22.4-GCCcore-10.2.0            (D)
   help2man/1.47.4
   help2man/1.47.8-GCCcore-8.3.0
   help2man/1.47.16-GCCcore-10.2.0        (D)
   hwloc/1.11.12-GCCcore-8.3.0
   hwloc/2.2.0-GCCcore-10.2.0
   iccifort/2019.5.281
   iccifort/2020.4.304                    (D)
   libarchive/3.4.3-GCCcore-10.2.0
   libevent/2.1.12-GCCcore-10.2.0
   libfabric/1.11.0-GCCcore-10.2.0
   libpciaccess/0.14-GCCcore-8.3.0
   libpciaccess/0.16-GCCcore-10.2.0       (D)
   libreadline/8.0-GCCcore-8.3.0
   libreadline/8.0-GCCcore-10.2.0         (D)
   libtool/2.4.6-GCCcore-8.3.0
   libtool/2.4.6-GCCcore-10.2.0           (D)
   libxml2/2.9.9-GCCcore-8.3.0
   libxml2/2.9.10-GCCcore-10.2.0          (D)
   makeinfo/6.7-GCCcore-8.3.0
   makeinfo/6.7-GCCcore-10.2.0            (D)
   ncurses/6.0
   ncurses/6.1-GCCcore-8.3.0
```

(continues on next page)

(continued from previous page)

```
   ncurses/6.2-GCCcore-10.2.0
   ncurses/6.2                          (D)
   numactl/2.0.12-GCCcore-8.3.0
   numactl/2.0.13-GCCcore-10.2.0        (D)
   pkg-config/0.29.2-GCCcore-10.2.0
   xorg-macros/1.19.2-GCCcore-8.3.0
   xorg-macros/1.19.2-GCCcore-10.2.0    (D)
   zlib/1.2.11-GCCcore-8.3.0
   zlib/1.2.11-GCCcore-10.2.0
   zlib/1.2.11                          (D)

  Where:
   D:  Default Module
   L:  Module is loaded

Module defaults are chosen based on Find First Rules due to Name/Version/Version modules␣
↪found in the module tree.
See https://lmod.readthedocs.io/en/latest/060_locating.html for details.

Use "module spider" to find all possible modules and extensions.
Use "module keyword key1 key2 ..." to search for all possible modules matching
any of the "keys".
```

4.3.4 List of Software Modules on TinkerCliffs A100 Nodes

We realize this list is long, but we provide it here for users who want to peruse and/or search for what they need. For a more cleanly-formatted option, see *this table*.

```

---------------------------- /cm/local/modulefiles ----------------------------
   apps                  (L)    ipmitool/1.8.18
   cluster-tools/8.2            lua/5.3.5
   cm-cloud-copy/8.2            module-git
   cmd                          module-info
   cmsub                        null
   cray                  (L)    openldap
   cuda-dcgm/2.0.15.1           openmpi/mlnx/gcc/64/4.0.3rc4
   dot                          python2
   freeipmi/1.6.2               python36
   gcc/8.2.0                    shared                      (L)

---------------------------- /usr/share/modulefiles ---------------------------
   DefaultModules (L)

---------------------------- /cm/shared/modulefiles ---------------------------
   amd-blis/aocc/64/2.1
   amd-blis/gcc/64/2.1
   amd-libflame/aocc/64/2.1
   amd-libflame/gcc/64/2.1
```

(continues on next page)

(continued from previous page)

```
   aocc/aocc-compiler-2.1.0
   aocc/aocc-compiler-2.2.0           (D)
   blacs/openmpi/gcc/64/1.1patch03
   blas/gcc/64/3.8.0
   bonnie++/1.97.3
   cm-pmix3/3.1.4
   cuda-latest/blas/11.2.0
   cuda-latest/fft/11.2.0
   cuda-latest/nsight/11.2.0
   cuda-latest/profiler/11.2.0
   cuda-latest/toolkit/11.2.0         (L)
   cuda11.2/blas/11.2.0
   cuda11.2/fft/11.2.0
   cuda11.2/nsight/11.2.0
   cuda11.2/profiler/11.2.0
   cuda11.2/toolkit/11.2.0
   default-environment
   fftw2/openmpi/gcc/64/double/2.1.5
   fftw2/openmpi/gcc/64/float/2.1.5
   fftw3/openmpi/gcc/64/3.3.8
   gdb/8.2
   globalarrays/openmpi/gcc/64/5.7
   hdf5/1.10.1
   hdf5_18/1.8.20
   hpl/2.2
   hwloc/1.11.11
   ics/2020.0
   intel-tbb-oss/ia32/2020.2
   intel-tbb-oss/intel64/2020.2
   iozone/3_482
   lapack/gcc/64/3.8.0
   mpich/ge/gcc/64/3.3
   mvapich2/gcc/64/2.3.2
   netcdf/gcc/64/4.6.1
   netperf/2.7.0
   openblas/dynamic/0.2.20
   openmpi/gcc/64/1.10.7
   openmpi/gcc/64/4.0.3
   openmpi/gcc/64/4.0.4               (D)
   openmpi/ics/64/4.0.3
   scalapack/openmpi/gcc/64/2.0.2
   sge/2011.11p1
   slurm/20.02.3                      (L)
   ucx/1.6.0

------------------------------ /apps/modulefiles -------------------------------
   containers/singularity/3.7.1
   site/tinkercliffs-rome_a100/easybuild/arc.arcadm
   site/tinkercliffs-rome_a100/easybuild/setup      (D)
   site/tinkercliffs/easybuild/arc.arcadm
   site/tinkercliffs/easybuild/setup                (L,D)
   tinkercliffs-rome_a100/matlab/R2021a
```

(continues on next page)

(continued from previous page)

```
   useful_scripts                                     (L)

------------------------------- /opt/modulefiles -------------------------------
   gcc/8.1.0

---------------------------- /opt/cray/modulefiles -----------------------------
   PrgEnv-cray/1.0.6

--------------------------- /opt/cray/pe/modulefiles ---------------------------
   cce/10.0.0                 cray-mvapich2_nogpu/2.3.4
   cdt/20.05                  cray-mvapich2_nogpu_gnu/2.3.3
   cray-ccdb/3.0.5            cray-mvapich2_nogpu_gnu/2.3.4        (D)
   cray-cti/1.0.7             craype-dl-plugin-py3/mvapich/20.05.1
   cray-fftw/3.3.8.5          craype-dl-plugin-py3/openmpi/20.05.1
   cray-fftw_impi/3.3.8.5     craype/2.6.4
   cray-impi/5                craypkg-gen/1.3.7
   cray-lgdb/3.0.10           gdb4hpc/3.0.10
   cray-libsci/20.03.1        papi/5.7.0.3
   cray-mvapich2/2.3.3        perftools-base/20.03.0
   cray-mvapich2_gnu/2.3.3    valgrind4hpc/1.0.1

------------------- /opt/cray/pe/craype/default/modulefiles --------------------
   craype-accel-nvidia20    craype-ivybridge
   craype-accel-nvidia35    craype-mic-knl
   craype-accel-nvidia52    craype-network-infiniband (L)
   craype-accel-nvidia60    craype-network-opa
   craype-accel-nvidia70    craype-sandybridge
   craype-broadwell         craype-x86-rome           (L)
   craype-haswell           craype-x86-skylake

-------------- /apps/easybuild/modules/tinkercliffs-rome_a100/all --------------
   ABAQUS/2018
   ATK/2.34.1-GCCcore-8.3.0
   ATK/2.36.0-GCCcore-10.2.0                               (D)
   Anaconda3/2020.07
   Anaconda3/2020.11                                       (D)
   Autoconf/2.69-GCCcore-8.3.0
   Autoconf/2.69-GCCcore-10.2.0                            (D)
   Automake/1.16.1-GCCcore-8.3.0
   Automake/1.16.2-GCCcore-10.2.0                          (D)
   Autotools/20180311-GCCcore-8.3.0
   Autotools/20200321-GCCcore-10.2.0                       (D)
   Bazel/3.7.2-GCCcore-10.2.0
   Bison/3.3.2-GCCcore-8.3.0
   Bison/3.3.2
   Bison/3.5.3
   Bison/3.7.1-GCCcore-10.2.0
   Bison/3.7.1                                             (D)
   Boost/1.74.0-GCC-10.2.0
   CMake/3.15.3-GCCcore-8.3.0
   CMake/3.18.4-GCCcore-10.2.0                             (D)
   CUDA/11.1.1-GCC-10.2.0
```

(continues on next page)

(continued from previous page)

```
   CUDAcore/11.1.1
   Check/0.15.2-GCCcore-10.2.0
   DB/18.1.32-GCCcore-8.3.0
   DB/18.1.40-GCCcore-10.2.0                        (D)
   DBus/1.13.12-GCCcore-8.3.0
   Doxygen/1.8.20-GCCcore-10.2.0
   EasyBuild/4.3.4
   EasyBuild/4.4.0
   EasyBuild/4.4.2                                  (D)
   Eigen/3.3.8-GCCcore-10.2.0
   FFTW/3.3.8-gompi-2020b
   FFTW/3.3.8-gompic-2020b                          (D)
   FFmpeg/4.3.1-GCCcore-10.2.0
   FriBidi/1.0.5-GCCcore-8.3.0
   FriBidi/1.0.10-GCCcore-10.2.0                    (D)
   GCC/10.2.0
   GCCcore/8.3.0
   GCCcore/10.2.0                                   (D)
   GDRCopy/2.1-GCCcore-10.2.0-CUDA-11.1.1
   GLib/2.62.0-GCCcore-8.3.0
   GLib/2.66.1-GCCcore-10.2.0                       (D)
   GMP/6.1.2-GCCcore-8.3.0
   GMP/6.2.0-GCCcore-10.2.0                         (D)
   GObject-Introspection/1.63.1-GCCcore-8.3.0-Python-3.7.4
   GObject-Introspection/1.66.1-GCCcore-10.2.0      (D)
   HDF5/1.10.7-gompic-2020b
   ICU/64.2-GCCcore-8.3.0
   ICU/67.1-GCCcore-10.2.0                          (D)
   Java/11.0.2                                      (11)
   JsonCpp/1.9.4-GCCcore-10.2.0
   LAME/3.100-GCCcore-10.2.0
   LMDB/0.9.24-GCCcore-10.2.0
   LibTIFF/4.0.10-GCCcore-8.3.0
   LibTIFF/4.1.0-GCCcore-10.2.0                     (D)
   M4/1.4.18-GCCcore-8.3.0
   M4/1.4.18-GCCcore-10.2.0
   M4/1.4.18                                        (D)
   MPFR/4.1.0-GCCcore-10.2.0
   Meson/0.51.2-GCCcore-8.3.0-Python-3.7.4
   Meson/0.55.3-GCCcore-10.2.0                      (D)
   NASM/2.14.02-GCCcore-8.3.0
   NASM/2.15.05-GCCcore-10.2.0                      (D)
   NCCL/2.8.3-CUDA-11.1.1
   NVHPC/21.2
   Ninja/1.9.0-GCCcore-8.3.0
   Ninja/1.10.1-GCCcore-10.2.0                      (D)
   OpenBLAS/0.3.12-GCC-10.2.0
   OpenMM/7.5.0-fosscuda-2020b
   OpenMM/7.5.1-fosscuda-2020b                      (D)
   OpenMPI/4.0.5-GCC-10.2.0
   OpenMPI/4.0.5-gcccuda-2020b                      (D)
   PCRE/8.43-GCCcore-8.3.0
```

(continues on next page)

(continued from previous page)

```
   PCRE/8.44-GCCcore-10.2.0                                    (D)
   PMIx/3.1.5-GCCcore-10.2.0
   Perl/5.30.0-GCCcore-8.3.0
   Perl/5.32.0-GCCcore-10.2.0                                  (D)
   Pillow/8.0.1-GCCcore-10.2.0
   PyCharm/2021.1.1
   PyTorch/1.7.1-fosscuda-2020b
   PyYAML/5.3.1-GCCcore-10.2.0
   Python/2.7.18-GCCcore-10.2.0
   Python/3.7.4-GCCcore-8.3.0
   Python/3.8.6-GCCcore-10.2.0                                 (D)
   SQLite/3.29.0-GCCcore-8.3.0
   SQLite/3.33.0-GCCcore-10.2.0                                (D)
   SWIG/4.0.2-GCCcore-10.2.0
   ScaLAPACK/2.1.0-gompi-2020b
   ScaLAPACK/2.1.0-gompic-2020b                                (D)
   SciPy-bundle/2020.11-fosscuda-2020b
   Szip/2.1.1-GCCcore-10.2.0
   Tcl/8.6.9-GCCcore-8.3.0
   Tcl/8.6.10-GCCcore-10.2.0                                   (D)
   TensorFlow/2.4.1-fosscuda-2020b
   UCX/1.9.0-GCCcore-10.2.0-CUDA-11.1.1
   UCX/1.9.0-GCCcore-10.2.0                                    (D)
   UnZip/6.0-GCCcore-10.2.0
   X11/20190717-GCCcore-8.3.0
   X11/20201008-GCCcore-10.2.0                                 (D)
   XZ/5.2.4-GCCcore-8.3.0
   XZ/5.2.5-GCCcore-10.2.0                                     (D)
   Yasm/1.3.0-GCCcore-10.2.0
   Zip/3.0-GCCcore-10.2.0
   binutils/2.32-GCCcore-8.3.0
   binutils/2.32
   binutils/2.34
   binutils/2.35-GCCcore-10.2.0
   binutils/2.35                                               (D)
   bzip2/1.0.8-GCCcore-8.3.0
   bzip2/1.0.8-GCCcore-10.2.0                                  (D)
   cURL/7.66.0-GCCcore-8.3.0
   cURL/7.72.0-GCCcore-10.2.0                                  (D)
   cairo/1.16.0-GCCcore-8.3.0
   cairo/1.16.0-GCCcore-10.2.0                                 (D)
   cuDNN/8.0.4.30-CUDA-11.1.1
   double-conversion/3.1.5-GCCcore-10.2.0
   expat/2.2.7-GCCcore-8.3.0
   expat/2.2.9-GCCcore-10.2.0                                  (D)
   flatbuffers-python/1.12-GCCcore-10.2.0
   flatbuffers/1.12.0-GCCcore-10.2.0
   flex/2.6.4-GCCcore-8.3.0
   flex/2.6.4-GCCcore-10.2.0
   flex/2.6.4                                                  (D)
   fontconfig/2.13.1-GCCcore-8.3.0
   fontconfig/2.13.92-GCCcore-10.2.0                           (D)
```

(continues on next page)

(continued from previous page)

```
   foss/2020b
   fosscuda/2020b
   freetype/2.10.1-GCCcore-8.3.0
   freetype/2.10.3-GCCcore-10.2.0                        (D)
   gcccuda/2020b
   gettext/0.19.8.1
   gettext/0.20.1-GCCcore-8.3.0
   gettext/0.21-GCCcore-10.2.0
   gettext/0.21                                          (D)
   giflib/5.2.1-GCCcore-10.2.0
   git/2.28.0-GCCcore-10.2.0-nodocs
   gompi/2020b
   gompic/2020b
   gperf/3.1-GCCcore-8.3.0
   gperf/3.1-GCCcore-10.2.0                              (D)
   groff/1.22.4-GCCcore-8.3.0
   groff/1.22.4-GCCcore-10.2.0                           (D)
   help2man/1.47.4
   help2man/1.47.8-GCCcore-8.3.0
   help2man/1.47.16-GCCcore-10.2.0                       (D)
   hwloc/2.2.0-GCCcore-10.2.0
   hypothesis/5.41.2-GCCcore-10.2.0
   hypothesis/5.41.5-GCCcore-10.2.0                      (D)
   intltool/0.51.0-GCCcore-8.3.0
   intltool/0.51.0-GCCcore-10.2.0                        (D)
   libarchive/3.4.3-GCCcore-10.2.0
   libevent/2.1.12-GCCcore-10.2.0
   libfabric/1.11.0-GCCcore-10.2.0
   libffi/3.2.1-GCCcore-8.3.0
   libffi/3.3-GCCcore-10.2.0                             (D)
   libiconv/1.16-GCCcore-10.2.0
   libjpeg-turbo/2.0.3-GCCcore-8.3.0
   libjpeg-turbo/2.0.5-GCCcore-10.2.0                    (D)
   libpciaccess/0.16-GCCcore-10.2.0
   libpng/1.6.37-GCCcore-8.3.0
   libpng/1.6.37-GCCcore-10.2.0                          (D)
   libreadline/8.0-GCCcore-8.3.0
   libreadline/8.0-GCCcore-10.2.0                        (D)
   libtool/2.4.6-GCCcore-8.3.0
   libtool/2.4.6-GCCcore-10.2.0                          (D)
   libxml2/2.9.9-GCCcore-8.3.0
   libxml2/2.9.10-GCCcore-10.2.0                         (D)
   libyaml/0.2.5-GCCcore-10.2.0
   magma/2.5.4-fosscuda-2020b
   makeinfo/6.7-GCCcore-8.3.0
   makeinfo/6.7-GCCcore-10.2.0                           (D)
   mpi4py/3.0.2-gompi-2020b-timed-pingpong
   mpi4py/3.1.1-gompi-2020b-timed-pingpong               (D)
   ncurses/6.0
   ncurses/6.1-GCCcore-8.3.0
   ncurses/6.2-GCCcore-10.2.0
   ncurses/6.2                                           (D)
```

(continues on next page)

(continued from previous page)

```
   nsync/1.24.0-GCCcore-10.2.0
   numactl/2.0.13-GCCcore-10.2.0
   pixman/0.38.4-GCCcore-8.3.0
   pixman/0.40.0-GCCcore-10.2.0                         (D)
   pkg-config/0.29.2-GCCcore-8.3.0
   pkg-config/0.29.2-GCCcore-10.2.0                     (D)
   pkgconfig/1.5.1-GCCcore-10.2.0-python
   protobuf-python/3.14.0-GCCcore-10.2.0
   protobuf/3.14.0-GCCcore-10.2.0
   pybind11/2.6.0-GCCcore-10.2.0
   snappy/1.1.8-GCCcore-10.2.0
   typing-extensions/3.7.4.3-GCCcore-10.2.0
   util-linux/2.34-GCCcore-8.3.0
   util-linux/2.36-GCCcore-10.2.0                       (D)
   x264/20201026-GCCcore-10.2.0
   x265/3.3-GCCcore-10.2.0
   xorg-macros/1.19.2-GCCcore-8.3.0
   xorg-macros/1.19.2-GCCcore-10.2.0                    (D)
   zlib/1.2.11-GCCcore-8.3.0
   zlib/1.2.11-GCCcore-10.2.0
   zlib/1.2.11                                          (D)

  Where:
   Aliases:  Aliases exist: foo/1.2.3 (1.2) means that "module load foo/1.2" will load␣
↪foo/1.2.3
   D:        Default Module
   L:        Module is loaded

Module defaults are chosen based on Find First Rules due to Name/Version/Version modules␣
↪found in the module tree.
See https://lmod.readthedocs.io/en/latest/060_locating.html for details.

Use "module spider" to find all possible modules and extensions.
Use "module keyword key1 key2 ..." to search for all possible modules matching
any of the "keys".
```

4.3.5 List of Software Modules on TinkerCliffs Intel AP Nodes

We realize this list is long, but we provide it here for users who want to peruse and/or search for what they need. For a more cleanly-formatted option, see *this table*.

```

---------------------------- /cm/local/modulefiles -----------------------------
   apps                  (L)    lua/5.3.5
   cluster-tools/8.2            module-git
   cm-cloud-copy/8.2            module-info
   cmd                          null
   cmsub                        openldap
   cray                  (L)    openmpi/mlnx/gcc/64/4.0.3rc4
```

(continues on next page)

(continued from previous page)

```
   dot                         python2
   freeipmi/1.6.2              python36
   gcc/8.2.0                   shared                        (L)
   ipmitool/1.8.18

---------------------------- /usr/share/modulefiles ----------------------------
   DefaultModules (L)

---------------------------- /cm/shared/modulefiles ----------------------------
   amd-blis/aocc/64/2.1
   amd-blis/gcc/64/2.1
   amd-libflame/aocc/64/2.1
   amd-libflame/gcc/64/2.1
   aocc/aocc-compiler-2.1.0
   aocc/aocc-compiler-2.2.0          (D)
   blacs/openmpi/gcc/64/1.1patch03
   blas/gcc/64/3.8.0
   bonnie++/1.97.3
   cm-pmix3/3.1.4
   cuda-latest/blas/11.2.0
   cuda-latest/fft/11.2.0
   cuda-latest/nsight/11.2.0
   cuda-latest/profiler/11.2.0
   cuda-latest/toolkit/11.2.0
   cuda11.2/blas/11.2.0
   cuda11.2/fft/11.2.0
   cuda11.2/nsight/11.2.0
   cuda11.2/profiler/11.2.0
   cuda11.2/toolkit/11.2.0
   default-environment
   fftw2/openmpi/gcc/64/double/2.1.5
   fftw2/openmpi/gcc/64/float/2.1.5
   fftw3/openmpi/gcc/64/3.3.8
   gdb/8.2
   globalarrays/openmpi/gcc/64/5.7
   hdf5/1.10.1
   hdf5_18/1.8.20
   hpl/2.2
   hwloc/1.11.11
   ics/2020.0
   intel-tbb-oss/ia32/2020.2
   intel-tbb-oss/intel64/2020.2
   iozone/3_482
   lapack/gcc/64/3.8.0
   mpich/ge/gcc/64/3.3
   mvapich2/gcc/64/2.3.2
   netcdf/gcc/64/4.6.1
   netperf/2.7.0
   openblas/dynamic/0.2.20
   openmpi/gcc/64/1.10.7
   openmpi/gcc/64/4.0.3
   openmpi/gcc/64/4.0.4              (D)
```

(continues on next page)

(continued from previous page)

```
   openmpi/ics/64/4.0.3
   scalapack/openmpi/gcc/64/2.0.2
   sge/2011.11p1
   slurm/20.02.3                     (L)
   ucx/1.6.0

------------------------------ /apps/modulefiles ------------------------------
   containers/singularity/3.6.3
   containers/singularity/3.7.1                          (D)
   site/tinkercliffs-cascade_lake/easybuild/arc.arcadm
   site/tinkercliffs-cascade_lake/easybuild/setup        (D)
   site/tinkercliffs/easybuild/arc.arcadm
   site/tinkercliffs/easybuild/setup                     (L,D)
   tinkercliffs-cascade_lake/matlab/R2021a
   tinkercliffs-cascade_lake/starccm+/15.04.010
   useful_scripts                                        (L)

------------------------------- /opt/modulefiles ------------------------------
   gcc/8.1.0

---------------------------- /opt/cray/modulefiles ----------------------------
   PrgEnv-cray/1.0.6

--------------------------- /opt/cray/pe/modulefiles ---------------------------
   cce/10.0.0                  cray-mvapich2_nogpu/2.3.4
   cdt/20.05                   cray-mvapich2_nogpu_gnu/2.3.3
   cray-ccdb/3.0.5             cray-mvapich2_nogpu_gnu/2.3.4        (D)
   cray-cti/1.0.7              craype-dl-plugin-py3/mvapich/20.05.1
   cray-fftw/3.3.8.5           craype-dl-plugin-py3/openmpi/20.05.1
   cray-fftw_impi/3.3.8.5      craype/2.6.4
   cray-impi/5                 craypkg-gen/1.3.7
   cray-lgdb/3.0.10            gdb4hpc/3.0.10
   cray-libsci/20.03.1         papi/5.7.0.3
   cray-mvapich2/2.3.3         perftools-base/20.03.0
   cray-mvapich2_gnu/2.3.3     valgrind4hpc/1.0.1

-------------------- /opt/cray/pe/craype/default/modulefiles --------------------
   craype-accel-nvidia20    craype-ivybridge
   craype-accel-nvidia35    craype-mic-knl
   craype-accel-nvidia52    craype-network-infiniband (L)
   craype-accel-nvidia60    craype-network-opa
   craype-accel-nvidia70    craype-sandybridge
   craype-broadwell         craype-x86-rome           (L)
   craype-haswell           craype-x86-skylake

------------ /apps/easybuild/modules/tinkercliffs-cascade_lake/all -------------
   ANSYS/21.1
   Anaconda3/2020.07
   Anaconda3/2020.11                   (D)
   Autoconf/2.69-GCCcore-9.3.0
   Autoconf/2.69-GCCcore-10.2.0
   Autoconf/2.71-GCCcore-10.3.0        (D)
```

(continues on next page)

(continued from previous page)

```
   Automake/1.16.1-GCCcore-9.3.0
   Automake/1.16.2-GCCcore-10.2.0
   Automake/1.16.3-GCCcore-10.3.0      (D)
   Autotools/20180311-GCCcore-9.3.0
   Autotools/20200321-GCCcore-10.2.0
   Autotools/20210128-GCCcore-10.3.0   (D)
   Bison/3.3.2
   Bison/3.5.3-GCCcore-9.3.0
   Bison/3.5.3
   Bison/3.7.1-GCCcore-10.2.0
   Bison/3.7.1
   Bison/3.7.6-GCCcore-10.3.0
   Bison/3.7.6                         (D)
   DB/18.1.40-GCCcore-10.2.0
   DB/18.1.40-GCCcore-10.3.0           (D)
   EasyBuild/4.3.0
   EasyBuild/4.3.3
   EasyBuild/4.3.4
   EasyBuild/4.4.0
   EasyBuild/4.4.2                     (D)
   FFTW/3.3.8-gompi-2020a
   GCC/9.3.0
   GCC/10.3.0                          (D)
   GCCcore/9.3.0
   GCCcore/10.2.0
   GCCcore/10.3.0                      (D)
   M4/1.4.18-GCCcore-9.3.0
   M4/1.4.18-GCCcore-10.2.0
   M4/1.4.18-GCCcore-10.3.0
   M4/1.4.18                           (D)
   OpenBLAS/0.3.9-GCC-9.3.0
   OpenMPI/4.1.1-GCC-10.3.0
   OpenSSL/1.1
   PMIx/3.2.3-GCCcore-10.3.0
   Perl/5.30.2-GCCcore-9.3.0
   Perl/5.32.0-GCCcore-10.2.0
   Perl/5.32.1-GCCcore-10.3.0          (D)
   ScaLAPACK/2.1.0-gompi-2020a
   UCX/1.8.0-GCCcore-9.3.0
   UCX/1.9.0-GCCcore-10.2.0
   UCX/1.10.0-GCCcore-10.3.0           (D)
   XZ/5.2.5-GCCcore-9.3.0
   XZ/5.2.5-GCCcore-10.3.0             (D)
   binutils/2.34-GCCcore-9.3.0
   binutils/2.34
   binutils/2.35-GCCcore-10.2.0
   binutils/2.35
   binutils/2.36.1-GCCcore-10.3.0
   binutils/2.36.1                     (D)
   expat/2.2.9-GCCcore-9.3.0
   expat/2.2.9-GCCcore-10.2.0
   expat/2.2.9-GCCcore-10.3.0          (D)
```

(continues on next page)

(continued from previous page)

```
   flex/2.6.4-GCCcore-9.3.0
   flex/2.6.4-GCCcore-10.2.0
   flex/2.6.4-GCCcore-10.3.0
   flex/2.6.4                          (D)
   foss/2020a
   gettext/0.20.1
   gettext/0.21                        (D)
   gompi/2020a
   groff/1.22.4-GCCcore-10.3.0
   help2man/1.47.4
   help2man/1.47.12-GCCcore-9.3.0
   help2man/1.47.16-GCCcore-10.2.0
   help2man/1.48.3-GCCcore-10.3.0      (D)
   hwloc/2.2.0-GCCcore-9.3.0
   hwloc/2.4.1-GCCcore-10.3.0
   iccifort/2020.1.217
   iccifort/2020.4.304                 (D)
   iimpi/2020a
   iimpi/2020b                         (D)
   imkl/2020.1.217-iimpi-2020a
   imkl/2020.4.304-iimpi-2020b         (D)
   impi/2019.7.217-iccifort-2020.1.217
   impi/2019.9.304-iccifort-2020.4.304 (D)
   intel/2020a
   intel/2020b                         (D)
   libevent/2.1.12-GCCcore-10.3.0
   libfabric/1.12.1-GCCcore-10.3.0
   libpciaccess/0.16-GCCcore-9.3.0
   libpciaccess/0.16-GCCcore-10.3.0    (D)
   libreadline/8.0-GCCcore-9.3.0
   libreadline/8.0-GCCcore-10.2.0
   libreadline/8.1-GCCcore-10.3.0      (D)
   libtool/2.4.6-GCCcore-9.3.0
   libtool/2.4.6-GCCcore-10.2.0
   libtool/2.4.6-GCCcore-10.3.0        (D)
   libxml2/2.9.10-GCCcore-9.3.0
   libxml2/2.9.10-GCCcore-10.3.0       (D)
   makeinfo/6.7-GCCcore-10.3.0
   ncurses/6.1
   ncurses/6.2-GCCcore-9.3.0
   ncurses/6.2-GCCcore-10.2.0
   ncurses/6.2-GCCcore-10.3.0
   ncurses/6.2                         (D)
   numactl/2.0.13-GCCcore-9.3.0
   numactl/2.0.13-GCCcore-10.2.0
   numactl/2.0.14-GCCcore-10.3.0       (D)
   pkg-config/0.29.2-GCCcore-9.3.0
   pkg-config/0.29.2-GCCcore-10.2.0
   pkg-config/0.29.2-GCCcore-10.3.0    (D)
   xorg-macros/1.19.2-GCCcore-9.3.0
   xorg-macros/1.19.3-GCCcore-10.3.0   (D)
   zlib/1.2.11-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
   zlib/1.2.11-GCCcore-10.2.0
   zlib/1.2.11-GCCcore-10.3.0
   zlib/1.2.11                         (D)

  Where:
   D:  Default Module
   L:  Module is loaded

Module defaults are chosen based on Find First Rules due to Name/Version/Version modules␣
↪found in the module tree.
See https://lmod.readthedocs.io/en/latest/060_locating.html for details.

Use "module spider" to find all possible modules and extensions.
Use "module keyword key1 key2 ..." to search for all possible modules matching
any of the "keys".
```

4.3.6 List of Software Modules on TinkerCliffs AMD Rome Nodes

We realize this list is long, but we provide it here for users who want to peruse and/or search for what they need. For a more cleanly-formatted option, see *this table*.

```

---------------------------- /cm/local/modulefiles ----------------------------
   apps                  (L)    lua/5.3.5
   cluster-tools/8.2            module-git
   cm-cloud-copy/8.2            module-info
   cmd                          null
   cmsub                        openldap
   cray                  (L)    openmpi/mlnx/gcc/64/4.0.3rc4
   dot                          python2
   freeipmi/1.6.2               python36
   gcc/8.2.0                    shared                          (L)
   ipmitool/1.8.18

---------------------------- /usr/share/modulefiles ----------------------------
   DefaultModules (L)

---------------------------- /cm/shared/modulefiles ----------------------------
   amd-blis/aocc/64/2.1
   amd-blis/gcc/64/2.1
   amd-libflame/aocc/64/2.1
   amd-libflame/gcc/64/2.1
   aocc/aocc-compiler-2.1.0
   aocc/aocc-compiler-2.2.0          (D)
   blacs/openmpi/gcc/64/1.1patch03
   blas/gcc/64/3.8.0
   bonnie++/1.97.3
   cm-pmix3/3.1.4
   cuda-latest/blas/11.2.0
```

(continues on next page)

(continued from previous page)

```
   cuda-latest/fft/11.2.0
   cuda-latest/nsight/11.2.0
   cuda-latest/profiler/11.2.0
   cuda-latest/toolkit/11.2.0
   cuda11.2/blas/11.2.0
   cuda11.2/fft/11.2.0
   cuda11.2/nsight/11.2.0
   cuda11.2/profiler/11.2.0
   cuda11.2/toolkit/11.2.0
   default-environment
   fftw2/openmpi/gcc/64/double/2.1.5
   fftw2/openmpi/gcc/64/float/2.1.5
   fftw3/openmpi/gcc/64/3.3.8
   gdb/8.2
   globalarrays/openmpi/gcc/64/5.7
   hdf5/1.10.1
   hdf5_18/1.8.20
   hpl/2.2
   hwloc/1.11.11
   ics/2020.0
   intel-tbb-oss/ia32/2020.2
   intel-tbb-oss/intel64/2020.2
   iozone/3_482
   lapack/gcc/64/3.8.0
   mpich/ge/gcc/64/3.3
   mvapich2/gcc/64/2.3.2
   netcdf/gcc/64/4.6.1
   netperf/2.7.0
   openblas/dynamic/0.2.20
   openmpi/gcc/64/1.10.7
   openmpi/gcc/64/4.0.3
   openmpi/gcc/64/4.0.4              (D)
   openmpi/ics/64/4.0.3
   scalapack/openmpi/gcc/64/2.0.2
   sge/2011.11p1
   slurm/20.02.3                     (L)
   ucx/1.6.0

------------------------------ /apps/modulefiles -------------------------------
   containers/singularity/3.6.0
   containers/singularity/3.7.1                         (D)
   site/tinkercliffs-rome/easybuild/arc.arcadm
   site/tinkercliffs-rome/easybuild/setup               (D)
   site/tinkercliffs/easybuild/arc.arcadm
   site/tinkercliffs/easybuild/setup                    (L,D)
   tinkercliffs-rome/AccelerateCFD_CE/20210615-foss-2020a
   tinkercliffs-rome/LSDyna/R12.0.0
   tinkercliffs-rome/Nastran/2021
   tinkercliffs-rome/Patran/2021
   tinkercliffs-rome/amd-uprof/3.4.475
   tinkercliffs-rome/aspect-2.2.0/intel-2019b
   tinkercliffs-rome/aspect-2.3.0/gcc-9.3.0
```

(continues on next page)

(continued from previous page)

```
   tinkercliffs-rome/aspect-2.3.0/intel-2019b                    (D)
   tinkercliffs-rome/boost-1.58.0/intel-2019b
   tinkercliffs-rome/dealii-9.2.0/intel-2019b
   tinkercliffs-rome/dealii-9.3.1/gcc-9.3.0
   tinkercliffs-rome/dxa/1.3.6-foss-2020b
   tinkercliffs-rome/glm-0.9.8.5/intel-2019b
   tinkercliffs-rome/guppyCPU/Anaconda3-2020.07
   tinkercliffs-rome/julia/1.6.1-foss-2020b
   tinkercliffs-rome/julia/1.6.1-gomkl-2020b
   tinkercliffs-rome/julia/1.6.2-foss-2020b                      (D)
   tinkercliffs-rome/kaldi/20210429-foss-2020b
   tinkercliffs-rome/ls-dyna/R12.0.0
   tinkercliffs-rome/ls-dyna/10.2.0-intel-2019b
   tinkercliffs-rome/ls-dyna/13.0.0-intel-2019b                  (D)
   tinkercliffs-rome/ls-prepost/4.8
   tinkercliffs-rome/matlab/R2021a
   tinkercliffs-rome/metis-5.1.0/gcc-8.3.0
   tinkercliffs-rome/metis-5.1.0/gcc-9.3.0
   tinkercliffs-rome/metis-5.1.0/intel-2019b                     (D)
   tinkercliffs-rome/p4est-2.2/gcc-9.3.0
   tinkercliffs-rome/p4est-2.2/intel-2019b                       (D)
   tinkercliffs-rome/p4est/gcc-8.3.0
   tinkercliffs-rome/parmetis-4.0.3/gcc-8.3.0
   tinkercliffs-rome/parmetis-4.0.3/gcc-9.3.0
   tinkercliffs-rome/parmetis-4.0.3/intel-2019b                  (D)
   tinkercliffs-rome/starccm+/12.04.011
   tinkercliffs-rome/starccm+/15.04.010                          (D)
   tinkercliffs-rome/tpl-4.4.18/GCC-9.3.0
   tinkercliffs-rome/tpl-4.4.18/intel-2019b                      (D)
   tinkercliffs-rome/trilinos-12.18.1/gcc-8.3.0
   tinkercliffs-rome/trilinos-12.18.1/gcc-9.3.0
   tinkercliffs-rome/trilinos-12.18.1/intel-2019b                (D)
   useful_scripts                                                (L)

---------------- /apps/easybuild/modules/tinkercliffs-rome/all -----------------
   ABAQUS/2018
   ABINIT/8.10.3-intel-2019b
   ABySS/2.1.5-gompi-2020a
   ANSYS/19.5
   ANSYS/20.1
   ANSYS/20.2
   ANSYS/21.1
   ANSYS/21.2                                                     (D)
   APR-util/1.6.1-GCCcore-10.2.0
   APR/1.7.0-GCCcore-10.2.0
   ATK/2.36.0-GCCcore-9.3.0
   ATK/2.36.0-GCCcore-10.2.0                                      (D)
   AUGUSTUS/3.4.0-foss-2020b
   Anaconda3/2020.07
   Anaconda3/2020.11                                              (D)
   AtomPAW/4.1.0.5-intel-2019b
   Autoconf/2.69-GCCcore-8.3.0
```

(continues on next page)

(continued from previous page)

```
   Autoconf/2.69-GCCcore-9.3.0
   Autoconf/2.69-GCCcore-10.2.0
   Autoconf/2.71-GCCcore-10.3.0                            (D)
   Automake/1.16.1-GCCcore-8.3.0
   Automake/1.16.1-GCCcore-9.3.0
   Automake/1.16.2-GCCcore-10.2.0
   Automake/1.16.3-GCCcore-10.3.0                          (D)
   Autotools/20180311-GCCcore-8.3.0
   Autotools/20180311-GCCcore-9.3.0
   Autotools/20200321-GCCcore-10.2.0
   Autotools/20210128-GCCcore-10.3.0                       (D)
   BCFtools/1.10.2-GCC-9.3.0
   BCFtools/1.11-GCC-10.2.0                                (D)
   BEDTools/2.29.2-GCC-9.3.0
   BLAST+/2.10.1-gompi-2020a
   BLAST+/2.11.0-gompi-2020b                               (D)
   BUSCO/5.0.0-foss-2020b
   BamTools/2.5.1-GCC-9.3.0
   BamTools/2.5.1-GCC-10.2.0                               (D)
   Bazel/3.7.2-GCCcore-10.2.0
   Biopython/1.75-intel-2019b-Python-3.7.4
   Biopython/1.78-foss-2020a-Python-3.8.2
   Biopython/1.78-foss-2020b                               (D)
   Bison/3.0.4
   Bison/3.0.5
   Bison/3.3.2-GCCcore-8.3.0
   Bison/3.3.2
   Bison/3.5.3-GCCcore-9.3.0
   Bison/3.5.3-intel-2019b
   Bison/3.5.3
   Bison/3.7.1-GCCcore-10.2.0
   Bison/3.7.1
   Bison/3.7.6-GCCcore-10.3.0
   Bison/3.7.6
   Bison/3.7.91                                            (D)
   Boost/1.71.0-iimpi-2019b
   Boost/1.72.0-gompi-2020a
   Boost/1.74.0-GCC-10.2.0                                 (D)
   Bowtie2/2.4.1-GCC-9.3.0
   CGAL/4.14.3-gompi-2020a-Python-3.8.2
   CMake/3.15.3-GCCcore-8.3.0
   CMake/3.16.4-GCCcore-9.3.0
   CMake/3.16.4-intel-2019b
   CMake/3.18.4-GCCcore-10.2.0
   CMake/3.20.1-GCCcore-10.3.0                             (D)
   CP2K/6.1-foss-2020a
   DB/18.1.40-GCCcore-10.2.0
   DB/18.1.40-GCCcore-10.3.0                               (D)
   DBus/1.13.12-GCCcore-8.3.0
   DBus/1.13.12-GCCcore-9.3.0
   DBus/1.13.18-GCCcore-10.2.0                             (D)
   Dalton/2020-iomkl-2019b-nompi
```

(continues on next page)

(continued from previous page)

```
   Dalton/2020-iomkl-2019b                                    (D)
   DendroPy/4.5.2-GCCcore-10.2.0
   Doxygen/1.8.16-GCCcore-8.3.0
   Doxygen/1.8.17-GCCcore-9.3.0
   Doxygen/1.8.20-GCCcore-10.2.0
   Doxygen/1.9.1-GCCcore-10.3.0                               (D)
   ELPA/2019.11.001-intel-2019b
   EasyBuild/4.2.2
   EasyBuild/4.3.2
   EasyBuild/4.3.3
   EasyBuild/4.3.4
   EasyBuild/4.4.0
   EasyBuild/4.4.2                                            (D)
   Eigen/3.3.7-GCCcore-9.3.0
   Eigen/3.3.7
   Eigen/3.3.8-GCCcore-10.2.0
   Eigen/3.3.9-GCCcore-10.3.0                                 (D)
   FDS/6.7.1-intel-2019b
   FDS/6.7.4-intel-2019b
   FDS/6.7.5-intel-2019b                                      (D)
   FFTW/3.3.8-gompi-2020a
   FFTW/3.3.8-gompi-2020b
   FFTW/3.3.8-intel-2019b
   FFTW/3.3.9-gompi-2021a                                     (D)
   FFmpeg/4.2.1-GCCcore-8.3.0
   FFmpeg/4.2.2-GCCcore-9.3.0
   FFmpeg/4.3.1-GCCcore-10.2.0                                (D)
   FLAC/1.3.3-GCCcore-10.2.0
   FLAC/1.3.3-GCCcore-10.3.0                                  (D)
   FlexiBLAS/3.0.4-GCC-10.3.0
   FriBidi/1.0.5-GCCcore-8.3.0
   FriBidi/1.0.9-GCCcore-9.3.0
   FriBidi/1.0.10-GCCcore-10.2.0                              (D)
   GCC/8.3.0
   GCC/9.3.0
   GCC/10.2.0
   GCC/10.3.0                                                 (D)
   GCCcore/8.2.0
   GCCcore/8.3.0
   GCCcore/9.3.0
   GCCcore/10.2.0
   GCCcore/10.3.0                                             (D)
   GDAL/3.0.4-foss-2020a-Python-3.8.2
   GDAL/3.3.0-foss-2021a                                      (D)
   GEOS/3.8.1-GCC-9.3.0-Python-3.8.2
   GEOS/3.9.1-GCC-10.3.0                                      (D)
   GLPK/4.65-GCCcore-8.3.0
   GLPK/4.65-GCCcore-9.3.0
   GLPK/4.65-GCCcore-10.2.0
   GLPK/5.0-GCCcore-10.3.0                                    (D)
   GLib/2.62.0-GCCcore-8.3.0
   GLib/2.64.1-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
   GLib/2.66.1-GCCcore-10.2.0
   GLib/2.68.2-GCCcore-10.3.0                              (D)
   GMP/6.1.2-GCCcore-8.3.0
   GMP/6.2.0-GCCcore-9.3.0
   GMP/6.2.0-GCCcore-10.2.0
   GMP/6.2.0-intel-2019b
   GMP/6.2.1-GCCcore-10.3.0                                (D)
   GMT/5.4.5-foss-2020a
   GObject-Introspection/1.63.1-GCCcore-8.3.0-Python-3.7.4
   GObject-Introspection/1.64.0-GCCcore-9.3.0-Python-3.8.2
   GObject-Introspection/1.66.1-GCCcore-10.2.0             (D)
   GROMACS/2020.1-foss-2020a-Python-3.8.2
   GROMACS/2020.3-foss-2020a-Python-3.8.2                  (D)
   GSL/2.6-GCC-9.3.0
   GSL/2.6-GCC-10.2.0
   GSL/2.6-iccifort-2019.5.281
   GSL/2.7-GCC-10.3.0                                      (D)
   GTK+/3.24.17-GCCcore-9.3.0
   GTK+/3.24.23-GCCcore-10.2.0                             (D)
   Gdk-Pixbuf/2.40.0-GCCcore-9.3.0
   Gdk-Pixbuf/2.40.0-GCCcore-10.2.0                        (D)
   Ghostscript/9.52-GCCcore-9.3.0
   Ghostscript/9.52-intel-2019b
   Ghostscript/9.53.3-GCCcore-10.2.0
   Ghostscript/9.54.0-GCCcore-10.3.0                       (D)
   GlobalArrays/5.7.2-intel-2019b
   GlobalArrays/5.7.2-iomkl-2019b                          (D)
   Go/1.14
   Guile/1.8.8-GCCcore-9.3.0
   HDF/4.2.15-GCCcore-10.3.0
   HDF5/1.10.2-intel-2019b
   HDF5/1.10.2-iomkl-2019b
   HDF5/1.10.5-iimpi-2019b
   HDF5/1.10.6-gompi-2020a
   HDF5/1.10.6-intel-2019b
   HDF5/1.10.7-gompi-2020b
   HDF5/1.10.7-gompi-2021a
   HDF5/1.12.0-gompi-2020a                                 (D)
   HMMER/3.3.2-gompi-2020b
   HMMER2/2.3.2-GCC-8.3.0
   HPL/2.3-foss-2020a
   HPL/2.3-intel-2019b                                     (D)
   HTSlib/1.10.2-GCC-9.3.0
   HTSlib/1.11-GCC-10.2.0                                  (D)
   HarfBuzz/2.6.4-GCCcore-8.3.0
   HarfBuzz/2.6.4-GCCcore-9.3.0
   HarfBuzz/2.6.7-GCCcore-10.2.0                           (D)
   Hypre/2.18.2-intel-2019b
   ICU/64.2-GCCcore-8.3.0
   ICU/66.1-GCCcore-9.3.0
   ICU/66.1-intel-2019b
   ICU/67.1-GCCcore-10.2.0
```

(continues on next page)

(continued from previous page)

```
   ICU/69.1-GCCcore-10.3.0                                     (D)
   ImageMagick/7.0.10-1-GCCcore-9.3.0
   ImageMagick/7.0.10-1-intel-2019b
   ImageMagick/7.0.10-35-GCCcore-10.2.0
   ImageMagick/7.0.11-14-GCCcore-10.3.0                        (D)
   JasPer/2.0.14-GCCcore-8.3.0
   JasPer/2.0.14-GCCcore-9.3.0
   JasPer/2.0.24-GCCcore-10.2.0
   JasPer/2.0.28-GCCcore-10.3.0                                (D)
   Java/11.0.2                                                 (11)
   Jellyfish/2.3.0-GCC-9.3.0
   JsonCpp/1.9.4-GCCcore-10.2.0
   Julia/1.4.2-linux-x86_64
   Julia/1.5.1-linux-x86_64                                    (D)
   LAME/3.100-GCCcore-8.3.0
   LAME/3.100-GCCcore-9.3.0
   LAME/3.100-GCCcore-10.2.0                                   (D)
   LAMMPS/3Mar2020-foss-2020a-Python-3.8.2-kokkos
   LLVM/9.0.0-GCCcore-8.3.0
   LLVM/9.0.1-GCCcore-9.3.0
   LLVM/11.0.0-GCCcore-10.2.0
   LLVM/11.1.0-GCCcore-10.3.0                                  (D)
   LMDB/0.9.24-GCCcore-9.3.0
   LMDB/0.9.24-GCCcore-10.2.0                                  (D)
   LibTIFF/4.0.10-GCCcore-8.3.0
   LibTIFF/4.1.0-GCCcore-8.3.0
   LibTIFF/4.1.0-GCCcore-9.3.0
   LibTIFF/4.1.0-GCCcore-10.2.0
   LibTIFF/4.2.0-GCCcore-10.3.0                                (D)
   Libint/1.1.6-foss-2020a
   Libint/2.6.0-GCC-10.2.0-lmax-6-cp2k                         (D)
   LittleCMS/2.9-GCCcore-8.3.0
   LittleCMS/2.9-GCCcore-9.3.0
   LittleCMS/2.11-GCCcore-10.2.0
   LittleCMS/2.12-GCCcore-10.3.0                               (D)
   Lua/5.1.5-GCCcore-8.3.0
   M4/1.4.17
   M4/1.4.18-GCCcore-8.2.0
   M4/1.4.18-GCCcore-8.3.0
   M4/1.4.18-GCCcore-9.3.0
   M4/1.4.18-GCCcore-10.2.0
   M4/1.4.18-GCCcore-10.3.0
   M4/1.4.18                                                   (D)
   MATLAB/2019b
   METIS/5.1.0-GCCcore-8.3.0
   METIS/5.1.0-GCCcore-9.3.0
   METIS/5.1.0-GCCcore-10.2.0                                  (D)
   MPFR/4.0.2-GCCcore-8.3.0
   MPFR/4.0.2-GCCcore-9.3.0
   MPFR/4.1.0-GCCcore-10.2.0                                   (D)
   MUMPS/5.2.1-foss-2020a-metis
   MUMPS/5.2.1-intel-2019b-metis                               (D)
```

(continues on next page)

(continued from previous page)

```
   Mako/1.1.0-GCCcore-8.3.0
   Mako/1.1.2-GCCcore-9.3.0
   Mako/1.1.3-GCCcore-10.2.0
   Mako/1.1.4-GCCcore-10.3.0                               (D)
   MariaDB-connector-c/3.1.7-GCCcore-9.3.0
   MariaDB-connector-c/3.1.7-intel-2019b                   (D)
   Mathematica/12.0.0
   Mesa/19.1.7-GCCcore-8.3.0
   Mesa/19.2.1-GCCcore-8.3.0
   Mesa/20.0.2-GCCcore-9.3.0
   Mesa/20.2.1-GCCcore-10.2.0
   Mesa/21.1.1-GCCcore-10.3.0                              (D)
   Meson/0.51.2-GCCcore-8.3.0-Python-3.7.4
   Meson/0.53.2-GCCcore-9.3.0-Python-3.8.2
   Meson/0.53.2-intel-2019b-Python-3.7.4
   Meson/0.55.1-GCCcore-9.3.0-Python-3.8.2
   Meson/0.55.3-GCCcore-10.2.0
   Meson/0.58.0-GCCcore-10.3.0                             (D)
   MetaEuk/4-GCC-10.2.0
   Miniconda3/4.7.10
   NAMD/2.13-foss-2020a-mpi
   NASM/2.14.02-GCCcore-8.3.0
   NASM/2.14.02-GCCcore-9.3.0
   NASM/2.15.05-GCCcore-10.2.0
   NASM/2.15.05-GCCcore-10.3.0                             (D)
   NLopt/2.6.1-GCCcore-8.3.0
   NLopt/2.6.1-GCCcore-9.3.0
   NLopt/2.6.2-GCCcore-10.2.0
   NLopt/2.7.0-GCCcore-10.3.0                              (D)
   NSPR/4.21-GCCcore-8.3.0
   NSPR/4.25-GCCcore-9.3.0                                 (D)
   NSS/3.45-GCCcore-8.3.0
   NSS/3.51-GCCcore-9.3.0                                  (D)
   NVHPC/20.7
   NVHPC/21.2                                              (D)
   Ninja/1.9.0-GCCcore-8.3.0
   Ninja/1.10.0-GCCcore-9.3.0
   Ninja/1.10.0-intel-2019b
   Ninja/1.10.1-GCCcore-10.2.0
   Ninja/1.10.2-GCCcore-10.3.0                             (D)
   OpenBLAS/0.3.9-GCC-9.3.0
   OpenBLAS/0.3.12-GCC-10.2.0
   OpenBLAS/0.3.15-GCC-10.3.0                              (D)
   OpenFOAM/v2006-foss-2020a
   OpenMM/7.4.1-intel-2019b-Python-3.7.4
   OpenMPI/3.1.4-iccifort-2019.5.281
   OpenMPI/4.0.3-GCC-9.3.0
   OpenMPI/4.0.3-iccifort-2019.5.281
   OpenMPI/4.0.5-GCC-10.2.0
   OpenMPI/4.1.1-GCC-10.3.0                                (D)
   OpenMolcas/18.09-intel-2019b-Python-3.7.4
   OpenMolcas/19.11-intel-2019b-Python-3.7.4               (D)
```

(continues on next page)

(continued from previous page)

```
   OpenSSL/1.1
   OpenSSL/1.1.1e-GCCcore-9.3.0
   OpenSSL/1.1.1e-intel-2019b                                (D)
   PCRE/8.43-GCCcore-8.3.0
   PCRE/8.44-GCCcore-9.3.0
   PCRE/8.44-GCCcore-10.2.0
   PCRE/8.44-GCCcore-10.3.0                                  (D)
   PCRE2/10.33-GCCcore-8.3.0
   PCRE2/10.34-GCCcore-9.3.0
   PCRE2/10.34-intel-2019b
   PCRE2/10.35-GCCcore-10.2.0
   PCRE2/10.36-GCCcore-10.3.0                                (D)
   PETSc/3.12.4-intel-2019b
   PLUMED/2.5.1-foss-2020a
   PLUMED/2.6.0-foss-2020a-Python-3.8.2                      (D)
   PMIx/3.1.5-GCCcore-8.3.0
   PMIx/3.1.5-GCCcore-10.2.0
   PMIx/3.2.3-GCCcore-10.3.0                                 (D)
   PROJ/7.0.0-GCCcore-9.3.0
   PROJ/8.0.1-GCCcore-10.3.0                                 (D)
   Pango/1.44.7-GCCcore-8.3.0
   Pango/1.44.7-GCCcore-9.3.0
   Pango/1.47.0-GCCcore-10.2.0                               (D)
   ParaView/5.8.0-foss-2020a-Python-3.8.2-mpi
   Perl/5.30.0-GCCcore-8.3.0
   Perl/5.30.2-GCCcore-9.3.0
   Perl/5.32.0-GCCcore-10.2.0
   Perl/5.32.1-GCCcore-10.3.0                                (D)
   Pillow/6.2.1-GCCcore-8.3.0
   Pillow/7.0.0-GCCcore-9.3.0-Python-3.8.2
   Pillow/8.0.1-GCCcore-10.2.0                               (D)
   PyCharm/2019.3.1
   PyCharm/2021.1.1                                          (D)
   PyTorch/1.4.0-foss-2020a-Python-3.8.2
   PyTorch/1.6.0-foss-2020a-Python-3.8.2
   PyTorch/1.6.0-gomkl-2020a-Python-3.8.2
   PyTorch/1.7.1-foss-2020b                                  (D)
   PyYAML/5.1.2-GCCcore-8.3.0
   PyYAML/5.3-GCCcore-9.3.0
   PyYAML/5.3.1-GCCcore-10.2.0                               (D)
   Pysam/0.16.0.1-GCC-9.3.0
   Python/2.7.16-GCCcore-8.3.0
   Python/2.7.18-GCCcore-9.3.0
   Python/2.7.18-GCCcore-10.2.0
   Python/3.7.4-GCCcore-8.3.0
   Python/3.8.2-GCCcore-9.3.0
   Python/3.8.6-GCCcore-10.2.0
   Python/3.9.5-GCCcore-10.3.0-bare
   Python/3.9.5-GCCcore-10.3.0                               (D)
   QIIME2/2020.6
   Qt5/5.13.1-GCCcore-8.3.0
   Qt5/5.14.1-GCCcore-9.3.0                                  (D)
```

(continues on next page)

(continued from previous page)

```
   Qualimap/2.2.1-foss-2020b-R-4.0.3
   QuantumESPRESSO/6.5-intel-2019b
   R-bundle-Bioconductor/3.12-foss-2020b-R-4.0.3
   R/4.0.2-foss-2020a
   R/4.0.3-foss-2020b
   R/4.1.0-foss-2021a                                    (D)
   Ruby/2.7.2-GCCcore-9.3.0
   Rust/1.52.1-GCCcore-10.3.0
   SAMtools/1.11-GCC-10.2.0
   SCOTCH/6.0.9-gompi-2020a
   SCOTCH/6.0.9-iimpi-2019b                              (D)
   SCons/4.0.1-GCCcore-10.2.0
   SEPP/4.4.0-foss-2020b
   SLEPc/3.12.2-intel-2019b
   SQLite/3.29.0-GCCcore-8.3.0
   SQLite/3.31.1-GCCcore-9.3.0
   SQLite/3.31.1-intel-2019b
   SQLite/3.33.0-GCCcore-10.2.0
   SQLite/3.35.4-GCCcore-10.3.0                          (D)
   SWIG/4.0.1-GCCcore-8.3.0
   SWIG/4.0.1-GCCcore-9.3.0                              (D)
   ScaFaCoS/1.0.1-foss-2020a
   ScaLAPACK/2.1.0-gompi-2020a
   ScaLAPACK/2.1.0-gompi-2020b
   ScaLAPACK/2.1.0-gompi-2021a-fb                        (D)
   SciPy-bundle/2019.10-intel-2019b-Python-3.7.4
   SciPy-bundle/2020.03-foss-2020a-Python-3.8.2
   SciPy-bundle/2020.03-gomkl-2020a-Python-3.8.2
   SciPy-bundle/2020.11-foss-2020b
   SciPy-bundle/2021.05-foss-2021a                       (D)
   Serf/1.3.9-GCCcore-10.2.0
   SoX/14.4.2-GCC-10.2.0
   SpaceRanger/1.2.2-GCC-9.3.0
   Subversion/1.14.0-GCCcore-10.2.0
   SuiteSparse/5.6.0-intel-2019b-METIS-5.1.0
   SuiteSparse/5.8.1-foss-2020b-METIS-5.1.0              (D)
   Szip/2.1.1-GCCcore-8.3.0
   Szip/2.1.1-GCCcore-9.3.0
   Szip/2.1.1-GCCcore-10.2.0
   Szip/2.1.1-GCCcore-10.3.0                             (D)
   TINKER/8.8.1-foss-2020a
   Tcl/8.6.9-GCCcore-8.3.0
   Tcl/8.6.10-GCCcore-9.3.0
   Tcl/8.6.10-GCCcore-10.2.0
   Tcl/8.6.10-intel-2019b
   Tcl/8.6.11-GCCcore-10.3.0                             (D)
   TensorFlow/2.4.1-foss-2020b
   Tk/8.6.10-GCCcore-9.3.0
   Tk/8.6.10-GCCcore-10.2.0
   Tk/8.6.10-intel-2019b
   Tk/8.6.11-GCCcore-10.3.0                              (D)
   Tkinter/3.8.2-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
   Tkinter/3.8.6-GCCcore-10.2.0                            (D)
   TopHat/2.1.2-iimpi-2019b
   UCX/1.8.0-GCCcore-8.3.0
   UCX/1.8.0-GCCcore-9.3.0
   UCX/1.9.0-GCCcore-10.2.0
   UCX/1.10.0-GCCcore-10.3.0                               (D)
   UDUNITS/2.2.26-GCCcore-8.3.0
   UDUNITS/2.2.26-GCCcore-9.3.0
   UDUNITS/2.2.26-GCCcore-10.2.0
   UDUNITS/2.2.28-GCCcore-10.3.0                           (D)
   UnZip/6.0-GCCcore-10.2.0
   UnZip/6.0-GCCcore-10.3.0                                (D)
   VASP/5.4.4-intel-2019b
   VTK/8.2.0-foss-2020a-Python-3.8.2
   Valgrind/3.16.1-gompi-2020a
   Valgrind/3.16.1-iimpi-2019b                             (D)
   VirtualGL/2.6.2-GCCcore-9.3.0
   Voro++/0.4.6-GCCcore-9.3.0
   WPS/4.2-foss-2020b-dmpar
   WRF/4.1.3-intel-2019b-dmpar
   WRF/4.2.2-foss-2020b-dm+sm
   WRF/4.2.2-foss-2020b-dmpar                              (D)
   Wannier90/2.0.1.1-intel-2019b-abinit
   X11/20190717-GCCcore-8.3.0
   X11/20200222-GCCcore-9.3.0
   X11/20200222-intel-2019b
   X11/20201008-GCCcore-10.2.0
   X11/20210518-GCCcore-10.3.0                             (D)
   XML-LibXML/2.0205-GCCcore-9.3.0
   XZ/5.2.4-GCCcore-8.3.0
   XZ/5.2.5-GCCcore-8.3.0
   XZ/5.2.5-GCCcore-9.3.0
   XZ/5.2.5-GCCcore-10.2.0
   XZ/5.2.5-GCCcore-10.3.0
   XZ/5.2.5-intel-2019b                                    (D)
   Xvfb/1.20.9-GCCcore-10.2.0
   Xvfb/1.20.11-GCCcore-10.3.0                             (D)
   Yasm/1.3.0-GCCcore-8.3.0
   Yasm/1.3.0-GCCcore-9.3.0
   Yasm/1.3.0-GCCcore-10.2.0                               (D)
   Zip/3.0-GCCcore-10.2.0
   archspec/0.1.0-GCCcore-9.3.0-Python-3.8.2
   at-spi2-atk/2.34.2-GCCcore-9.3.0
   at-spi2-atk/2.38.0-GCCcore-10.2.0                       (D)
   at-spi2-core/2.36.0-GCCcore-9.3.0
   at-spi2-core/2.38.0-GCCcore-10.2.0                      (D)
   bcl2fastq2/2.20.0-GCC-9.3.0
   binutils/2.30
   binutils/2.31.1
   binutils/2.32-GCCcore-8.3.0
   binutils/2.32
   binutils/2.34-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
   binutils/2.34-intel-2019b
   binutils/2.34
   binutils/2.35-GCCcore-10.2.0
   binutils/2.35
   binutils/2.36.1-GCCcore-10.3.0
   binutils/2.36.1                                      (D)
   bokeh/2.0.2-foss-2020a-Python-3.8.2
   bokeh/2.2.3-foss-2020b-Python-3.8.6                  (D)
   bzip2/1.0.8-GCCcore-8.3.0
   bzip2/1.0.8-GCCcore-9.3.0
   bzip2/1.0.8-GCCcore-10.2.0
   bzip2/1.0.8-GCCcore-10.3.0                           (D)
   cURL/7.66.0-GCCcore-8.3.0
   cURL/7.69.1-GCCcore-9.3.0
   cURL/7.72.0-GCCcore-10.2.0
   cURL/7.76.0-GCCcore-10.3.0                           (D)
   cairo/1.16.0-GCCcore-8.3.0
   cairo/1.16.0-GCCcore-9.3.0
   cairo/1.16.0-GCCcore-10.2.0
   cairo/1.16.0-GCCcore-10.3.0                          (D)
   canu/1.9-GCCcore-8.3.0-Java-11
   dask/2.18.1-foss-2020a-Python-3.8.2
   dask/2021.2.0-foss-2020b-Python-3.8.6                (D)
   double-conversion/3.1.4-GCCcore-8.3.0
   double-conversion/3.1.5-GCCcore-9.3.0
   double-conversion/3.1.5-GCCcore-10.2.0               (D)
   ea-utils/1.04.807-intel-2019b
   expat/2.2.7-GCCcore-8.3.0
   expat/2.2.9-GCCcore-9.3.0
   expat/2.2.9-GCCcore-10.2.0
   expat/2.2.9-GCCcore-10.3.0
   expat/2.2.9-intel-2019b                              (D)
   flatbuffers-python/1.12-GCCcore-10.2.0
   flatbuffers/1.12.0-GCCcore-10.2.0
   flex/2.6.4-GCCcore-8.3.0
   flex/2.6.4-GCCcore-9.3.0
   flex/2.6.4-GCCcore-10.2.0
   flex/2.6.4-GCCcore-10.3.0
   flex/2.6.4                                           (D)
   fontconfig/2.13.1-GCCcore-8.3.0
   fontconfig/2.13.92-GCCcore-9.3.0
   fontconfig/2.13.92-GCCcore-10.2.0
   fontconfig/2.13.92-intel-2019b
   fontconfig/2.13.93-GCCcore-10.3.0                    (D)
   foss/2020a
   foss/2020b
   foss/2021a                                           (D)
   freetype/2.10.1-GCCcore-8.3.0
   freetype/2.10.1-GCCcore-9.3.0
   freetype/2.10.3-GCCcore-10.2.0
   freetype/2.10.4-GCCcore-10.3.0                       (D)
   gaussian/09.e01
```

(continues on next page)

(continued from previous page)

```
   gc/7.6.12-GCCcore-9.3.0
   gettext/0.19.8.1
   gettext/0.20.1-GCCcore-8.3.0
   gettext/0.20.1-GCCcore-9.3.0
   gettext/0.20.1
   gettext/0.21-GCCcore-10.2.0
   gettext/0.21-GCCcore-10.3.0
   gettext/0.21                                          (D)
   gflags/2.2.2-GCCcore-9.3.0
   giflib/5.2.1-GCCcore-10.2.0
   git/2.28.0-GCCcore-10.2.0-nodocs
   glog/0.4.0-GCCcore-9.3.0
   gmsh/4.5.6-intel-2019b-Python-2.7.16
   gnuplot/5.2.8-GCCcore-8.3.0
   gomkl/2020a
   gomkl/2020b
   gomkl/2021a                                           (D)
   gompi/2020a
   gompi/2020b
   gompi/2021a                                           (D)
   gperf/3.1-GCCcore-8.3.0
   gperf/3.1-GCCcore-9.3.0
   gperf/3.1-GCCcore-10.2.0
   gperf/3.1-GCCcore-10.3.0                              (D)
   gperftools/2.8-GCCcore-10.2.0
   groff/1.22.4-GCCcore-10.3.0
   gzip/1.10-GCCcore-9.3.0
   gzip/1.10-GCCcore-10.2.0
   gzip/1.10-GCCcore-10.3.0                              (D)
   h5py/2.10.0-foss-2020a-Python-3.8.2
   help2man/1.47.4
   help2man/1.47.7-GCCcore-8.2.0
   help2man/1.47.8-GCCcore-8.3.0
   help2man/1.47.12-GCCcore-9.3.0
   help2man/1.47.16-GCCcore-10.2.0
   help2man/1.48.3-GCCcore-10.3.0                        (D)
   hwloc/1.11.12-GCCcore-8.3.0
   hwloc/2.2.0-GCCcore-8.3.0
   hwloc/2.2.0-GCCcore-9.3.0
   hwloc/2.2.0-GCCcore-10.2.0
   hwloc/2.4.1-GCCcore-10.3.0
   hypothesis/4.44.2-GCCcore-8.3.0-Python-3.7.4
   hypothesis/5.6.0-GCCcore-9.3.0-Python-3.8.2
   hypothesis/5.41.2-GCCcore-10.2.0
   hypothesis/5.41.5-GCCcore-10.2.0
   hypothesis/6.13.1-GCCcore-10.3.0                      (D)
   iccifort/2019.5.281
   iimpi/2019b
   iimpi/2021a                                           (D)
   imkl/2019.5.281-gompi-2020a
   imkl/2019.5.281-iimpi-2019b
   imkl/2019.5.281-iompi-2019b
```

(continues on next page)

(continued from previous page)

```
   imkl/2020.4.304-gompi-2020b
   imkl/2021.2.0-gompi-2021a
   imkl/2021.2.0-iimpi-2021a                            (D)
   impi/2018.5.288-iccifort-2019.5.281
   impi/2021.2.0-intel-compilers-2021.2.0               (D)
   intel-compilers/2021.2.0
   intel/2019b
   intel/2021a                                          (D)
   intltool/0.51.0-GCCcore-8.3.0
   intltool/0.51.0-GCCcore-9.3.0
   intltool/0.51.0-GCCcore-10.2.0
   intltool/0.51.0-GCCcore-10.3.0                       (D)
   iomkl/2019b
   iompi/2019b
   kim-api/2.1.3-foss-2020a
   libGLU/9.0.1-GCCcore-8.3.0
   libGLU/9.0.1-GCCcore-9.3.0
   libGLU/9.0.1-GCCcore-10.2.0
   libGLU/9.0.1-GCCcore-10.3.0                          (D)
   libarchive/3.4.3-GCCcore-10.2.0
   libarchive/3.5.1-GCCcore-10.3.0                      (D)
   libcerf/1.13-GCCcore-8.3.0
   libdrm/2.4.99-GCCcore-8.3.0
   libdrm/2.4.100-GCCcore-9.3.0
   libdrm/2.4.102-GCCcore-10.2.0
   libdrm/2.4.106-GCCcore-10.3.0                        (D)
   libepoxy/1.5.4-GCCcore-9.3.0
   libepoxy/1.5.4-GCCcore-10.2.0                        (D)
   libevent/2.1.11-GCCcore-8.3.0
   libevent/2.1.11-GCCcore-9.3.0
   libevent/2.1.12-GCCcore-10.2.0
   libevent/2.1.12-GCCcore-10.3.0                       (D)
   libfabric/1.11.0-GCCcore-8.3.0
   libfabric/1.11.0-GCCcore-10.2.0
   libfabric/1.12.1-GCCcore-10.3.0                      (D)
   libffi/3.2.1-GCCcore-8.3.0
   libffi/3.3-GCCcore-9.3.0
   libffi/3.3-GCCcore-10.2.0
   libffi/3.3-GCCcore-10.3.0
   libffi/3.3-intel-2019b                               (D)
   libgd/2.2.5-GCCcore-8.3.0
   libgeotiff/1.5.1-GCCcore-9.3.0
   libgeotiff/1.6.0-GCCcore-10.3.0                      (D)
   libgit2/1.1.0-GCCcore-10.3.0
   libglvnd/1.2.0-GCCcore-8.3.0
   libglvnd/1.2.0-GCCcore-9.3.0
   libglvnd/1.3.2-GCCcore-10.2.0
   libglvnd/1.3.3-GCCcore-10.3.0                        (D)
   libiconv/1.16-GCCcore-10.2.0
   libiconv/1.16-GCCcore-10.3.0                         (D)
   libjpeg-turbo/2.0.3-GCCcore-8.3.0
   libjpeg-turbo/2.0.4-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
   libjpeg-turbo/2.0.4-intel-2019b
   libjpeg-turbo/2.0.5-GCCcore-10.2.0
   libjpeg-turbo/2.0.6-GCCcore-10.3.0                     (D)
   libmatheval/1.1.11-GCCcore-9.3.0
   libogg/1.3.4-GCCcore-10.2.0
   libogg/1.3.4-GCCcore-10.3.0                            (D)
   libpciaccess/0.14-GCCcore-8.3.0
   libpciaccess/0.16-GCCcore-8.3.0
   libpciaccess/0.16-GCCcore-9.3.0
   libpciaccess/0.16-GCCcore-10.2.0
   libpciaccess/0.16-GCCcore-10.3.0
   libpciaccess/0.16-intel-2019b                          (D)
   libpng/1.6.37-GCCcore-8.3.0
   libpng/1.6.37-GCCcore-9.3.0
   libpng/1.6.37-GCCcore-10.2.0
   libpng/1.6.37-GCCcore-10.3.0                           (D)
   libreadline/8.0-GCCcore-8.3.0
   libreadline/8.0-GCCcore-9.3.0
   libreadline/8.0-GCCcore-10.2.0
   libreadline/8.1-GCCcore-10.3.0                         (D)
   libsndfile/1.0.28-GCCcore-8.3.0
   libsndfile/1.0.28-GCCcore-9.3.0
   libsndfile/1.0.28-GCCcore-10.2.0
   libsndfile/1.0.31-GCCcore-10.3.0                       (D)
   libtirpc/1.3.2-GCCcore-10.3.0
   libtool/2.4.6-GCCcore-8.3.0
   libtool/2.4.6-GCCcore-9.3.0
   libtool/2.4.6-GCCcore-10.2.0
   libtool/2.4.6-GCCcore-10.3.0                           (D)
   libunistring/0.9.10-GCCcore-9.3.0
   libunwind/1.3.1-GCCcore-8.3.0
   libunwind/1.3.1-GCCcore-9.3.0
   libunwind/1.4.0-GCCcore-10.2.0
   libunwind/1.4.0-GCCcore-10.3.0                         (D)
   libvorbis/1.3.7-GCCcore-10.2.0
   libvorbis/1.3.7-GCCcore-10.3.0                         (D)
   libxc/3.0.1-intel-2019b
   libxc/4.2.3-intel-2019b
   libxc/4.3.4-GCC-9.3.0
   libxc/4.3.4-iccifort-2019.5.281                        (D)
   libxml2/2.9.9-GCCcore-8.3.0
   libxml2/2.9.10-GCCcore-8.3.0
   libxml2/2.9.10-GCCcore-9.3.0
   libxml2/2.9.10-GCCcore-10.2.0
   libxml2/2.9.10-GCCcore-10.3.0
   libxml2/2.9.10-intel-2019b                             (D)
   libxsmm/1.10-GCC-9.3.0
   libyaml/0.2.2-GCCcore-8.3.0
   libyaml/0.2.2-GCCcore-9.3.0
   libyaml/0.2.5-GCCcore-10.2.0                           (D)
   lpsolve/5.5.2.11-GCC-10.2.0
   lz4/1.9.2-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
   lz4/1.9.2-GCCcore-10.2.0
   lz4/1.9.3-GCCcore-10.3.0                                     (D)
   makeinfo/6.7-GCCcore-10.3.0
   matplotlib/3.2.1-foss-2020a-Python-3.8.2
   matplotlib/3.3.3-foss-2020b                                  (D)
   minimap2/2.17-GCCcore-9.3.0
   molmod/1.4.5-foss-2020a-Python-3.8.2
   mpi4py/3.0.2-gompi-2020a-timed-pingpong
   mpi4py/3.0.2-iimpi-2019b-timed-pingpong
   mpi4py/3.1.1-gompi-2020b-timed-pingpong                      (D)
   nanopolish/0.13.2-foss-2020a-Python-3.8.2
   ncdf4/1.17-foss-2020b-R-4.0.3
   ncurses/6.0
   ncurses/6.1-GCCcore-8.3.0
   ncurses/6.1
   ncurses/6.2-GCCcore-9.3.0
   ncurses/6.2-GCCcore-10.2.0
   ncurses/6.2-GCCcore-10.3.0
   ncurses/6.2-intel-2019b
   ncurses/6.2                                                  (D)
   netCDF-Fortran/4.4.4-intel-2019b
   netCDF-Fortran/4.5.2-iimpi-2019b
   netCDF-Fortran/4.5.3-gompi-2020b                             (D)
   netCDF/4.6.1-intel-2019b
   netCDF/4.7.1-iimpi-2019b
   netCDF/4.7.4-gompi-2020a
   netCDF/4.7.4-gompi-2020b
   netCDF/4.8.0-gompi-2021a                                     (D)
   nettle/3.5.1-GCCcore-8.3.0
   nettle/3.6-GCCcore-10.2.0
   nettle/3.7.2-GCCcore-10.3.0                                  (D)
   networkx/2.4-foss-2020a-Python-3.8.2
   nodejs/12.16.1-GCCcore-9.3.0
   nodejs/12.19.0-GCCcore-10.2.0
   nodejs/14.17.0-GCCcore-10.3.0                                (D)
   nsync/1.24.0-GCCcore-10.2.0
   numactl/2.0.12-GCCcore-8.3.0
   numactl/2.0.13-GCCcore-8.3.0
   numactl/2.0.13-GCCcore-9.3.0
   numactl/2.0.13-GCCcore-10.2.0
   numactl/2.0.14-GCCcore-10.3.0                                (D)
   p4est/2.2-intel-2019b
   parallel/20190922-GCCcore-8.3.0
   parallel/20200522-GCCcore-9.3.0                              (D)
   picard/2.21.6-Java-11
   pigz/2.6-GCCcore-10.3.0
   pixman/0.38.4-GCCcore-8.3.0
   pixman/0.38.4-GCCcore-9.3.0
   pixman/0.40.0-GCCcore-10.2.0
   pixman/0.40.0-GCCcore-10.3.0                                 (D)
   pkg-config/0.29.2-GCCcore-8.3.0
   pkg-config/0.29.2-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
   pkg-config/0.29.2-GCCcore-10.2.0
   pkg-config/0.29.2-GCCcore-10.3.0                         (D)
   pkgconfig/1.5.1-GCCcore-9.3.0-Python-3.8.2
   pkgconfig/1.5.1-GCCcore-10.2.0-python                    (D)
   prodigal/2.6.3-GCCcore-10.2.0
   protobuf-python/3.10.0-foss-2020a-Python-3.8.2
   protobuf-python/3.10.0-gomkl-2020a-Python-3.8.2
   protobuf-python/3.10.0-intel-2019b-Python-3.7.4
   protobuf-python/3.14.0-GCCcore-10.2.0                    (D)
   protobuf/3.10.0-GCCcore-8.3.0
   protobuf/3.10.0-GCCcore-9.3.0
   protobuf/3.14.0-GCCcore-10.2.0                           (D)
   pybind11/2.4.3-GCCcore-8.3.0-Python-3.7.4
   pybind11/2.4.3-GCCcore-9.3.0-Python-3.8.2
   pybind11/2.6.0-GCCcore-10.2.0
   pybind11/2.6.2-GCCcore-10.3.0                            (D)
   rclone/1.42-amd64
   rclone/1.42-foss-2020a-amd64                             (D)
   re2c/1.2.1-GCCcore-8.3.0
   re2c/1.3-GCCcore-9.3.0                                   (D)
   scikit-build/0.10.0-foss-2020a-Python-3.8.2
   snappy/1.1.7-GCCcore-8.3.0
   snappy/1.1.8-GCCcore-9.3.0
   snappy/1.1.8-GCCcore-10.2.0                              (D)
   sparsehash/2.0.3-GCCcore-9.3.0
   tbb/2020.1-GCCcore-9.3.0
   tcsh/6.22.02-GCCcore-8.3.0
   tcsh/6.22.03-GCCcore-10.2.0                              (D)
   time/1.9-GCCcore-8.3.0
   time/1.9-GCCcore-10.2.0                                  (D)
   typing-extensions/3.7.4.3-GCCcore-10.2.0
   utf8proc/2.5.0-GCCcore-10.2.0
   util-linux/2.34-GCCcore-8.3.0
   util-linux/2.35-GCCcore-9.3.0
   util-linux/2.35-intel-2019b
   util-linux/2.36-GCCcore-10.2.0
   util-linux/2.36-GCCcore-10.3.0                           (D)
   x264/20190925-GCCcore-8.3.0
   x264/20191217-GCCcore-9.3.0
   x264/20201026-GCCcore-10.2.0                             (D)
   x265/3.2-GCCcore-8.3.0
   x265/3.3-GCCcore-9.3.0
   x265/3.3-GCCcore-10.2.0                                  (D)
   xorg-macros/1.19.2-GCCcore-8.3.0
   xorg-macros/1.19.2-GCCcore-9.3.0
   xorg-macros/1.19.2-GCCcore-10.2.0
   xorg-macros/1.19.3-GCCcore-10.3.0                        (D)
   yaff/1.6.0-foss-2020a-Python-3.8.2
   zlib/1.2.11-GCCcore-8.2.0
   zlib/1.2.11-GCCcore-8.3.0
   zlib/1.2.11-GCCcore-9.3.0
   zlib/1.2.11-GCCcore-10.2.0
```

(continues on next page)

(continued from previous page)

```
   zlib/1.2.11-GCCcore-10.3.0
   zlib/1.2.11                                         (D)
   zstd/1.4.4-GCCcore-9.3.0
   zstd/1.4.5-GCCcore-10.2.0
   zstd/1.4.9-GCCcore-10.3.0                           (D)

------------------------------- /opt/modulefiles ------------------------------
   gcc/8.1.0

---------------------------- /opt/cray/modulefiles ----------------------------
   PrgEnv-cray/1.0.6

--------------------------- /opt/cray/pe/modulefiles ---------------------------
   cce/10.0.0                  cray-mvapich2_nogpu/2.3.4
   cdt/20.05                   cray-mvapich2_nogpu_gnu/2.3.3
   cray-ccdb/3.0.5             cray-mvapich2_nogpu_gnu/2.3.4        (D)
   cray-cti/1.0.7              craype-dl-plugin-py3/mvapich/20.05.1
   cray-fftw/3.3.8.5           craype-dl-plugin-py3/openmpi/20.05.1
   cray-fftw_impi/3.3.8.5      craype/2.6.4
   cray-impi/5                 craypkg-gen/1.3.7
   cray-lgdb/3.0.10            gdb4hpc/3.0.10
   cray-libsci/20.03.1         papi/5.7.0.3
   cray-mvapich2/2.3.3         perftools-base/20.03.0
   cray-mvapich2_gnu/2.3.3     valgrind4hpc/1.0.1

------------------- /opt/cray/pe/craype/default/modulefiles --------------------
   craype-accel-nvidia20    craype-ivybridge
   craype-accel-nvidia35    craype-mic-knl
   craype-accel-nvidia52    craype-network-infiniband (L)
   craype-accel-nvidia60    craype-network-opa
   craype-accel-nvidia70    craype-sandybridge
   craype-broadwell         craype-x86-rome           (L)
   craype-haswell           craype-x86-skylake

  Where:
   Aliases:  Aliases exist: foo/1.2.3 (1.2) means that "module load foo/1.2" will load␣
↪foo/1.2.3
   D:        Default Module
   L:        Module is loaded

Module defaults are chosen based on Find First Rules due to Name/Version/Version modules␣
↪found in the module tree.
See https://lmod.readthedocs.io/en/latest/060_locating.html for details.

Use "module spider" to find all possible modules.
Use "module keyword key1 key2 ..." to search for all possible modules matching
any of the "keys".
```

4.4 Use of ARC for geospatial analysis

WIP, working with Forestry and iGEP to flesh out relevant examples . . .

4.4.1 Introduction

Geospatial analysis problems often require specialized software and data considerations. Here, we lay out some common softwares and give examples of use specific to the geospatial community. We will be forward looking and devote this page to TinkerCliffs and Infer only.

4.4.2 Data location

TinkerCliffs has two main storage systems:

- /projects served by BGFS parallel storage
- /fastscratch served by VAST flash storage

In addition, each compute node has local disk and RAM mounted as a volume.

Generally, data should be moved to the local node for the compute nodes during the computation and results saved, then transfered back to main ARC storage. To see what local storage is available on each compute node, type `env | grep TMP`. This will list the environment variables you can use to access the different storage locations.

4.4.3 Common software and availability

- Python
- Julia
- R
- qGIS

pdal

4.4.4 Interface

There are two types of environments in which the R application can be used on ARC resources:

- Graphical interface via Rstudio *OnDemand*
- Command-line interface. You can also start R from the command line through the Singularity container.

Note: larger computations should be submitted as jobs, via a *traditional job submission* script.

4.4.5 R from the command line

To run R from the command line, we need to load the container software and then jump into the container to see R. From TinkerCliffs, this would look like so:

```
module load containers/singularity/3.7.1
singularity exec -bind=/work,/projects \
    /projects/arcsingularity/ood-rstudio141717-bio_4.1.0.sif R
```

Note: both `/work` and `/projects` are mounted into the container (via bind) so that we can access files outside the container from those storage locations.

4.4.6 R startup, .Renviron and adding packages

R startup is a bit complicated. There is a really nice writeup here:https://rviews.rstudio.com/2017/04/19/r-for-enterprise-understanding-r-s-startup/

The R in the container is expecting a startup file at `~/.Renvron.OOD`. This file needs to have the location of the user packages, for example:

R_LIBS_USER=~/R/OOD/Ubuntu-20.04-4.1.1

This directory must exist prior to starting R. If you use the OnDemand Rstudio, it will be automatically created on your first start of Rstudio.

To install packages from Rstudio, simply do:

install.packages("package of interest")

Warning: When using R rom the command line, you need to reverse the search path of the installed packages prior to installing packages. Make sure the first path in `.libPaths()` is one you can write to:

```
> .libPaths()
> .libPaths(.libPaths()[3:1])
> install.packages("package of interest")
```

R from a Script

As we scale up our computing, we will often find the compute takes too long or we need to run many scripts (models) to get our work done. When this happens, we need to turn to using R via a script. The R script needs to hands free, ie no user action necessary in execution of the full script. To accomplish this on ARC, we actually need two scripts:

1. an R script with the actual R code we are needing to run
2. a shell script for submission to the cluster batch schedulers

The R script should load/generate the data, do the compute, and save the results. As an example, from a login node, you can type:

```
sbatch run_R.sh
```

This will submit the script `run_R.sh` to the (slurm) scheduler. This script in turn, loads the singularity software for running R and runs the R script, `hp_mpg.R`, via Rscript. Both scripts are shown below.

```
## hp_mpg.R
## R script for generating a plot of mpg vs hp
library(ggplot2)
attach(mtcars)
p <- gglot(data=mtcars, aes(x=hp, y=mpg)) + geom_line()
ggsave(file="hp_mpg.pdf",p)
```

Given the R script, we still need a seperate script as the job submission script. This script should contain Slurm directives detailing what compute resources are needed, loading of any required software, and finally running the application of interest.

```
#!/bin/bash

### run_R.sh
###########################################################################
## environment & variable setup
####### job customization
#SBATCH --name="mpg plot"
#SBATCH -N 1
#SBATCH -n 16
#SBATCH -t 1:00:00
#SBATCH -p normal_q
#SBATCH -A <your account>
####### end of job customization
# end of environment & variable setup
###########################################################################
#### add modules on TC/Infer
module load module load containers/singularity/3.7.1
### from DT/CA, use module load singularity
module list
#end of add modules
###########################################################################
###print script to keep a record of what is done
cat hp_mpg.R
cat run_R.sh
###########################################################################
echo start running R
## note, on DT/CA, you should replace projects with groups

singularity exec -bind=/work,/projects \
    /projects/arcsingularity/ood-rstudio141717-bio_4.1.0.sif Rscript hp_mpg.R

exit;
```

Parallel Computing in R

parallel package

MPI

Coming soon-ish

4.5 LS-DYNA

4.5.1 Introduction

LS-DYNA is a general-purpose finite element program capable of simulating complex real world problems. It is used by the automobile, aerospace, construction, military, manufacturing, and bioengineering industries. LS-DYNA is optimized for shared and distributed memory Unix, Linux, and Windows based, platforms, and it is fully QA'd by LSTC. The code's origins lie in highly nonlinear, transient dynamic finite element analysis using explicit time integration.

4.5.2 Availability

LS-DYNA is available on *several ARC systems*. Virginia Tech maintains a limited quantity of LS-DYNA network licenses through the university's IT Procurement and Licensing Solutions which can be used for the SMP, MPP, and Hybrid versions of LS-DYNA. LSTC also develops its own preprocessor, LS-PrePost, which is freely distributed and runs without a license.

License availability

Recent installations of LS-DYNA on ARC systems make available LSTC's license tools which can be used to query the server for licenses which have been checked out, how many are currently available, and kill and "zombified" license checkouts (as happens if LS-DYNA terminates in an unexpected manner).

For the following commands to work, you must have loaded an LS-DYNA module which provides these programs. If it does not provide them, you will get an error like `lstc_qrun: no such file or directory`

Check Number of Licenses Available

- Load the LS-DYNA module (eg. `module load tinkercliffs-rome/ls-dyna/10.2.0-intel-2019b` for v. 10.2 on Tinkercliffs)
- Set and export the `LSTC_LICENSE_SERVER` evironment variable to the name of the license server you want to check (eg. `ansys.software.vt.edu` for the main Virginia Tech LS-DYNA license server).
- Run the command `lstc_qrun -L LS-DYNA` to query SMP licenses or `lstc_qrun -L MPPDYNA` to query MPP licenses.

For example:

```
$ module load tinkercliffs-rome/ls-dyna/10.2.0-intel-2019b
$ export LSTC_LICENSE_SERVER=ansys.software.vt.edu
$ lstc_qrun -L MPPDYNA
Defaulting to server 1 specified by LSTC_LICENSE_SERVER variable
 500 LICENSE(S) AVAILABLE for PROG=MPPDYNA USER=brownm12 HOST=tinkercliffs2 IP=198.82.
↪249.14
$ lstc_qrun -L LS-DYNA
```

(continues on next page)

(continued from previous page)

```
Defaulting to server 1 specified by LSTC_LICENSE_SERVER variable
 500 LICENSE(S) AVAILABLE for PROG=LS-DYNA USER=brownm12 HOST=tinkercliffs2 IP=198.82.
↪249.14
```

Query Licenses Currently Checked Out From License Server

- Load the LS-DYNA module (eg. `module load tinkercliffs-rome/ls-dyna/10.2.0-intel-2019b` for v. 10.2 on Tinkercliffs)
- Set and export the `LSTC_LICENSE_SERVER` evironment variable to the name of the license server you want to check (eg. `ansys.software.vt.edu` for the main Virginia Tech LS-DYNA license server).
- Run the command `lstc_qrun`

```
$ module load tinkercliffs-rome/ls-dyna/10.2.0-intel-2019b
$ export LSTC_LICENSE_SERVER=ansys.software.vt.edu
$ lstc_qrun
Defaulting to server 1 specified by LSTC_LICENSE_SERVER variable


                     Running Programs

    User             Host          Program              Started       # procs
-----------------------------------------------------------------------------
brownm12   205377@tc154.cm.cluster MPPDYNA          Wed Oct 20 10:00    16
No programs queued
```

Kill a zombified LS-DYNA license

- Load the LS-DYNA module (eg. `module load tinkercliffs-rome/ls-dyna/10.2.0-intel-2019b` for v. 10.2 on Tinkercliffs)
- Set and export the `LSTC_LICENSE_SERVER` evironment variable to the name of the license server you want to use (eg. `ansys.software.vt.edu` for the main Virginia Tech LS-DYNA license server).
- Run the command `lstc_qrun` (see above) to and note the "Host" column entry for the program to kill.
- Run the command `lstc_qkill <program to kill>`

```
$ module load tinkercliffs-rome/ls-dyna/10.2.0-intel-2019b
$ export LSTC_LICENSE_SERVER=ansys.software.vt.edu
$ lstc_qkill 205377@tc154.cm.cluster
```

4.5.3 Interface

There are two types of environments in which the LSTC applications can be used on ARC resources:

- Graphical interface for LS-PrePost via *OnDemand*
- Command-line interface. You can also start LS-DYNA from the command line on Unix systems where MATLAB is installed. Note that the command line runs on the login node, so big computations should be submitted as jobs via a *traditional job submission*.

4.5.4 Parallel Computing with LS-DYNA

There are three primary modes of obtaining parallelism in LS-DYNA. All of these are also built to take advantage of microarchitecture vectorization instructions like AVX2 and AVX512 and ARC attempts to provide LS-DYNA executables optimized for local the microarchitecture of the system.

- **SMP**: Shared Memory Parallel. Execution is limited to a single node since the threads require shared access to the same memory space.
- **MPP**: Message Passing Parallel. Several or many processes are launched and run as if each is on its own computer with dedicated memory. The discretization of the domain is divided equally (more or less) between the processes (ie. "domain decomposition") and each process is carries out the simulation on its subdomain. Neighboring subdomains affect each other, so processes must pass messages (MPI) to share the necessary data. This mode can scale to a large number of processors across many machines, but the overhead of subdividing the domain and passing messages becomes significant.
- **Hybrid**: MPP combined with SMP.

As of October 2021, Virginia Tech's central license pool is for `500` concurrent cores which can be allocated among all running programs.

4.5.5 Job Submission

Hybrid

To use the LS-DYNA hybrid mode of parallelism, you need to consider how many MPI processes (aka tasks/ranks) you want and how much SMP (shared memory parallelism) to provide to each MPI process. This combination is also constrained by the total number of licenses available when your job starts. So `ntasks * cpus-per-task` must be a licensable number.

Some scaling tests with example code on Tinkercliffs suggest that the time-to-completion in Hybrid mode does not improve beyond 16 MPP procs and that when the number of MPP procs is scaled beyond 32, it will increase instead of decrease. So we suggest `$SBATCH --ntasks=16` or smaller.

Similar tests show that when the number of SMP threads exceeds 8, the time-to-completion shows high variability and diminished returns, so we suggest `$SBATCH --cpus-per-task=8` with 4 and 16 possibly providing comparable performance.

The `--cpus-per-task` and `--ntasks` options work together to inform Slurm how many cores to allocate for the job and also how to lauch the processes when the `srun` launcher is used. But LS-DYNA also needs to be directed how many threads to use and this is accomplished by providing the `ncpu=-##` option to the LS-DYNA hybrid program.

```
#SBATCH --ntasks=4
#SBATCH --cpus-per-task=8

module reset
module load tinkercliffs-rome/ls-dyna/10.2.0-intel-2019b
export LSTC_LICENSE_SERVER=ansys.software.vt.edu

srun --mpi=pmi2 ls-dyna_hyb_d_R10_2_0_x64_centos65_ifort160_avx2_intelmpi-2018 i=shock02.
↪k ncpu=-$SLURM_CPUS_PER_TASK
```

4.5.6 Example Scaling Results for Hybrid:

```
shock02_nt-8_cpt-2: Elapsed time      22 seconds for   47494 cycles using     8 MPP␣
↪procs and  2 SMP threads
shock02_nt-4_cpt-2: Elapsed time      23 seconds for   47494 cycles using     4 MPP␣
↪procs and  2 SMP threads
shock02_nt-8_cpt-4: Elapsed time      23 seconds for   47494 cycles using     8 MPP␣
↪procs and  4 SMP threads
shock02_nt-4_cpt-4: Elapsed time      24 seconds for   47494 cycles using     4 MPP␣
↪procs and  4 SMP threads
shock02_nt-4_cpt-64: Elapsed time      24 seconds for    7264 cycles using     4 MPP␣
↪procs and 64 SMP threads
shock02_nt-8_cpt-4: Elapsed time      24 seconds for   47494 cycles using     8 MPP␣
↪procs and  4 SMP threads
shock02_nt-4_cpt-1: Elapsed time      25 seconds for   47494 cycles using     4 MPP␣
↪procs and  1 SMP thread
shock02_nt-4_cpt-4: Elapsed time      25 seconds for   47494 cycles using     4 MPP␣
↪procs and  4 SMP threads
shock02_nt-4_cpt-4: Elapsed time      25 seconds for   47494 cycles using     4 MPP␣
↪procs and  4 SMP threads
shock02_nt-4_cpt-8: Elapsed time      25 seconds for   47494 cycles using     4 MPP␣
↪procs and  8 SMP threads
shock02_nt-16_cpt-2: Elapsed time      26 seconds for   47494 cycles using    16 MPP␣
↪procs and  2 SMP threads
shock02_nt-8_cpt-4: Elapsed time      26 seconds for   47494 cycles using     8 MPP␣
↪procs and  4 SMP threads
shock02_nt-2_cpt-8: Elapsed time      27 seconds for   47494 cycles using     2 MPP␣
↪procs and  8 SMP threads
shock02_nt-4_cpt-8: Elapsed time      27 seconds for   47494 cycles using     4 MPP␣
↪procs and  8 SMP threads
shock02_nt-8_cpt-1: Elapsed time      27 seconds for   47494 cycles using     8 MPP␣
↪procs and  1 SMP thread
shock02_nt-16_cpt-2: Elapsed time      28 seconds for   47494 cycles using    16 MPP␣
↪procs and  2 SMP threads
shock02_nt-2_cpt-1: Elapsed time      28 seconds for   47494 cycles using     2 MPP␣
↪procs and  1 SMP thread
shock02_nt-2_cpt-4: Elapsed time      28 seconds for   47494 cycles using     2 MPP␣
↪procs and  4 SMP threads
shock02_nt-8_cpt-16: Elapsed time      28 seconds for   47494 cycles using     8 MPP␣
↪procs and 16 SMP threads
shock02_nt-8_cpt-2: Elapsed time      28 seconds for   47494 cycles using     8 MPP␣
↪procs and  2 SMP threads
shock02_nt-16_cpt-1: Elapsed time      29 seconds for   47494 cycles using    16 MPP␣
↪procs and  1 SMP thread
shock02_nt-2_cpt-8: Elapsed time      29 seconds for   47494 cycles using     2 MPP␣
↪procs and  8 SMP threads
shock02_nt-1_cpt-4: Elapsed time      30 seconds for   47494 cycles using     1 MPP proc␣
↪ and  4 SMP threads
shock02_nt-2_cpt-2: Elapsed time      30 seconds for   47494 cycles using     2 MPP␣
↪procs and  2 SMP threads
shock02_nt-16_cpt-2: Elapsed time      31 seconds for   47494 cycles using    16 MPP␣
↪procs and  2 SMP threads
shock02_nt-32_cpt-1: Elapsed time      31 seconds for   47494 cycles using    32 MPP␣
↪procs and  1 SMP thread
```

(continues on next page)

(continued from previous page)

```
shock02_nt-32_cpt-1: Elapsed time      31 seconds for   47494 cycles using    32 MPP␣
↪procs and  1 SMP thread
shock02_nt-16_cpt-4: Elapsed time      32 seconds for   47494 cycles using    16 MPP␣
↪procs and  4 SMP threads
shock02_nt-32_cpt-2: Elapsed time      32 seconds for   47494 cycles using    32 MPP␣
↪procs and  2 SMP threads
shock02_nt-16_cpt-4: Elapsed time      33 seconds for   47494 cycles using    16 MPP␣
↪procs and  4 SMP threads
shock02_nt-2_cpt-16: Elapsed time      33 seconds for   47494 cycles using     2 MPP␣
↪procs and 16 SMP threads
shock02_nt-2_cpt-2: Elapsed time      33 seconds for   47494 cycles using     2 MPP␣
↪procs and  2 SMP threads
shock02_nt-2_cpt-8: Elapsed time      33 seconds for   47494 cycles using     2 MPP␣
↪procs and  8 SMP threads
shock02_nt-32_cpt-2: Elapsed time      33 seconds for   47494 cycles using    32 MPP␣
↪procs and  2 SMP threads
shock02_nt-8_cpt-1: Elapsed time      33 seconds for   47494 cycles using     8 MPP␣
↪procs and  1 SMP thread
shock02_nt-8_cpt-2: Elapsed time      33 seconds for   47494 cycles using     8 MPP␣
↪procs and  2 SMP threads
```

4.6 MATLAB

4.6.1 Introduction

MATLAB handles a range of computing tasks in engineering and science, from data acquisition and analysis to application development. The MATLAB environment integrates mathematical computing, visualization, and a powerful technical language. It is especially well-suited to vectorized calculations and has a Parallel Computing Toolbox (not included in all licenses) that streamlines parallelization of code.

4.6.2 Availability

MATLAB is available on *several ARC systems*. ARC maintains a MATLAB Distributed Computing Server license for parallel MATLAB through cooperation with the university's IT Procurement and Licensing Solutions, who also offer discounted licenses to departments and students (note that MATLAB is also included in some of the Student Bundles).

4.6.3 Interface

There are two types of environments in which the MATLAB application can be used on ARC resources:

- Graphical interface via *OnDemand*
- Command-line interface. You can also start MATLAB from the command line on Unix systems where MATLAB is installed. Note that the command line runs on the login node, so big computations should be submitted as jobs, either from via a *traditional job submission* or from within MATLAB.

4.6.4 Parallel Computing in MATLAB

There are two primary means of obtaining parallelism in MATLAB:

- **parfor**: Replacing a for loop with a parfor loop splits the loop iterations among a group of processors. This requires that the loop iterations be independent of each other.
- **spmd**: Single program multiple data (spmd) allows multiple processors to execute a single program (similar to MPI).

4.6.5 Job Submission

This page contains instructions for submitting jobs from MATLAB to ARC clusters.

Note: Right now this documentation applies to TinkerCliffs and Infer only, and only allows intracluster job submission (from cluster login nodes). More general information on jobs on ARC machines is available *here* and in the *video tutorials*.

Setup

Setup is as simple as starting MATLAB on a login node and then running

```
>> configCluster
```

Note: Do this once on TinkerCliffs or Infer, or anytime you switch between clusters. (Or anytime you start MATLAB - it won't hurt to run it more often than necessary.)

Running Jobs

After that, the key commands are:

- `c=parcluster` to get the cluster configuration
- `c.AdditionalProperties` is a structure where you can set job parameters. You must set `AccountName` to the allocation account to which you want to submit the job; the other paramters are optional. Commonly-used properties are:
  - `AccountName`: Allocation account (required)
  - `WallTime`
  - `Partition`
  - `GpusPerNode`
  - `AdditionalSubmitArgs`: Any other standard flags that you want to submit directly to the scheduler
- `batch(c,...)` to submit the job

An example is *below*.

Checking Jobs

The job structure returned by `batch()` can be queried to get the job state, outputs, diary (command line output), etc. See the *example* below.

MATLAB also comes with a Job Monitor to allow tracking of remote jobs via a graphical interface. Right-clicking on jobs will allow you to show its output, load its variables, delete it, etc.

Remote Output Files

Remote MATLAB jobs start in the directory specified by the `CurrentFolder` parameter to `batch()`. Output files written to remote jobs will be saved in this location. Alternatively, you may specify the full path to where you want it to save the file, e.g.

```
save('/home/johndoe/output')
```

Note that if you submit from your personal machine, these files will not be copied back to your local machine; you will need to manually *log into the machine* to get them. Alternatively, you can tell MATLAB to change to the directory on the ARC cluster where job information is stored; MATLAB will automatically mirror this location to your local machine when the job completes. Here is an example command for switching to the job directory:

```
cd(sprintf('%s/%s',getenv('MDCE_STORAGE_LOCATION'),getenv('MDCE_JOB_LOCATION')));
```

Note that once the job completes, you will need to look in its local job directory to get the output files; this location can be configured in your local cluster profile. Be sure to remove any output files you need before deleting your job (e.g. with the `delete` command).

Full Example

Here we set up a cluster profile and then submit a job to compute the number of primes between 1 and 10 million using the prime_fun parallel MATLAB example. MATLAB runs the job and returns the correct answer: 664,579.

(Note that to run this example, we've downloaded the code to a directory on TinkerCliffs and then changed to that directory.)

```
[johndoe@tinkercliffs2 prime_fun]$ module load $LMOD_SYSTEM_NAME/matlab/R2021a
[johndoe@tinkercliffs2 prime_fun]$ matlab -nodisplay

< M A T L A B (R) >
Copyright 1984-2021 The MathWorks, Inc.
R2021a (9.10.0.1602886) 64-bit (glnxa64)
February 17, 2021


To get started, type doc.
For product information, visit www.mathworks.com.

>> configCluster
>> c = parcluster;
>> c.AdditionalProperties.AccountName = 'arcadm';
>> j = batch(c,@prime_fun,1,{10000000},'pool',4);

additionalSubmitArgs =
```

(continues on next page)

(continued from previous page)

```
    '--ntasks=5 --cpus-per-task=1 --ntasks-per-core=1 -A arcadm'

>> j.State

ans =

    'running'

>> j.State

ans =

    'finished'

>> j.fetchOutputs{1}

ans =

      664579
```

4.6.6 Submitting Jobs from the Linux Command Line

MATLAB jobs can also be submitted from the Linux command line like any other jobs; however, the parallelism is currently limited to the cores on a single node. This example uses `parfor` to count in parallel the prime numbers between 1 and 10,000,000. (The correct answer is 664,579.) A submission script to submit it as a job from the command line is provided here. To submit it as a job using your `personal` *allocation* use:

```
sbatch -Apersonal matlab_tinkercliffs_rome.sh
```

More general information on jobs on ARC machines is available *here* and in the *video tutorials*.

4.6.7 Changing MATLAB's Path

To add a folder to MATLAB's path on ARC's systems, edit the `MATLABPATH` environment variable. This can be made permanent by editing it in your `.bashrc` file. For example, this line would add the folder `mydir` in your Home directory to MATLAB's path anytime it opens in your account:

```
echo "export MATLABPATH=\\$HOME/mydir:\$MATLABPATH\" >> ~/.bashrc
```

An alternative is to create a `pathdef.m` file in the directory where MATLAB starts. This will add folders to MATLAB's path whenever it starts in the folder where `pathdef.m` is located. For example, the following at the MATLAB command line would add `mydir` to the path when MATLAB opens in your Home directory:

```
addpath('/home/johndoe/mydir');
savepath('/home/johndoe/pathdef.m')
```

4.6.8 Using the MATLAB Compiler (mex)

To compile C/C++ or Fortran code in MATLAB, just make sure to load the compiler *module* that you want to use before you open MATLAB. Here is an example of compiling MatConvNet, which in this case requires the GCC compiler, which is available via the `foss` module:

```
#load modules
module reset; module load foss/2020b matlab/R2021a

#open matlab and do the install
#(vl_compilenn is the installer script in this case)
matlab -nodisplay
[matlab starts]
>> vl_compilenn
```

4.7 Python

4.7.1 Introduction

Python is free software for computing and graphics used heavily in the AI/ML space.

4.7.2 Availability

Python is available on all clusters in all queues (partitions) through Python modules, Anaconda modules or Singularity containers.

4.7.3 Interface

There are two types of environments in which the python application can be used on ARC resources:

- Graphical interface via *OnDemand* using Jupyter
- Command-line interface. You can also start python from the command line after loading the required software module.

Note: Larger computations should be submitted as jobs, via a *traditional job submission* script.

4.7.4 Managing environments

The power of python is through extension of the base functionality via python packages. Managing and configuring your local python environment is best accomplished through a combination of a package manager (pip or conda) and an evironment manager Anaconda (or miniconda or micoromamba). Creation and use of conda environments allows one to activate the environment for later use. You can have several environments, each with different software dependencies, where you activate the one of interest at run time. Commonly, you will create a conda env, install software into it via conda/pip and then activate it for use. For example:

```
module load Anaconda3/2020.11
conda create -n mypy3 python=3.8 pip
source activate mypy3
conda install ipykernel
pip install plotly kaleido
```

Source activating the environment ensures later conda or pip installs will install into the environment location. For a more full discussion and examples, please see the Anaconda documentation:https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

4.7.5 Running without environments

If you prefer to use python without an environment, you will need to set the PYTHONUSERBASE environment variable to a location you can write to. For example:

```
#load a python module
module reset; module load Python/3.8.6-GCCcore-10.2.0
#give python a directory where it can install/load personalized packages
#you may want to make this more specific to cluster/node type/python version
export PYTHONUSERBASE=$HOME/python3
#install a package (--user tells python to install to the location
#specified by PYTHONUSERBASE)
pip install --user plotly
```

4.7.6 Command line running of Python scripts

First, we need both a python script and (likely) the conda environment setup. The environment for this example was shown above as mypy3.

```
## violins.py
import plotly.express as px
# using the tips dataset
df = px.data.tips()
# plotting the violin chart
fig = px.violin(df, x="day", y="total_bill")
fig.write_image("fig1.jpeg")
```

Second, we need a shell script to submit to the Slurm scheduler. The script needs to specify the required compute resources, load the required software and finally run the actual script.

```
#!/bin/bash

### python.sh
###########################################################################
## environment & variable setup
####### job customization
#SBATCH -N 1
#SBATCH -n 16
#SBATCH -t 1:00:00
#SBATCH -p normal_q
#SBATCH -A <your account>
```

(continues on next page)

(continued from previous page)

```
####### end of job customization
# end of environment & variable setup
###########################################################################
#### add modules:
module load Anaconda/2020.11
module list
#end of add modules
###########################################################################
###print script to keep a record of what is done
cat python.sh
echo "python code"
cat violins.py
###########################################################################
echo start load env and run python

source activate mypy3
python violins.py

exit;
```

Finally, to run both the batch script and python, we type:

```
sbatch python.sh
```

This will output a job number. You will have two output files:

- fig1.jpeg
- slurm-JOBID.log

The slurm log contains any output you would have seen had you typed `python violins.py` at the command line.

4.7.7 Parallel Computing in Python

Coming soon-ish. In the meantime, an mpi4py example is provided as part of ARC's examples repository.

4.8 PyTorch

4.8.1 Introduction

Pytorch, as described on their website is: "an open source machine learning framework that accelerates the path from research prototyping to production deployment".

4.8.2 Availability

PyTorch is not implicitly installed on ARC systems, but is readily installed via Conda, pip or source. To install via Conda on TinkerCliffs or Infer, you should first get an interactive job on a GPU node (or CPU if that is where you will compute), load Anaconda and then create the environment.

```
## on TC for a100 nodes:
interact --account=<your research allocation> --partition=a100_normal_q -N 1 -n 12 --
↪gres=gpu:1
module load Anaconda3/2020.11
module list ## make sure cuda is loaded if you are using the GPU
nvidia-smi  ## make sure you see GPUs
conda create -n pytorch
source activate pytorch
conda install pytorch torchvision torchaudio matplotlib numpy -c pytorch
```

Warning: NOTE: GPU support for AI/ML codes can offer SIGNIFFICANT computational speed improvments. Simply installing the defaults as per the docs may or may not result in code utilizing the GPUs. Test your code with a small example prior to running your full dataset. You can ssh to the node your job is running on and use nvidia-smi to see that your code is running on the GPU.

4.8.3 Interaction

You can run PyTorch code from Jupyter Notebooks or via the command line (interactive or scripts). Ideally, you will prototype your code via Jupyter which is easily accessible from Open OnDemand (ood). If instead, you would prefer to prototype your code via the command line, first get an interactive job as above in the install notes, then load the required software, eg Anaconda.

4.8.4 Quick example from the pytorch.org site

The PyTorch tutorials are excellant. For brevity, we can run through the CIFAR10 example from the PyTorch docs:https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html#sphx-glr-beginner-blitz-cifar10-tutorial-py

Here is the example python script, you can run it manually or via `python cifar10.py`

```
## cifar10.py
## import libraries
import torch
import torchvision
import torchvision.transforms as transforms

## get data and set class labels
transform = transforms.Compose(
    [transforms.ToTensor(),
     transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))])

batch_size = 4

trainset = torchvision.datasets.CIFAR10(root='./data', train=True,
                                        download=True, transform=transform)
```

(continues on next page)

(continued from previous page)

```
trainloader = torch.utils.data.DataLoader(trainset, batch_size=batch_size,
                                          shuffle=True, num_workers=2)

testset = torchvision.datasets.CIFAR10(root='./data', train=False,
                                       download=True, transform=transform)
testloader = torch.utils.data.DataLoader(testset, batch_size=batch_size,
                                         shuffle=False, num_workers=2)

classes = ('plane', 'car', 'bird', 'cat',
           'deer', 'dog', 'frog', 'horse', 'ship', 'truck')

## plot some dat for fun, if doing this via a script, you need to push this to a file or␣
↪comment out
import matplotlib.pyplot as plt
import numpy as np

# functions to show an image


def imshow(img):
    img = img / 2 + 0.5     # unnormalize
    npimg = img.numpy()
    plt.imshow(np.transpose(npimg, (1, 2, 0)))
    plt.show()


# get some random training images
dataiter = iter(trainloader)
images, labels = dataiter.next()

# show images
imshow(torchvision.utils.make_grid(images))
# print labels
print(' '.join('%5s' % classes[labels[j]] for j in range(batch_size)))

## setup the NN
import torch.nn as nn
import torch.nn.functional as F


class Net(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(3, 6, 5)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(6, 16, 5)
        self.fc1 = nn.Linear(16 * 5 * 5, 120)
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, 10)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
```

(continues on next page)

(continued from previous page)

```
        x = self.pool(F.relu(self.conv2(x)))
        x = torch.flatten(x, 1) # flatten all dimensions except batch
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x


net = Net()

## define the loss function
import torch.optim as optim

criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)

## train the network

for epoch in range(2):  # loop over the dataset multiple times

    running_loss = 0.0
    for i, data in enumerate(trainloader, 0):
        # get the inputs; data is a list of [inputs, labels]
        inputs, labels = data

        # zero the parameter gradients
        optimizer.zero_grad()

        # forward + backward + optimize
        outputs = net(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

        # print statistics
        running_loss += loss.item()
        if i % 2000 == 1999:    # print every 2000 mini-batches
            print('[%d, %5d] loss: %.3f' %
                  (epoch + 1, i + 1, running_loss / 2000))
            running_loss = 0.0

print('Finished Training')

## save it if you want to keep it
PATH = './cifar_net.pth'
torch.save(net.state_dict(), PATH)

## test it if that's your thing
dataiter = iter(testloader)
images, labels = dataiter.next()

# print images
```

(continues on next page)

(continued from previous page)

```
imshow(torchvision.utils.make_grid(images))
print('GroundTruth: ', ' '.join('%5s' % classes[labels[j]] for j in range(4)))
```

4.8.5 Parallel Computing in Python

Coming soon-ish

4.8.6 Command line running of Python

Coming soon-ish

```
module load Anaconda3/2020.11
conda create -n mypython3 python=3
source activate mypython3
```

4.8.7 Managing environments

https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

Full Example

4.9 R

4.9.1 Introduction

R is free software for statistical computing and graphics.

4.9.2 Availability

R is available on all our systems. We are moving towards making R available via containers, specifically Singularitiy. Our containers are built using Docker and converted to Singularity. Several versions of R are available. Each R version is usually available with different package subsets for specific domain usages:

- ood-rstudio-basic
- ood-rstudio-bio
- ood-rstudio-geospatial
- ood-rstudio-keras
- ood-rstudio-qiime2

The Dockerfiles are available on GitHub searching for "ood-rstudio" and the images available on DockerHub searching for "rsettlag/ood-rstudio". The easiest way to see what libraries are installed in the container is to simply start the Rstudio app via Open Ondemand.

If you need additional packages or R versions, please open an issue on GitHub.

4.9.3 Interface

There are two types of environments in which the R application can be used on ARC resources:

- Graphical interface via Rstudio *OnDemand*
- Command-line interface. You can also start R from the command line through the Singularity container.

Note: larger computations should be submitted as jobs, via a *traditional job submission* script.

4.9.4 R from the command line

To run R from the command line, we need to load the container software and then jump into the container to see R. From TinkerCliffs, this would look like so:

```
module load containers/singularity/3.7.1
singularity exec -bind=/work,/projects \
    /projects/arcsingularity/ood-rstudio141717-bio_4.1.0.sif R
```

Note: both `/work` and `/projects` are mounted into the container (via bind) so that we can access files outside the container from those storage locations.

4.9.5 R startup, .Renviron and adding packages

R startup is a bit complicated. There is a really nice writeup here:https://rviews.rstudio.com/2017/04/19/r-for-enterprise-understanding-r-s-startup/

The R in the container is expecting a startup file at `~/.Renvron.OOD`. This file needs to have the location of the user packages, for example:

R_LIBS_USER=~/R/OOD/Ubuntu-20.04-4.1.1

This directory must exist prior to starting R. If you use the OnDemand Rstudio, it will be automatically created on your first start of Rstudio.

To install packages from Rstudio, simply do:

install.packages("package of interest")

Warning: When using R rom the command line, you need to reverse the search path of the installed packages prior to installing packages. Make sure the first path in `.libPaths()` is one you can write to:

```
> .libPaths()
> .libPaths(.libPaths()[3:1])
> install.packages("package of interest")
```

R from a Script

As we scale up our computing, we will often find the compute takes too long or we need to run many scripts (models) to get our work done. When this happens, we need to turn to using R via a script. The R script needs to hands free, ie no user action necessary in execution of the full script. To accomplish this on ARC, we actually need two scripts:

1. an R script with the actual R code we are needing to run
2. a shell script for submission to the cluster batch schedulers

The R script should load/generate the data, do the compute, and save the results. As an example, from a login node, you can type:

```
sbatch run_R.sh
```

This will submit the script `run_R.sh` to the (slurm) scheduler. This script in turn, loads the singularity software for running R and runs the R script, `hp_mpg.R`, via Rscript. Both scripts are shown below.

```
## hp_mpg.R
## R script for generating a plot of mpg vs hp
library(ggplot2)
attach(mtcars)
p <- ggplot(data=mtcars, aes(x=hp, y=mpg)) + geom_line()
ggsave(file="hp_mpg.pdf",p)
```

Given the R script, we still need a seperate script as the job submission script. This script should contain Slurm directives detailing what compute resources are needed, loading of any required software, and finally running the application of interest.

```
#!/bin/bash

### run_R.sh
###########################################################################
## environment & variable setup
####### job customization
#SBATCH --job-name="mpg plot"
#SBATCH -N 1
#SBATCH -n 16
#SBATCH -t 1:00:00
#SBATCH -p normal_q
#SBATCH -A <your account>    #### <-------- change me
####### end of job customization
# end of environment & variable setup
###########################################################################
#### add modules on TC/Infer
module load containers/singularity/3.7.1
### from DT/CA, use module load singularity
module list
#end of add modules
###########################################################################
###print script to keep a record of what is done
cat hp_mpg.R
cat run_R.sh
###########################################################################
echo start running R
```

(continues on next page)

(continued from previous page)

```
## note, on DT/CA, you should replace projects with groups

singularity exec --bind=/work,/projects \
    /projects/arcsingularity/ood-rstudio141717-bio_4.1.0.sif Rscript hp_mpg.R

exit;
```

4.9.6 Parallel Computing in R

There are multiple ways to afford parallelism from within R. Depending on how you parallelize, you may need to alter your SLURM job request.

parallel package

bootstrap example with mcapply

```
# parallel_mcapply.R
library(parallel)

### make some data
x <- matrix(c(rep(1,100),runif(100),runif(100,max=10)),ncol=3,byrow=FALSE)
beta <- matrix(1:3,nrow=3)
y <- x %*% beta + rnorm(100)

f <- function(x_mat=x,y_mat=y,z){
  boot_coef <- sample(1:100,size=100,replace=TRUE);
  results<-lm(y_mat[boot_coef]~0+x_mat[boot_coef,])$coefficients
  names(results)<-c("beta0","beta1","beta2")
  return(results)
}

#numCores <- detectCores()
numCores <- parallelly::availableCores()
numreps <- 10000
results <- rep(0,numreps) ## preallocate to get compute timing

cat("setting cores to: ",numCores,sep="\n")

cat("using lapply \n")
system.time(
  results <- lapply(1:numreps,function(i)  f())
)
rowMeans(sapply(results,"["))

cat("using mcapply \n")
system.time(
  results <- mclapply(1:numreps,function(i)  f(), mc.cores = numCores)
)
rowMeans(sapply(results,"["))
```

To use:

```
interact -N 1 -c 12 --partition=intel_q --time=5:00:00 --account=<your account>
module load containers/singularity
singularity exec /projects/arcsingularity/ood-rstudio141717-bio_4.1.0.sif Rscript␣
↪parallel_mcapply.R
```

Note: a) specify the number of cores via SLURM `--cores-per-task`, NOT `--ntasks`.b) detectCores() does not work as intended. detectCores() will query to get the cores on the node, not the cores in the job. Use availableCores() from the parallelly package instead.

doParallel example

```
# parallel_doparallel.R
library(foreach)
library(doParallel)
numCores <- parallelly::availableCores()

registerDoParallel(numCores)  # use multicore, set to the number of our cores
foreach (i=1:100, .combine=c) %dopar% {
  tanh(i)
}

stopImplicitCluster() ## clean up
```

WIP:

Danger: proceed with caution below, you may encounter bumps. . .

MPI

Still in testing, but, we are using a bind option to get OpenMPI into the container. See here for a discussion. From there, we need to install Rmpi.

```
$ module load OpenMPI/4.1.1-GCC-10.3.0 containers/singularity
$ export SINGULARITYENV_LD_LIBRARY_PATH=$LD_LIBRARY_PATH
$ singularity exec --writable-tmpfs
      --bind=$TMPFS:/tmp,/usr/include/bits,/apps,/cm,/usr/bin/ssh \
      --bind=/home/rsettlag/.Renviron.OOD:/usr/local/lib/R/etc/Renviron.site \
      /projects/arcsingularity/ood-rstudio141717-bio_4.1.0.sif bash
singularity> R CMD INSTALL Rmpi_0.6-7.tar.gz --configure-args=--with-mpi=/apps/easybuild/
↪software/tinkercliffs-cascade_lake/OpenMPI/4.1.1-GCC-10.3.0 --no-test-load
```

To use Rmpi, we need to:

a) make sure the configuration of the job matches what we desire in terms of processes and coresb) use `mpirun` to launch R and subsequently Rmpi

```
# current working example:
export PMIX_MCA_gds=hash ## was supposedly fixed in OMPI 4.0.3+, but here we are in 4.1.
↪1...

mpirun -np 8 singularity exec --writable-tmpfs --bind=$TMPFS:/tmp,/usr/include/bits,/
↪apps,/cm,/usr/bin/ssh /projects/arcsingularity/ood-rstudio141717-bio_4.1.0.sif /home/
↪rsettlag/examples/mpitest

## prepping for some of the R errors:
mpirun -np 8 --mca mpi_warn_on_fork 0 --mca btl_openib_allow_ib 1 --mca rmaps_base_
↪inherit 1 singularity exec --writable-tmpfs --bind=$TMPFS:/tmp,/usr/include/bits,/apps,
↪/cm,/usr/bin/ssh,/home/rsettlag/.Renviron.OOD:/usr/local/lib/R/etc/Renviron.site /
↪projects/arcsingularity/ood-rstudio141717-bio_4.1.0.sif /home/rsettlag/examples/mpitest
```

Where mpitest.c is:

```
# mpitest.c
#include <mpi.h>
#include <stdio.h>
#include <stdlib.h>

int main (int argc, char **argv) {
        int rc;
        int size;
        int myrank;

        rc = MPI_Init (&argc, &argv);
        if (rc != MPI_SUCCESS) {
                fprintf (stderr, "MPI_Init() failed");
                return EXIT_FAILURE;
        }

        rc = MPI_Comm_size (MPI_COMM_WORLD, &size);
        if (rc != MPI_SUCCESS) {
                fprintf (stderr, "MPI_Comm_size() failed");
                goto exit_with_error;
        }

        rc = MPI_Comm_rank (MPI_COMM_WORLD, &myrank);
        if (rc != MPI_SUCCESS) {
                fprintf (stderr, "MPI_Comm_rank() failed");
                goto exit_with_error;
        }

        fprintf (stdout, "Hello, I am rank %d/%d\n", myrank, size);

        MPI_Finalize();

        return EXIT_SUCCESS;

 exit_with_error:
        MPI_Finalize();
        return EXIT_FAILURE;
```

(continues on next page)

(continued from previous page)

```
}
```

compiled from `INSIDE` the container with:

> mpicc -o mpitest mpitest.c

Example coming soon…

Current non-working Rmpi example – non-working IN a container…

```
# mpi_r.R
# Load the R MPI package if it is not already loaded.
if (!is.loaded("mpi_initialize")) {
library("Rmpi")
}
print(mpi.universe.size())
ns <- mpi.universe.size() - 1
mpi.spawn.Rslaves(nslaves=ns)
#
# In case R exits unexpectedly, have it automatically clean up
# resources taken up by Rmpi (slaves, memory, etc...)
.Last <- function(){
if (is.loaded("mpi_initialize")){
if (mpi.comm.size(1) > 0){
print("Please use mpi.close.Rslaves() to close slaves.")
mpi.close.Rslaves()
}
print("Please use mpi.quit() to quit R")
.Call("mpi_finalize")
}
}
# Tell all slaves to return a message identifying themselves
mpi.remote.exec(paste("I am",mpi.comm.rank(),"of",mpi.comm.size(),system("hostname",
↪intern=T)))
# Test computations
x <- 5
x <- mpi.remote.exec(rnorm, x)
length(x)
x
# Tell all slaves to close down, and exit the program
mpi.close.Rslaves()
```

4.10 Singularity

4.10.1 Introduction

Singularity is free software for containerizing applications.

4.10.2 Availability

Singularity is available across all our systems.

4.10.3 Usage

Using containers on our systems amounts to loading the software and starting the image. On Tinkercliffs/Infer, to run a Jupyter container with Julia:

```
module load containers/singularity
singularity exec --bind=/work,/projects,`pwd`:/opt/julia/logs \
    /projects/arcsingularity/AMD/ood-jupyter-datascience_tcamd_1Dec2020.sif julia
```

The above commands load the singularity software using our module system, then starts Julia wihtin the container. To make data from our main storage locations available within the container, we use the `--bind` command. Additionally, Julia wants to write logs to `/opt/julia/logs/`. Since the container is not writable, we need to bind a mountable location to that container location as given by `pwd`:/opt/julia/logs. This makes the current location available IN the container as `/opt/julia/logs/` and allows Julia to create a log file.

4.10.4 Container building workflow

Because Singularity can build from DockerHub and the public help via Google searches is vastly greater when creating Docker images, our general recommendation is to take advantage of this.

Our workflow is to:

1. create a docker image
2. push docker image to dockerhub
3. `singularity build image.sif docker://<docker user>/image:tag`

4.11 STATA

4.11.1 Introduction

Stata is free software for statistical computing and graphics.

4.11.2 Availability

STATA is available on Dragonstooth and Cascades systems. Currently, only STATA 14.0 is available. This is a 16-core MP license.

4.11.3 Interface

There are two types of environments in which the STATA application can be used on ARC resources:

- Graphical interface via *OnDemand*
- Command-line interface. You can also start STATA from the command line after loading the software module.

Note: larger command line computations should be submitted as jobs, via a *traditional job submission*.

4.11.4 STATA from the command line

To run STATA from the command line, we need to:

1. start a job (either interactive or in a script)
2. load the software module
3. start stata

From Dragonstooth for an interactive job, this would look like so:

```
interact -N 1 -n 16 --partition=normal_q --time=1:00:00 --account=<your account>
module load stata/14.0
stata-mp
```

The above lines should be typed from one of the Dragonstooth login nodes. Note, the interactive job request is looking for 16-cores on a single node where <your account> should be replaced with a Slurm allocation you have access to. If you are unsure what accounts you have access to, go to ood.arc.vt.edu, go to the Tinkercliffs shell, type `showusage` to get a summary of your accounts.

Full Script Example

To run STATA via a script, you need to create a `do` file and execute that in a hands free mode, ie no user input.

As an example of a `do` file named `cool_stata_analysis.do` which assumes you have a data file named `filename` with variables included as shown:

```
* cool_stata_analysis.do
clear
set mem 200m
use filename
log using mylog,text replace
ge lsales3 = log(sales3)
xi:boxcox sales3 pr* i.store
regress lsales3 pr* i.store
log close
```

Now, to run this file in a script, we need to create a submission script:

```
#!/bin/bash

### STATA.sh

```

(continues on next page)

(continued from previous page)

```
###########################################################################
## environment & variable setup

####### job customization
#SBATCH -N 1
#SBATCH -n 16
#SBATCH -t 1:00:00
#SBATCH -p normal_q
#SBATCH -A <your account>
####### end of job customization
# end of environment & variable setup
###########################################################################
#### add modules:
module load stata/14.0
module list
#end of add modules
###########################################################################
###print script to keep a record of what is done
cat STATA.sh
echo "stata code"
cat cool_stata_analysis.do
###########################################################################
echo start running stata
stata -b cool_stata_analysis.do

exit;
```

Finally, to run both the batch script and stata, we type:

```
sbatch STATA.sh
```

This will output a job number. You will have two output files:

- cool_stata_analysis.log
- slurm-JOBID.log

The first, you already know about. The second contains any output you would have seen had you typed `stata -b cool_stata_analysis.do` at the command line.

4.12 Tensorflow

4.12.1 Introduction

Tensorflow is free software for AI/ML applications.

4.12.2 Availability

4.12.3 Interface

4.12.4 Parallel Computing in Python

Coming soon-ish

4.12.5 Command line running of Python

Coming soon-ish

```
module load Anaconda3/2020.11
conda create -n mypython3 python=3
source activate mypython3
```

4.12.6 Managing environments

https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

Full Example

CHAPTER FIVE

USAGE

Contents:

5.1 Allocations

5.1.1 Introduction

ARC's primary mission is to facilitate breakthrough research at Virginia Tech. To this end, ARC uses an allocation system to ensure that system time is distributed in a manner appropriate to research needs while allowing faculty members and PIs the flexibility to ensure that the time allocated to a given project is managed (e.g., among graduate students) so as to maximum productivity. An allocation is a system time account requested and managed by a single person (e.g., a project PI). Many users (e.g., Co-PIs or graduate students) can then be granted access to charge against a single allocation.

Note: Allocation applications can also include requests for other resources (e.g., additional storage) required to make a project successful.

5.1.2 Allocation Types

There are two types of allocations, which differ somewhat in how they are awarded:

Research Allocations are provided for research projects and usually managed by the project's Principal Investigator (PI) (see Eligibility for Research Allocations, below). They are typically granted for a single year and can be renewed annually for the length of the project. Multi-year research allocations, such as for inclusion in a proposal submission, may be granted through negotiation with ARC. Instructional Allocations support academic classes and are managed by the faculty member or instructor responsible for the course. Instructional allocations are typically smaller, available for shorter time periods (e.g., for the duration of the associated course), and may be limited to a select set of systems. Eligibility for Research Allocations

Funds on ARC's systems are intended to ensure that users have the computing resources required to complete their research while also ensuring that no single user or group of users dominates the systems to the detriment of others. As such, allocations are awarded on a project-by-project basis and intended to be managed by the individual responsible for overseeing the research.

In order to manage a research project or allocation on ARC's systems, a user must fall into one of the following categories:

Be a current faculty member or post-doctoral researcher at Virginia Tech, OR Be an employee of Virginia Tech and the Principal Investigator (PI) for a research computing-related project, OR Be an employee of Virginia Tech and the Co-PI for a research computing-related project led by a non-Virginia Tech PI Adjunct professors must provide a letter from their department chair, indicating that they are qualified to lead an internal research project, before their project and allocation requests can be approved.

Undergraduate and graduate students are not eligible to apply directly for projects and allocations, but must instead work under the sponsorship of a qualified researcher.

5.1.3 Student eligibility

Undergraduate and graduate students should ask their advisor or research project PI to submit an allocation request. Once the request has been granted, they can be added to the project and submit jobs.

5.2 Frequently Asked Questions

5.2.1 Why can't I log in?

Log in problems can occur for a number of reasons. If you cannot log into one of ARC's systems, please check the following:

1. **Is your PID password expired?** Try logging into onecampus.vt.edu. If you cannot log in there, then your PID password has likely expired and needs to be changed. (Contact 4Help for help with this issue.)
2. **Are you on-campus?** If you are not on-campus, you will need to connect to the Virginia Tech VPN in order to access ARC's systems.
3. **Is the hostname correct?** Please check the name of the login node(s) for the system you are trying to access. For example, for login to *TinkerCliffs*, the hostname is not `tinkercliffs.arc.vt.edu` but rather `tinkercliffs1.arc.vt.edu` or `tinkercliffs2.arc.vt.edu`.
4. **Do you have an account?** You must request an account on a system before you can log in.
5. **Is there a maintenance outage?** ARC systems are occassionally taken offline for maintenance purposes. Users are typically notified via email well ahead of maintenance outages.

If you have checked all of the above and are still not sure why you cannot log in, please submit a help ticket.

5.2.2 How much does it cost to use ARC's systems?

ARC's systems are free to use, within limits. This means that Virginia Tech researchers can simply request an account to get access and run. Usage beyond fairly restrictive personal limits does require an approved *allocation* requested by a faculty member or project principal investigator; this requires some basic information to be provided, but getting an allocation does not require monetary payment of any kind. Researchers who need access to more resources beyond what we provide for free or who would like to purchase dedicated hardware can do so through our *Cost Center* or *Investment* programs. More information on how to get started with ARC is *here*.

5.2.3 Why is my job not starting?

Typically the squeue command will provide the reason a job isn't starting. This shows information about all pending or queued jobs, so it may be helpful to query for only your own jobs `squeue -u <your pid>` or only for a particular job `squeue -j <jobid>`. For example:

```
[brownm12@calogin2 ~]$ squeue -u brownm12
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)
310926 normal_q bash brownm12 PD 0:00 64 (PartitionNodeLimit)
```

This job has been submitted with a request for 64 nodes which exceeds the per-job limit on the `normal_q` partition.

Other common reasons:

Reason	Meaning
`Priority` or `Resources`	These two are the most common reasons given for a job being pending (PD). They simply mean that the job is waiting in the queue for resources to become available.
`QOSMaxJobsPerUserLimit`	QOS applied to the partition restricts users to a maximum number of concurrent running jobs. As your jobs complete, queued jobs will be allowed to start.
`QOSMaxCpuMinutesPerJobLimit`	QOS applied to the partition restricts jobs to a maximum number of CPU-minutes. To run, the job must request either fewer CPUs or less time.
`PartitionTimeLimit`	Requested timelimit exceeds the maximum for the partition
`AssocGrpBillingMinutes`	The *allocation* to which your submitted the job has exceeded its available resources (e.g., in the *free tier*)

5.2.4 Why can't I run on the login node?

One of the most common beginner mistakes on compute clusters is to log into the cluster and then immediately start running a computation. When you log into a cluster, you land on a *login node*. Login nodes are individual computers that represent a very small segment of the overall cluster and, crucially, are shared by *many* of the users who are logged into the cluster at a given time. So while basic tasks (editing files, checking jobs, perhaps making simple plots or compiling software) are fine to do on the login nodes, when you run a computationally-intensive task on the login node, you are adversely impacting other users (since the node is shared) while getting worse performance for yourself (by not using the bulk of the cluster). You should therefore submit your computationally intensive tasks to compute nodes by submitting a job to the scheduler. See *here* for documentation about job submission; we also have a video tutorial that will walk you through the process in a few minutes. Users who run problematic programs on the login node can have those tasks killed without warning. Users who repeatedly violate this policy arc subject to having their ARC account suspended.

5.2.5 When will my job start?

Adding the `--start` flag to `squeue` will provide the system's best guess as to when the job will start, or give a reason for why the job will not start in the `NODELIST(REASON)` column. If no estimated start time is provided, please see Why is my job not starting? for more information.

5.2.6 How do I submit an interactive job?

A user can request an interactive session on a compute node (e.g., for debugging purposes), using `interact`, a wrapper on `srun`. By default, this script will request one core (with one GPU on Infer) for one hour on a default partition (often `interactive_q` or `dev_q`, depending on the cluster). An allocation should be provided:

```
interact -A yourallocation
```

The request can be customized with standard job submission flags used by `srun` or `sbatch`. Examples include:

- Request two hours:

  ```
  interact -A yourallocation -t 2:00:00
  ```

- Request two hours on the `normal_q` partition:

```
interact -A yourallocation -t 2:00:00 -p normal_q
```

- Request two hours on one core and one GPU on Infer's t4_dev_q:

```
interact -A yourallocation -t 2:00:00 -p t4_dev_q -n 1 --gres=gpu:1
```

- Get additional details on what interact is doing:

```
interact -A yourallocation --verbose
```

(The flags for requesting resources may vary from system to system; please see the documentation for the system that you want to use.)

Once the job has been submitted, the system may print out some information about the defaults that interact has chosen. Once the resources requested are available, you will then get a prompt on a compute node. You can issue commands on the compute node as you would on the login node or any other system. To exit the interactive session, simply type exit.

Note: As with any other job, if all resources on the requested queue are being used by running jobs at the time an interactive job is submitted, it may take some time for the interactive job to start.

5.2.7 How do I change a job's stack size limit?

If your MPI code needs higher stack sizes then you may specify the stack size in the command that you specify to MPI. For example:

```
mpirun -bind-to-core -np $SLURM_NTASKS /bin/bash -c ulimit -s unlimited; ./your_program
```

5.2.8 How do I check my job's resource usage?

The jobload command will report core and memory usage for each node of a given job. Example output is:

```
[jkrometi@tinkercliffs2 04/06 09:21:13 ~]$ jobload 129722
Basic job information:
     JOBID       PARTITION         NAME      ACCOUNT       USER     STATE          TIME ␣
↪TIME_LIMIT  NODES NODELIST(REASON)
    129722         normal_q tinkercliffs       someaccount    someuser   RUNNING       ␣
↪43:43      8:00:00      2 tc[082-083]

Job is running on nodes: tc082 tc083

Node utilization is:
    node  cores    load    pct      mem     used    pct
   tc082    128   128.0  100.0  251.7GB  182.1GB   72.3
   tc083    128    47.9   37.4  251.7GB  187.2GB   74.3
```

This TinkerCliffs job is using all 128 cores on one node but only 48 cores on the second node. In this case, we know that the job has requested two full nodes, so some optimization may be in order to ensure that they're using all of the requested resources. The job is, however, using 70-75% memory on both nodes, so the resource request may be intentional. If more information is required about a given node, the command scontrol show node tc083 can provide it.

5.2.9 How can I monitor GPU utilization during my job?

The `nvidia-smi` command with no other options diplays this information but prints to standard out and only once. But there are many options which can be added to tap into lots of extended functionality of this tool.

Add a line like this to a batch script prior to starting training:

```
nvidia-smi --query-gpu=timestamp,name,pci.bus_id,driver_version,temperature.gpu,
↪utilization.gpu,utilization.memory,memory.total,memory.free,memy.used --format=csv -l␣
↪3 > $SLURM_JOBID.gpu.log &
```

The & causes the query to run in the background and keep running until the job ends or this process is manually killed. The > `$SLURM_JOBID.gpu.log` causes the output to be redirected to a file whose name is the numerical job id followed by `.gpu.log`.

The `-l 5` is for the repeating polling interval. From the `nvidia-smi` manual:

```
-l SEC, --loop=SEC
    Continuously report query data at the specified interval, rather than the default of␣
↪just once.
```

For details on query options: `nvidia-smi --help-query-gpu`

Output from `nvidia-smi` run as above looks like this:

```
2021/10/29 16:36:30.047, A100-SXM-80GB, 00000000:CB:00.0, 460.73.01, 41, 0 %, 0 %, 81251␣
↪MiB, 81248 MiB, 3 MiB
2021/10/29 16:36:33.048, A100-SXM-80GB, 00000000:07:00.0, 460.73.01, 58, 16 %, 4 %,␣
↪81251 MiB, 66511 MiB, 14740 MiB
2021/10/29 16:36:33.053, A100-SXM-80GB, 00000000:CB:00.0, 460.73.01, 41, 0 %, 0 %, 81251␣
↪MiB, 81248 MiB, 3 MiB
2021/10/29 16:36:36.054, A100-SXM-80GB, 00000000:07:00.0, 460.73.01, 65, 98 %, 15 %,␣
↪81251 MiB, 66571 MiB, 14680 MiB
2021/10/29 16:36:36.055, A100-SXM-80GB, 00000000:CB:00.0, 460.73.01, 41, 0 %, 0 %, 81251␣
↪MiB, 81248 MiB, 3 MiB
2021/10/29 16:36:39.055, A100-SXM-80GB, 00000000:07:00.0, 460.73.01, 67, 100 %, 36 %,␣
↪81251 MiB, 66571 MiB, 14680 MiB
2021/10/29 16:36:39.056, A100-SXM-80GB, 00000000:CB:00.0, 460.73.01, 41, 0 %, 0 %, 81251␣
↪MiB, 81248 MiB, 3 MiB
2021/10/29 16:36:42.057, A100-SXM-80GB, 00000000:07:00.0, 460.73.01, 54, 10 %, 2 %,␣
↪81251 MiB, 66571 MiB, 14680 MiB
2021/10/29 16:36:42.058, A100-SXM-80GB, 00000000:CB:00.0, 460.73.01, 41, 0 %, 0 %, 81251␣
↪MiB, 81248 MiB, 3 MiB
2021/10/29 16:36:45.059, A100-SXM-80GB, 00000000:07:00.0, 460.73.01, 54, 0 %, 0 %, 81251␣
↪MiB, 66571 MiB, 14680 MiB
2021/10/29 16:36:45.060, A100-SXM-80GB, 00000000:CB:00.0, 460.73.01, 41, 0 %, 0 %, 81251␣
↪MiB, 81248 MiB, 3 MiB
2021/10/29 16:36:48.060, A100-SXM-80GB, 00000000:07:00.0, 460.73.01, 68, 100 %, 26 %,␣
↪81251 MiB, 66571 MiB, 14680 MiB
2021/10/29 16:36:48.061, A100-SXM-80GB, 00000000:CB:00.0, 460.73.01, 41, 0 %, 0 %, 81251␣
↪MiB, 81248 MiB, 3 MiB
2021/10/29 16:36:51.062, A100-SXM-80GB, 00000000:07:00.0, 460.73.01, 52, 20 %, 3 %,␣
↪81251 MiB, 66571 MiB, 14680 MiB
2021/10/29 16:36:51.063, A100-SXM-80GB, 00000000:CB:00.0, 460.73.01, 41, 0 %, 0 %, 81251␣
↪MiB, 81248 MiB, 3 MiB
```

(continues on next page)

(continued from previous page)

```
2021/10/29 16:36:54.064, A100-SXM-80GB, 00000000:07:00.0, 460.73.01, 52, 0 %, 0 %, 81251␣
↪MiB, 66571 MiB, 14680 MiB
```

You can monitor the utilization information in near-real-time from a login node by navigating to the output directory for the job and using `tail` to follow the output with `tail -f <jobid>.gpu.log` and the CSV formatting makes it easy to analyze or generate graphics with other tools such as `python`, `R`, or `matlab`.

5.2.10 I need a software package for my research. Can you install it for me?

At any given time, ARC staff is trying to balance many high-priority tasks to improve, refine, or augment our systems. Unfortunately, this means that we typically cannot install all or even most of the software that our users require to do their research. As a result, the set of applications on each system does not typically change unless a new software package is requested by a large number of users. However, users are welcome to install software that they require for their research in their Home directory. This generally involves copying the source code into one of your personal or group storage locations and then following the directions provided with the software to build that source code into an executable. If the vendor does not provide source code and just provides an executable (which is true of some commercial software packages), then you need to select the right executable for the system hardware and copy that into your home directory. ARC provides a script called *setup_app* that helps automate setup of directories and creation of personal modules.

5.2.11 How can I add my own software installation to my module system?

The key is to create a modulefile for the software and make sure that it is in a location that can be found by `MODULEPATH`. Starting on TinkerCliffs and later systems, ARC provides a script called `setup_app` that automates much of this process. See also this video tutorial. Start by providing a software package and version, e.g.,

```
[jkrometi@tinkercliffs2 ~]$ setup_app julia 1.6.1-foss-2020b
Create directories /home/jkrometi/apps/tinkercliffs-rome/julia/1.6.1-foss-2020b and /
↪home/jkrometi/easybuild/modules/tinkercliffs-rome/all/julia?
```

Enter `y` to let it proceed. The script will then set up the directory and the modulefile. It finishes by printing some information about what you need to do to finish off the install:

```
Done. To finish your build:
 1. Install your app in /home/jkrometi/apps/tinkercliffs-rome/julia/1.6.1-foss-2020b/
 2. Edit the modulefile in /home/jkrometi/easybuild/modules/tinkercliffs-rome/all/julia/
↪1.6.1-foss-2020b.lua
    - Set or remove modules to load in the load() line
    - Edit description and URL
    - Check the variable names
    - Edit paths (some packages don't have, e.g., an include/)

Note: You may need to refresh the cache, e.g.,
  module --ignore_cache spider julia
to find the module the first time.
```

Note that `setup_app` also provides a `--base` flag that will allow installation somewhere other than the default location, e.g.,

```
setup_app --base=/projects/myproject julia 1.6.1-foss-2020b
```

5.2.12 What is the best way to make sure everyone in my group has the same access to all the files in our shared directory?

The first step is to make sure the group id (GID) of all the files and directories are consistent and match the group id of the shared directory. The `chgrp` command does this but only the owner of a file can change its gid. So each member of the group needs to find files which they own and `chgrp` them to correct the GID and also `chmod` them to ensure correct mode. Here is a template command sequence to do that:

```
# Show numeric group id of current user. This is the GID which will be used in the next␣
↪step to identify files
id -g
# Find files in the shared directory matching current user's GID and execute a chgrp on␣
↪them
find /projects/MYGROUPNAME -gid `id -g` -exec chgrp arc.MYGROUPNAME {} \;
# Find files in the shared directory matching current user's UID and execute a chmod on␣
↪them to all group members to have read access
find /projects/MYGROUPNAME -uid `id -u` -exec chmod g+r arc.MYGROUPNAME {} \;
```

Any member of the group who has files in the shared directory with their GID will need to run that command. Group ownership of files in the shared directories is inherited for newly created files and for files transferred with `rsync` with the correct options, but `scp` generally does not respect the parent gid, unfortunately.

5.2.13 What does a "Disk quota exceeded" error mean?

This typically means that one of your *storage locations* has exceeded the maximum allowable size. You will need to reduce the space consumed in order to run jobs successfully again. Note that the quota system for Project and Work storage on *TinkerCliffs* and *Infer* can be counterintuitive in some ways, so if you are getting a "quota exceeded" error on those file systems and think you should not be, see *this description* for details and fixes.

5.2.14 What does a `module: command not found` error mean?

If your job returns an error that looks like

```
/cm/local/apps/slurm/var/spool/job275621/slurm_script: line 11: module: command not found
```

then you are likely hitting a race condition during job startup. We are occassionally seeing this issue on *TinkerCliffs* but have been unable to identify a cause or tie it to specific nodes. When resubmitted, these jobs typically run without incident. However, you should be able to ensure that your job will not fail with this error by adding the following lines to your submission script before any commands (e.g., `module` commands) are run:

```
if [ -z ${HOME+x} ]; then
  export HOME=$(echo ~)
  source /etc/profile
  source /etc/bashrc
  source $HOME/.bashrc
fi
```

These lines will manually setup the environment should Slurm fail to do so.

5.2.15 What does a Detected 1 oom-kill event(s) error mean?

If your job fails with an error like

```
slurmstepd: error: Detected 1 oom-kill event(s)
```

then your job triggered Linux's Out of Memory Killer process. This means that it tried to use more memory than allocated to the job. The `seff` command (Slurm job efficiency) also provides some information on resource utilization:

```
[user@infer1 ~]$ seff 1447
Job ID: 1447
Cluster: infer
User/Group: someuser/someuser
State: OUT_OF_MEMORY (exit code 0)
Nodes: 2
Cores per node: 32
CPU Utilized: 02:43:59
CPU Efficiency: 1.56% of 7-07:21:36 core-walltime
Job Wall-clock time: 02:44:24
Memory Utilized: 174.83 GB
Memory Efficiency: 49.11% of 356.00 GB
```

If your job is requesting a subset of a node, you will need to request more cores (which will give you more memory). If you are already requesting a full node, you will need to either edit your code or problem to use less memory or submit to different hardware that has more memory (e.g., the high memory nodes on TinkerCliffs) – check the details for each cluster to find an option that might work for you.

5.2.16 Why are basic commands like `sbatch` not recognized?

Starting with Tinkercliffs and Infer, ARC provides a default set of modules that are automatically loaded when you log in. If basic commands like `sbatch` are not recognized, it is often because these default modules have been removed (e.g., via `module purge`). Please run `module reset` and see if that solves your problem.

5.2.17 How do I add a user to an allocation?

To add a user to *an existing allocation*, follow these steps:

1. Go to ColdFront. (You may be prompted for a password.)
2. You will see a list of your Projects. Click on the one you want to modify.
3. Scroll down to Users and select Add Users.
4. Under Search String enter the user's PID (or a list of PIDs) and click Search.
5. Scroll down, select the user whom you want to add, and click Add Selected Users to Project.
6. The page will refresh and the user's PID should be included in the Users table. They are now added to the project and its associated allocations.

5.2.18 How do I attach to my process for debugging?

Short Answer: Attaching to a process for debugging no longer requires any special steps on ARC resources.

Longer Answer: Debuggers like gdb make software development much more efficient. Attaching to a process for debugging requires that the targeted process and the user's current process be in the same group. When ARC used Moab and Torque for scheduling and resource management, processes launched by the scheduler were started under a group other than the user's group. Special steps were then required to switch groups before trying to attach with gdb. However, the Slurm scheduler now used by ARC launches processes under the user's group, so these steps are no longer required. You may simply `ssh` to the compute node where the process is running, look up the process ID (e.g., with `top` or `ps`), and then attach to it.

5.2.19 How can I submit a job that depends on the completion of another job?

Sometimes it may be useful to split one large computation into multiple jobs (e.g. due to queue limits), but submit those jobs all at once. Jobs can be made dependent on each other using the `--dependency=after:job_id` flag to `sbatch`. Additional dependency options can be found in the documentation for `sbatch`. For example, here we submit three jobs, each of which depends on the preceding one:

```
[johndoe@tinkercliffs2 ~]$ sbatch test.sh
Submitted batch job 126448
[johndoe@tinkercliffs2 ~]$ sbatch --dependency=after:126448 test.sh
Submitted batch job 126449
[johndoe@tinkercliffs2 ~]$ sbatch --dependency=after:126449 test.sh
Submitted batch job 126450
```

The first job starts right away, but the second doesn't start until the first one finishes and the third job doesn't start until the second one finishes. This allows the user to split their job up into multiple pieces, submit them all right away, and then just monitor them as they run one after the other to completion.

5.2.20 How can I run multiple serial tasks inside one job?

Users with serial (sequential) programs may want to package multiple serial tasks into a single job submitted to the scheduler. This can be done with third-party tools (gnu parallel is a good one) or using a loop within the job submission script. (A similar structure can be used to *run multiple short, parallel tasks inside a job*.) The basic structure is to loop through the number of tasks using while or for, start the task in the background using the & operator, and then use the wait command to wait for the tasks to finish:

```
    # Define variables
    numtasks=16
    np=1
    # Loop through numtasks tasks
    while [ $np -le $numtasks ]
    do
      # Run the task in the background with input and output depending on the variable np
      ./a.out $np > $np.out &

      # Increment task counter
      np=$((np+1))
    done

    # Wait for all of the tasks to finish
    wait
```

Please note that the above structure will only work within a single node. To ensure that the same program (with the same inputs) isn't being run multiple times, users should make sure that the loop variable (np, above) is used to specify input files or parameters.

5.2.21 How can I run multiple short, parallel tasks inside one job?

Sometimes users have a parallel application that runs quickly, but that they need to run many times. In this case, it may be useful to package multiple parallel runs into a single job. This can be done using a loop within the job submission script. An example structure:

```
# Specify the list of tasks
    tasklist=task1 task2 task3

    # Loop through the tasks
    for tsk in $tasklist; do
      # run the task $tsk
      mpirun -np $SLURM_NTASKS ./a.out $tsk
    done
```

To ensure that the same program (with the same inputs) isn't being run multiple times, users should make sure that the loop variable (tsk, above) is used to specify input files or parameters. Note that, unlike when *running multiple serial tasks at once*, in this case each task will not start until the previous one has finished.

5.3 Software Modules

ARC uses the lmod environment modules system to enable access to centrally-installed (ARC-maintained) scientific software packages. This provides for the dynamic modification of a user's environment for an application or set of applications, enabling streamlined management of software versions and dependencies.

The modules on ARC's systems fall into two categories depending on when the cluster was deployed:

- *EasyBuild*: ARC systems deployed in 2020 or later (*TinkerCliffs* and *Infer*) mostly rely on EasyBuild for module deployment.
- *Hierarchical*: ARC systems deployed prior to 2019 use a hierarchical module structure.

These two systems are described in the following sections.

5.3.1 EasyBuild

Newer (2020 and later) ARC clusters use a module system mostly built around EasyBuild, a software build and installation framework that allows you to manage (scientific) software on High Performance Computing (HPC) systems in an efficient way. EasyBuild is maintained by a broad user community and makes it easier for ARC to provide stable, performant, and updated scientific software. It also makes it trivial in some cases for users to install their own versions of packages if they so desire.

Toolchains

EasyBuild is built around toolchains, which describe the sequence of dependencies, such as compiler, linear algebra library, and MPI implementation, used to build packages. There are two main ones:

- `foss` ("Free Open Source Software"): GCC compilers, OpenBLAS for linear algebra, OpenMPI for MPI, etc
- `intel`: Intel compilers, Intel MKL for linear algebra, Intel MPI

However, we have upon request supported others, such as:

- `iomkl`: Intel compilers, Intel MKL for linear algebra, and OpenMPI for MPI
- `gomkl`: GCC compilers, Intel MKL for linear algebra, and OpenMPI for MPI

So please reach out if the toolchains that we provide are not what you need.

Toolchains are typically updated twice per year (`a` and `b` versions) and we try to stay up-to-date with those updates.

As an example, the modules active after loading the `foss/2020b` toolchain are (note that the first few modules in the list are defaults provided by ARC):

```
[arcuser@tinkercliffs2 ~]$ module reset; module load foss/2020b; module list
Resetting modules to system default

Currently Loaded Modules:
  1) shared                              8) useful_scripts                  15) XZ/5.2.5-
↪GCCcore-10.2.0          22) PMIx/3.1.5-GCCcore-10.2.0
  2) slurm/20.02.3                       9) DefaultModules                  16) libxml2/2.
↪9.10-GCCcore-10.2.0     23) OpenMPI/4.0.5-GCC-10.2.0
  3) apps                               10) GCCcore/10.2.0                  17)␣
↪libpciaccess/0.16-GCCcore-10.2.0  24) OpenBLAS/0.3.12-GCC-10.2.0
  4) site/tinkercliffs/easybuild/setup  11) zlib/1.2.11-GCCcore-10.2.0     18) hwloc/2.2.
↪0-GCCcore-10.2.0        25) gompi/2020b
  5) cray                               12) binutils/2.35-GCCcore-10.2.0   19) libevent/
↪2.1.12-GCCcore-10.2.0    26) FFTW/3.3.8-gompi-2020b
  6) craype-x86-rome                    13) GCC/10.2.0                      20) UCX/1.9.0-
↪GCCcore-10.2.0          27) ScaLAPACK/2.1.0-gompi-2020b
  7) craype-network-infiniband          14) numactl/2.0.13-GCCcore-10.2.0  21) libfabric/
↪1.11.0-GCCcore-10.2.0    28) foss/2020b
```

We see here that lower-level software (e.g., `binutils`) is also included in the module structure, reducing the risk of conflicts in adding new versions later.

Usage

In this section we will describe how to use EasyBuild's module system. We will use Gromacs as our example software. We begin by noting that, even though Gromacs is a popular software package on HPC systems, upon login its executable `gmx` is nowhere to be found:

```
[arcuser@tinkercliffs2 ~]$ which gmx
/usr/bin/which: no gmx in (/apps/useful_scripts/bin:/cm/shared/apps/slurm/20.02.3/sbin:/
↪cm/shared/apps/slurm/20.02.3/bin:/home/arcuser/util:/usr/local/bin:/usr/bin:/usr/local/
↪sbin:/usr/sbin:/opt/ibutils/bin:/usr/share/lmod/lmod/libexec)
```

To find it, we need to load the Gromacs module. To find a software package, you can use `module spider`. For example:

```
[arcuser@tinkercliffs2 ~]$ module spider gromacs

-----------------------------------------------------------------------------------------
↪-----------------------------------------------------------------
  GROMACS:
-----------------------------------------------------------------------------------------
↪-----------------------------------------------------------------
    Description:
      GROMACS is a versatile package to perform molecular dynamics, i.e. simulate the␣
↪Newtonian equations of motion for systems with hundreds to millions
      of particles. This is a CPU only build, containing both MPI and threadMPI builds␣
↪for both single and double precision. It also contains the gmxapi
      extension for the single precision MPI build.

     Versions:
        GROMACS/2020.1-foss-2020a-Python-3.8.2
        GROMACS/2020.3-foss-2020a-Python-3.8.2

-----------------------------------------------------------------------------------------
↪-----------------------------------------------------------------
  For detailed information about a specific "GROMACS" module (including how to load the␣
↪modules) use the module's full name.
  For example:

     $ module spider GROMACS/2020.3-foss-2020a-Python-3.8.2
-----------------------------------------------------------------------------------------
↪-----------------------------------------------------------------
```

Note: You can also use `module avail` to list all modules, although the output is quite long. We provide it *here*, in case it helps you find what you need.

To then load the module, you can use `module load`:

```
[arcuser@tinkercliffs2 ~]$ module reset; module load GROMACS/2020.3-foss-2020a-Python-3.
↪8.2
Resetting modules to system default
```

We can use `module list` to list the modules we have loaded now:

```
[arcuser@tinkercliffs2 ~]$ module list

Currently Loaded Modules:
  1) shared                             14) numactl/2.0.13-GCCcore-9.3.0     27) ncurses/
↪6.2-GCCcore-9.3.0
  2) slurm/20.02.3                      15) XZ/5.2.5-GCCcore-9.3.0           28)␣
↪libreadline/8.0-GCCcore-9.3.0
  3) apps                               16) libxml2/2.9.10-GCCcore-9.3.0     29) Tcl/8.6.
↪10-GCCcore-9.3.0
  4) site/tinkercliffs/easybuild/setup  17) libpciaccess/0.16-GCCcore-9.3.0  30) SQLite/
↪3.31.1-GCCcore-9.3.0
  5) cray                               18) hwloc/2.2.0-GCCcore-9.3.0        31) GMP/6.2.
↪0-GCCcore-9.3.0
```

(continues on next page)

(continued from previous page)

```
  6) craype-x86-rome                     19) UCX/1.8.0-GCCcore-9.3.0          32) libffi/
↪3.3-GCCcore-9.3.0
  7) craype-network-infiniband           20) OpenMPI/4.0.3-GCC-9.3.0          33) Python/
↪3.8.2-GCCcore-9.3.0
  8) useful_scripts                      21) OpenBLAS/0.3.9-GCC-9.3.0         34)␣
↪pybind11/2.4.3-GCCcore-9.3.0-Python-3.8.2
  9) DefaultModules                      22) gompi/2020a                      35) SciPy-
↪bundle/2020.03-foss-2020a-Python-3.8.2
 10) GCCcore/9.3.0                       23) FFTW/3.3.8-gompi-2020a           36)␣
↪networkx/2.4-foss-2020a-Python-3.8.2
 11) zlib/1.2.11-GCCcore-9.3.0           24) ScaLAPACK/2.1.0-gompi-2020a      37) GROMACS/
↪2020.3-foss-2020a-Python-3.8.2
 12) binutils/2.34-GCCcore-9.3.0         25) foss/2020a
 13) GCC/9.3.0                           26) bzip2/1.0.8-GCCcore-9.3.0
```

We can see that the system now can find the Gromacs gmx executable:

```
[arcuser@tinkercliffs2 ~]$ which gmx
/apps/easybuild/software/tinkercliffs-rome/GROMACS/2020.3-foss-2020a-Python-3.8.2/bin/gmx
```

Finally, to clear out modules, we recommend using `module reset`, which will return the modules to their default state:

```
[arcuser@tinkercliffs2 ~]$ module reset; module list
Resetting modules to system default

Currently Loaded Modules:
  1) shared          3) apps                             5) cray              7)␣
↪craype-network-infiniband   9) DefaultModules
  2) slurm/20.02.3   4) site/tinkercliffs/easybuild/setup   6) craype-x86-rome   8)␣
↪useful_scripts
```

Warning: Do not use `module purge`. As you see above, ARC includes a number of important packages, such as the *Slurm scheduler* in the default modules. `module purge` will remove those, too, *breaking key functionality*. If you accidentally use `module purge`, simply use `module reset` to reset to the default.

Using EasyBuild to Build Your Own Software

EasyBuild can also be used by users to install packages. We describe the steps briefly below; see also our *video tutorial* on the subject.

The basic steps are:

1. Load the EasyBuild module to get access to the eb executable:

   ```
   module reset; module load EasyBuild
   ```

2. Use `eb -S` to search for the software package that you need (the output is quite long in this case so we only show a snippet):

   ```
   [arcuser@tinkercliffs2 ~]$ eb -S ^netCDF
    * $CFGS3/n/netCDF/netCDF-4.7.1-iimpi-2019b.eb
   ```

(continues on next page)

(continued from previous page)

```
 * $CFGS3/n/netCDF/netCDF-4.7.1-iimpic-2019b.eb
 * $CFGS3/n/netCDF/netCDF-4.7.4-fix-mpi-info-f2c.patch
 * $CFGS3/n/netCDF/netCDF-4.7.4-gompi-2020a.eb
 * $CFGS3/n/netCDF/netCDF-4.7.4-gompi-2020b.eb
 * $CFGS3/n/netCDF/netCDF-4.7.4-gompic-2020a.eb
```

3. Pick one of the versions and use `eb -Dr filename.eb` to see what it is going to do (the `D` in this case is for "dry run"). The `[x]` lines indicate packages that are already installed. The `[ ]` lines are packages that will need to be installed.

```
[arcuser@tinkercliffs2 ~]$ eb -Dr netCDF-4.7.4-gompi-2020b.eb
== Temporary log file in case of crash /localscratch/eb-ceKHhw/easybuild-asf_l0.log
== found valid index for /apps/easybuild/software/tinkercliffs-rome/EasyBuild/4.4.0/
↪easybuild/easyconfigs, so using it...
== found valid index for /apps/easybuild/software/tinkercliffs-rome/EasyBuild/4.4.0/
↪easybuild/easyconfigs, so using it...
Dry run: printing build status of easyconfigs and dependencies
CFGS=/apps/easybuild
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/M4/M4-1.4.18.eb (module: M4/1.4.18)
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/Bison/Bison-3.7.1.eb (module: Bison/3.7.
↪1)
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/bzip2/bzip2-1.0.8-GCCcore-10.2.0.eb␣
↪(module: bzip2/1.0.8-GCCcore-10.2.0)
 * [ ] $CFGS/software/tinkercliffs-rome/EasyBuild/4.4.0/easybuild/easyconfigs/l/
↪libiconv/libiconv-1.16-GCCcore-10.2.0.eb (module: libiconv/1.16-GCCcore-10.2.0)
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/expat/expat-2.2.9-GCCcore-10.2.0.eb␣
↪(module: expat/2.2.9-GCCcore-10.2.0)
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/CMake/CMake-3.18.4-GCCcore-10.2.0.eb␣
↪(module: CMake/3.18.4-GCCcore-10.2.0)
 * [ ] $CFGS/software/tinkercliffs-rome/EasyBuild/4.4.0/easybuild/easyconfigs/d/
↪Doxygen/Doxygen-1.8.20-GCCcore-10.2.0.eb (module: Doxygen/1.8.20-GCCcore-10.2.0)
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/libevent/libevent-2.1.12-GCCcore-10.2.0.
↪eb (module: libevent/2.1.12-GCCcore-10.2.0)
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/numactl/numactl-2.0.13-GCCcore-10.2.0.
↪eb (module: numactl/2.0.13-GCCcore-10.2.0)
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/OpenMPI/OpenMPI-4.0.5-GCC-10.2.0.eb␣
↪(module: OpenMPI/4.0.5-GCC-10.2.0)
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/gompi/gompi-2020b.eb (module: gompi/
↪2020b)
 * [x] $CFGS/ebfiles_repo/tinkercliffs-rome/HDF5/HDF5-1.10.7-gompi-2020b.eb␣
↪(module: HDF5/1.10.7-gompi-2020b)
 * [ ] $CFGS/software/tinkercliffs-rome/EasyBuild/4.4.0/easybuild/easyconfigs/n/
↪netCDF/netCDF-4.7.4-gompi-2020b.eb (module: netCDF/4.7.4-gompi-2020b)
== Temporary log file(s) /localscratch/eb-ceKHhw/easybuild-asf_l0.log* have been␣
↪removed.
== Temporary directory /localscratch/eb-ceKHhw has been removed.
```

4. If you are okay with installing the packages marked with `[ ]`, you can install them with `eb -r filename.eb` (i.e., remove the `D` for "dry run" from the previous command):

```
[arcuser@tinkercliffs2 ~]$ eb -r netCDF-4.7.4-gompi-2020b.eb
== Temporary log file in case of crash /localscratch/eb-lsT7pO/easybuild-zdQblI.log
```

(continues on next page)

(continued from previous page)

```
== found valid index for /apps/easybuild/software/tinkercliffs-rome/EasyBuild/4.4.0/
↪easybuild/easyconfigs, so using it...
== found valid index for /apps/easybuild/software/tinkercliffs-rome/EasyBuild/4.4.0/
↪easybuild/easyconfigs, so using it...
== resolving dependencies ...
== processing EasyBuild easyconfig /apps/easybuild/software/tinkercliffs-rome/
↪EasyBuild/4.4.0/easybuild/easyconfigs/l/libiconv/libiconv-1.16-GCCcore-10.2.0.eb
== building and installing libiconv/1.16-GCCcore-10.2.0...
== fetching files...
== creating build dir, resetting environment...
== unpacking...
== patching...
== preparing...
== configuring...
== building...
== testing...
== installing...
```

This process can be time-consuming depending on the package, so it may be worth starting it in, e.g., a `screen` session. If the process ultimately completes with a line that looks like

```
== COMPLETED: Installation ended successfully
```

then you have successfully installed your software package. It can then be loaded from the module system like any other module. In this case, we would use

```
module reset; module load netCDF/4.7.4-gompi-2020b
```

where we get the module name by converting the first `-` in the `.eb` file name to a `/` or by noting that EasyBuild printed the following during installation:

```
== building and installing netCDF/4.7.4-gompi-2020b...
```

Environment variables

Sometimes it is important to know where a software package is installed so that you can point to it when installing other software. For this purpose, EasyBuild provides `$EBROOTSOFTWARE` to point to the software installation location. For example:

```
[arcuser@tinkercliffs2 ~]$ module reset; module load netCDF/4.7.4-gompi-2020a
Resetting modules to system default
[arcuser@tinkercliffs2 ~]$ ls $EBROOTNETCDF
bin  easybuild  include  lib64  share
```

So to link to NetCDF libraries, one might use `-L$EBROOTNETCDF/lib64`.

5.3.2 Hierarchical

Structure

Modules on ARC systems are based on a hierarchical structure where the modules that are available in one level of the hierarchy depend on the modules loaded from the previous level. This ensures that users do not inadvertently select module combinations that are incompatible and/or give inferior performance. The module levels are:

1. **Compiler:** Users first select the compiler that they want to use.
2. **MPI Stack:** Users then select the MPI stack that they want to use. MPI stack availability depends on the compiler that is loaded.
3. **High Level Software:** Once a user has selected both a compiler and an MPI stack, they can load higher-level software built against that compiler and MPI stack.

Please consult the software documentation for each system to determine that system's default compiler and MPI stack. Note that the default compiler and MPI stack are automatically loaded, so if a user wishes to use the system defaults for each, they can simply start loading higher-level modules as soon as they log in. If not, the user may use the module swap command to replace one module with another or the module purge command to remove all modules and then load the modules that they want.

Usage

To change or view modules, the module command is used. The most common subcommands are: - View a list of available modules (depends on the currently loaded modules):

```
module avail
```

- List all possible modules with the name modulename:

```
module spider modulename
```

- Print information about the modulename module, such as what the software package is, what environment variables and paths it sets, and what its dependencies are:

```
module show modulename
```

- View a list of modules currently loaded in your environment:

```
module list
```

- Add a module to your environment with one of the following:

```
module add modulename
module load modulename
```

- Remove a module from your environment with one of:

```
module rm modulename
module unload modulename
```

- Replace module1 with module2 in your environment. Any dependent modules in the module tree will be reloaded to reflect the change.

```
module swap module1 module2
```

- Remove all modules from your environment:

```
module purge
```

The module command can be used at the command line and within job launch scripts.

Loading Software

The most basic module usage would be loading the Intel compiler and the HDF5 data management library built against it:

```
module purge            #Make sure no modules are loaded
module load intel/18.2  #Load intel compiler
module load hdf5/1.8.17 #Load HDF5 (built against the intel compiler)
module list             #Print currently loaded modules
```

We see that an Intel module and an HDF5 module are loaded:

```
Currently Loaded Modules:
  1) intel/18.2   2) hdf5/1.8.17
```

Now the system knows where the `h5cc` binary is located:

```
[arcuser@calogin2 ~]$ which h5cc
/opt/apps/intel18_2/hdf5/1.8.17/bin/h5cc
```

Finding a Software Package

To see what versions of the molecular dynamics software gromacs are installed, use:

```
module spider gromacs
```

In this case, we see that version 5.1.2 is available:

```
  -------------------------------------------------------------------------
  gromacs:
  -------------------------------------------------------------------------
    Description:
      GROMACS

     Versions:
        gromacs/5.1.2

  -------------------------------------------------------------------------
  For detailed information about a specific "gromacs" module (including how to load the␣
↪modules) use the module's full name.
  For example:

     $ module spider gromacs/5.1.2
  -------------------------------------------------------------------------
```

To see how to access gromacs version 5.1.2:

```
module spider gromacs/5.1.2
```

We see that it is built against several compiler/MPI stack combinations:

```
  ------------------------------------------------------------------------
  gromacs: gromacs/5.1.2
  ------------------------------------------------------------------------
    Description:
      GROMACS

    You will need to load all module(s) on any one of the lines below before the
↪"gromacs/5.1.2" module is available to load.

      gcc/5.2.0  mvapich2/2.2
      gcc/5.2.0  openmpi/3.0.0
      gcc/5.2.0  openmpi/3.1.2
      gcc/6.1.0  openmpi/3.0.0
      gcc/6.1.0  openmpi/3.1.2
      intel/15.3  mvapich2/2.2
      intel/15.3  openmpi/3.0.0
      intel/15.3  openmpi/3.1.2
      intel/18.2  openmpi/3.0.0

    Help:
      GROMACS is a versatile and extremely well optimized package to perform
      molecular dynamics computer simulations and subsequent trajectory analysis.

      Define Environment Variables:

                  $GROMACS_DIR - root
                  $GROMACS_BIN - binaries
                  $GROMACS_INC - includes
                  $GROMACS_LIB - libraries

      Prepend Environment Variables:
```

So we can load one of them (it turns out that `fftw` is also required to load the module, as you will see if you leave it out):

```
module purge; module load intel/18.2 openmpi/3.0.0 fftw/3.3.8 gromacs/5.1.2
```

And now the system knows where the gmx binary is:

```
[arcuser@calogin2 ~]$ which gmx
/opt/apps/intel18_2/openmpi3_0/gromacs/5.1.2/bin/gmx
```

5.4 Slurm Scheduler Interaction

Jobs are submitted to ARC resources through a job queuing system, or scheduler. Submission of jobs through a queueing system means that jobs may not run immediately, but will wait until the resources it requires are available. The queuing system thus keeps the compute servers from being overloaded and allocates dedicated resources across running jobs. This will allow each job to run optimally once it leaves the queue. ARC uses the Slurm scheduler; descriptions of common interactions with Slurm are provided below. For a more detailed Slurm user guide, check out SchedMD's online documentation and videos here: https://slurm.schedmd.com/tutorials.html. If you are familiar commands from another resource manager (e.g., Moab/PBS/Torque) and simply need to translate them to Slurm, see https://slurm.schedmd.com/rosetta.html.

5.4.1 Submission Script

Jobs are submitted with submission scripts that describe what resources the job requires and what the system should do once the job runs. Example submissions scripts are provided in the documentation for each system and can be used as a template for getting started. Note that jobs can also be started interactively, which can be very useful during testing and debugging. The resource requests are similar to PBS/Torque and include:

- **Partition** (denoted by `#SBATCH -p`). Indicates the partition (or queue) to which the job should be submitted. Different partitions are intended for different use cases (e.g., production, development, visualization) or hardware and therefore have different usage limits. The partition parameters are described in the documentation for each system.
- **Walltime** (denoted by `#SBATCH -t`). This is the time that you expect your job to run; so if you submit your job at 5:00pm on Wednesday and you expect it to finish at 5:00pm on Thursday, the walltime would be 24:00:00. Note that if your job exceeds the walltime estimated during submission, the scheduler will kill it. So it is important to be conservative (i.e., to err on the high side) with the walltime that you include in your submission script. Acceptable time formats include `minutes`, `minutes:seconds`, `hours:minutes:seconds`, `days-hours`,`days-hours:minutes` and `days-hours:minutes:seconds`.
- **Hardware** (denoted by `#SBATCH --gres=gpu:1`, `#SBATCH --mem=500G`, `#SBATCH --exclusive`, etc). This is the hardware that you want to reserve for your job. The types and quantity of available hardware, how to request them, and the limits for each user are described in the documentation for each system.
- **Account** (denoted by `#SBATCH --account=[allocation]`). Indicates the allocation account to which you want to charge the job. (Only applies to some systems - see system documentation.)

The submission script should also specify what should happen when the job runs:

- **Software Modules**. Use module commands to add the software modules that your job will need to run.
- **Run**. Finally, you need to specify what commands you want to run to execute your computation. This can be execution of your own program or a call to a software package.

As an example, the following is a basic hello world example.

```
#!/bin/bash
#SBATCH -J hello-world
#SBATCH -p normal_q
#SBATCH -N 1 --ntasks-per-node=1 --cpus-per-task=1 # this requests 1 node, 1 core.
#SBATCH -t 10:00 # 10 minutes
#SBATCH --gres=gpu:pascal:4
#SBATCH --account=test
#SBATCH --export=NONE # this makes sure the compute environment is clean
echo hello world
```

5.4.2 Job Management

To submit your job to the queuing system, use the command `sbatch`. For example, if your script is in JobScript.sh, the command would be:

```
sbatch ./JobScript.sh
```

This will return a message with your job id such as:

```
Submitted batch job 5123
```

Here 5123 is the job number. Once a job is submitted to a queue, it will wait until requested resources are available within that queue, and will then run if eligible. Eligibility to run is influenced by the resource policies in effect for the queue. To check a job's status, use the `squeue` command:

```
squeue -v 5123
```

To check the status of more than one job or the queues in general, use squeue. Examples include:

```
squeue --state=Running   #View all running jobs
squeue --users=username  #View only a given user's jobs
```

If your job has not started and you are unsure why, this FAQ provides some common explanations. To remove a job from the queue, or stop a running job, use the command `scancel`. For job number 5123, the command would be:

```
scancel 5123
```

5.4.3 Output

When your job has finished running, any outputs to stdout or stderr will be placed in a file in the directory where the job was submitted. For example, for a job submitted from JobScript.sh and with job ID 5123, the output would be in:

```
slurm-5123.out  # Output and errors will be here
```

This behavior can be modified using the `--output=` and `--error=` flags. Any files that the job writes to permanent storage locations will simply remain in those locations. Files written to locations only available during the life of the job (e.g. TMPFS or TMPDIR) will be removed once the job is completed, so those files must be moved to a permanent location at the end of the submission script.

5.5 Video Tutorials

ARC provides a number of video tutorials on our channel on video.vt.edu. In particular, the following sequence walks a user through the fundamentals of ARC usage in less than an hour:

5.5.1 Login

These videos will walk the user through accessing our systems for the first time (and streamlining access for subsequent logins):

- Login with SSH plus Using SSH Keys and Agent to simplify logins, and/or
- Open OnDemand

5.5.2 Accessing Software

The following videos will walk the user through accessing software that ARC has installed or through setting up your own packages:

- Using Modules to Access Scientific Software - EasyBuild (TinkerCliffs/Infer) version, and/or
- Using Modules to Access Scientific Software - Hierarchical (Pre-2020) version, and/or
- Creating Custom Software Modules with EasyBuild, and/or
- Manual Install of Custom Software Modules

5.5.3 Scheduler interaction (job submission)

The following will walk the user through the process of submitting interactive jobs for testing/development and batch jobs for production research runs:

- Interactive and Batch Jobs

Note that these videos require a VT Login to access. Also, each video has a table of contents that can be used to skip between sections; this can be accessed by clicking the "hamburger" (three horizontal bars) button at the top left of the video.

Contents:

- *Getting Started*: Basic information for people new to HPC or just new to ARC
- *Resources*: Descriptions of the hardware and services that we offer
- *Software*: Lists of and user guides for software installed on ARC systems
- *Usage*: Tutorials for how to use ARC systems
- *PI Information*: Key information for faculty members or project principal investigators (PIs)

To request help:

- Visit Office Hours
- Request a Consultation

Other key links:

- Create an ARC User Account
- *Video tutorials*
- *Frequently asked questions*