SHARCNET™

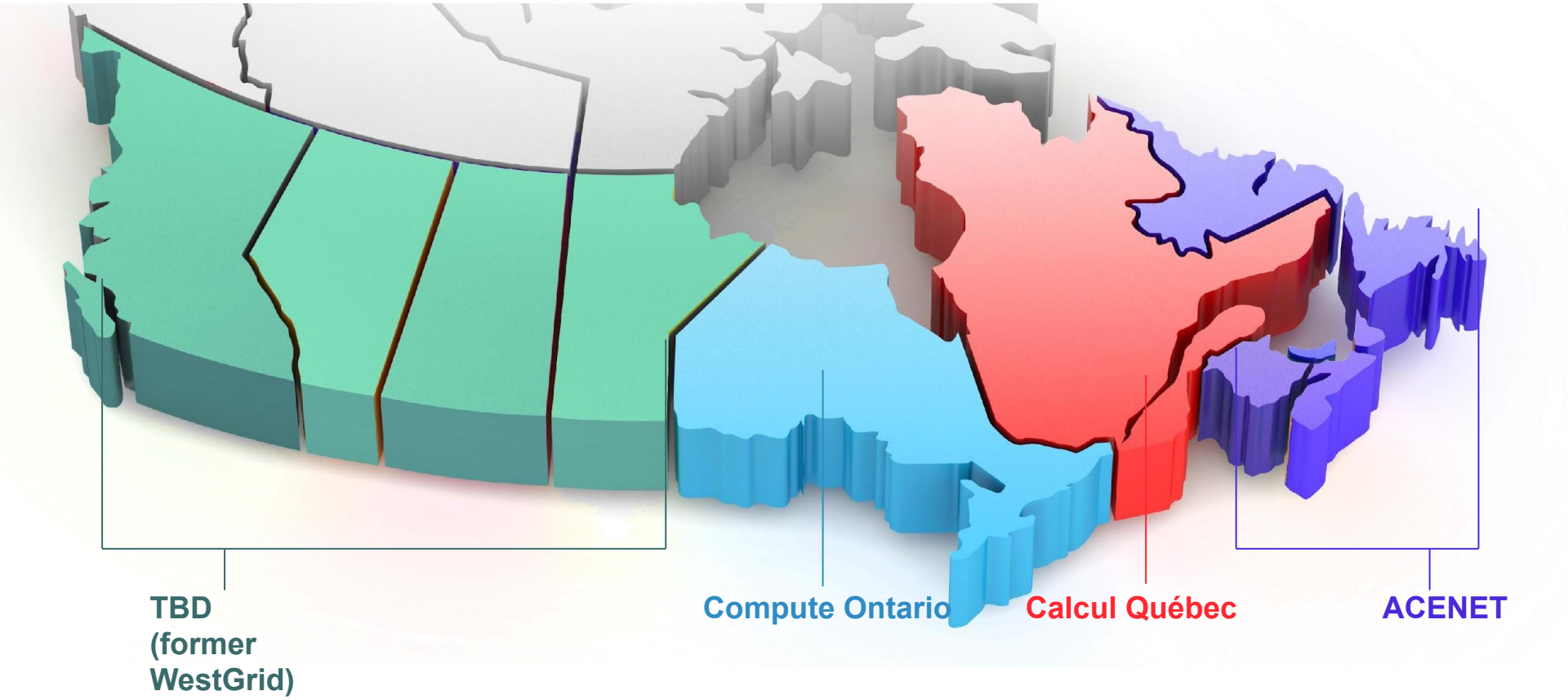
Digital Research Alliance of Canada

New User Seminar

SHARCNET™

Digital Research Alliance of Canada

Single account...



One can access all national supercomputers across the country, for free.

SHARCNET™

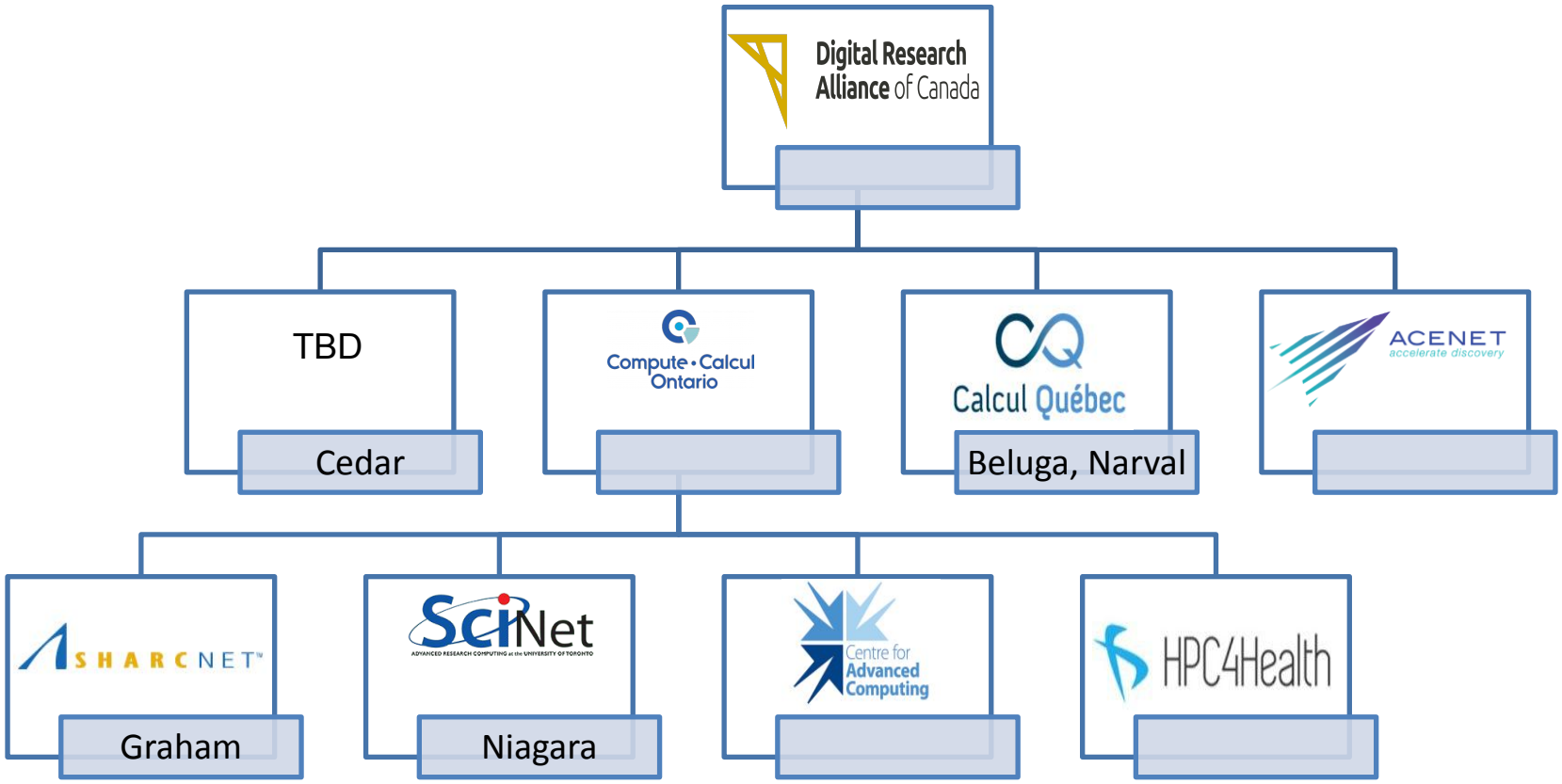
A consortium of 19 Ontario institutions providing advanced computing resources and support...

- Member of Alliance and Compute Ontario
- 3,000+ Canadian and international users

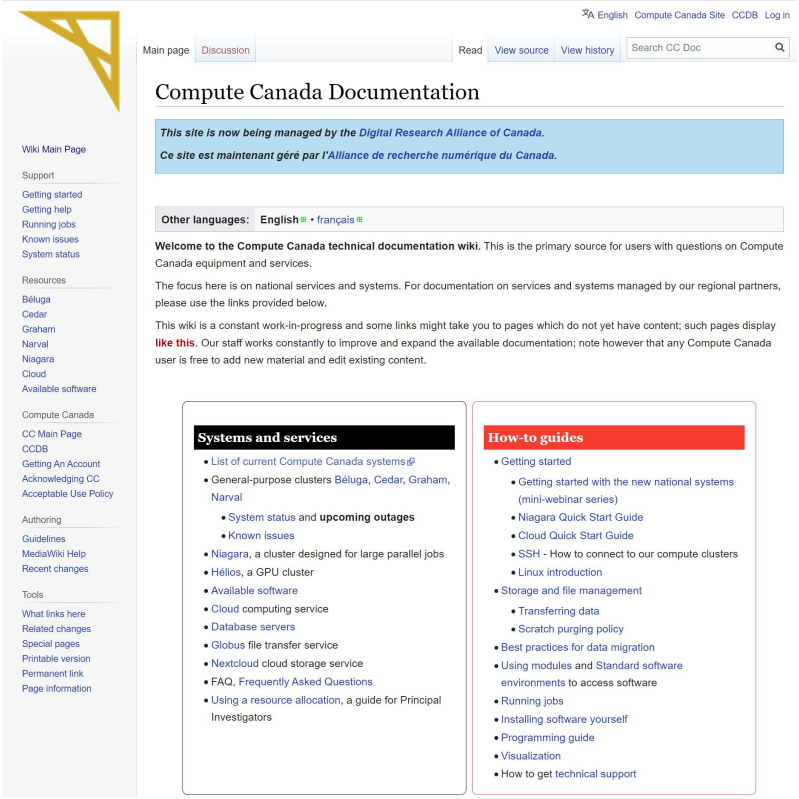
Shared
Hierarchical
Academic
Research
Computing
NETwork

- ~50,000 CPU cores
- 370+ GPUs
- 10 Gb/s network
- 100 Gb/s between nat'l centres

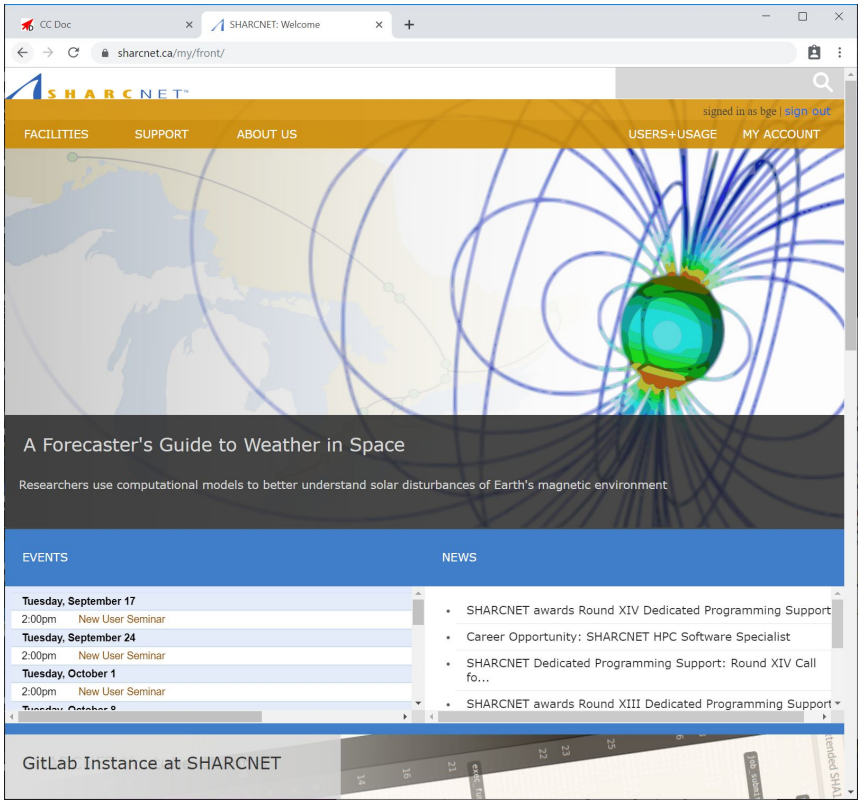
Alliance's Regional Partners



Where to look for information and get help



https://docs.computecanada.ca/



https://www.sharcnet.ca/

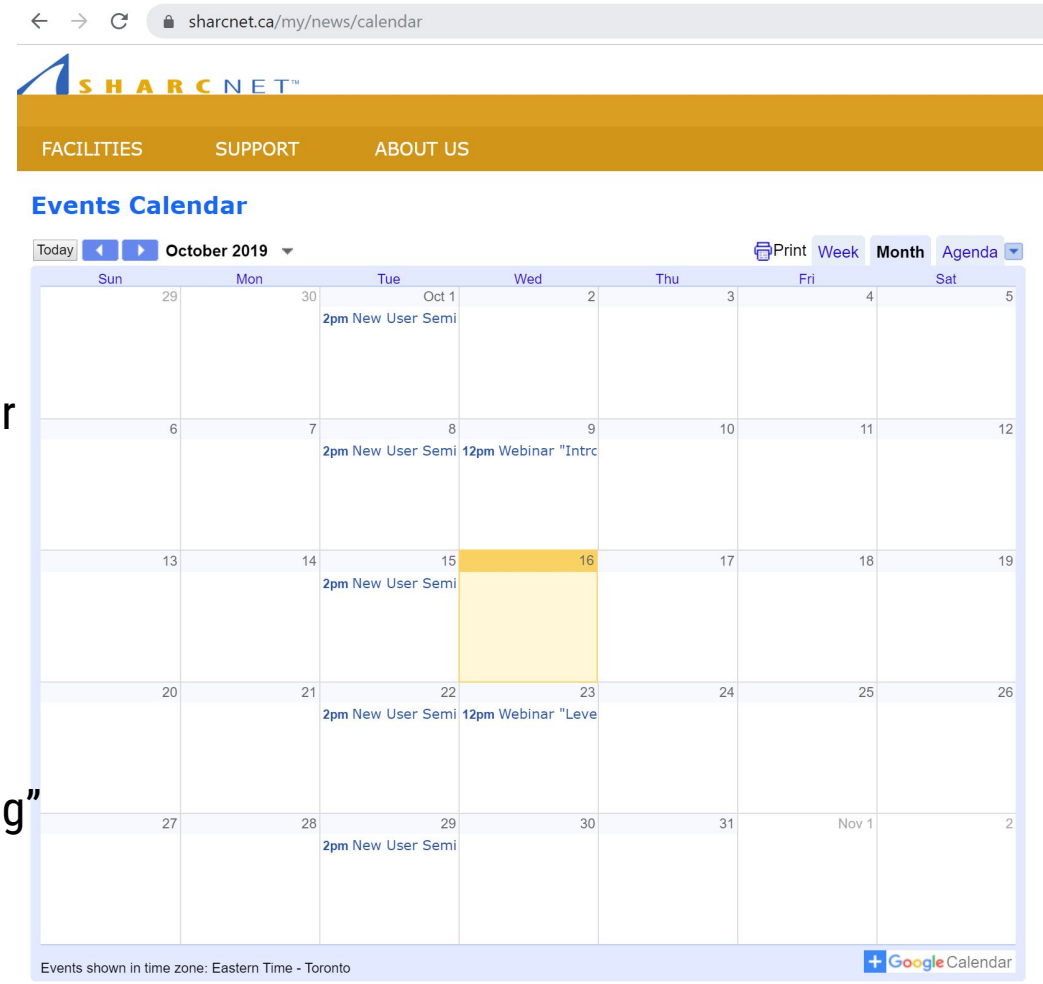
SHARCNET™

Digital Research Alliance of Canada

Where to look for information and get help

Online events

- New user seminar every Tuesday at 2pm.
- Bi-weekly general interest seminars at noon on Wednesdays.
- Check out recorded seminars on our youtube channel:
  http://youtube.sharcnet.ca
- To subscribe to our Events mailing list, send an email to events+subscribe@sharcnet.ca
- Google for “compute canada training” to find the calendar for all Compute Canada events



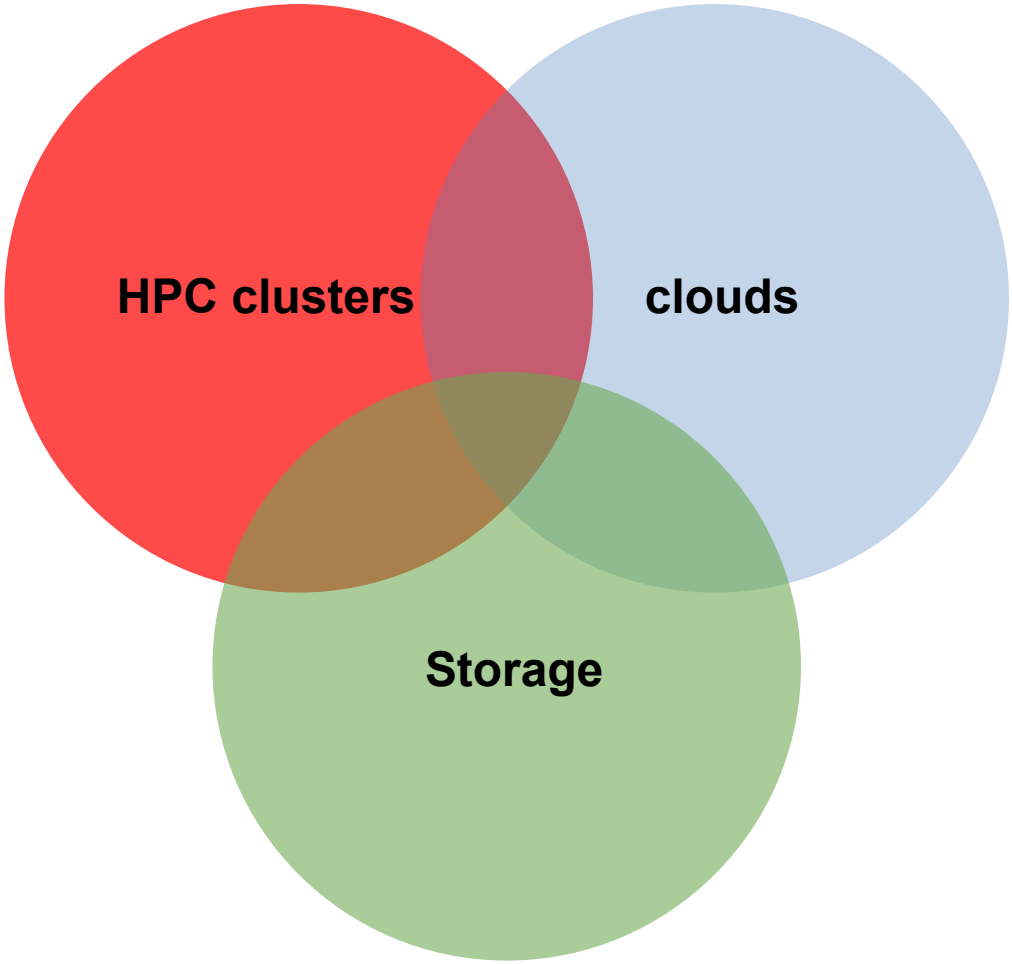
Where to look for information and get help

Interactive help

- Ticketing system via support@computecanada.ca or help@sharcnet.ca
- E-mail us - check staff contact info on https://www.sharcnet.ca/
- Phone us
- Office visit*

Use of systems
Installation of software
Access to commercial software and site licence
Debugging and optimizing code
Programming
RAC applications
...

The facilities and resources



The facilities and resources

Clusters across the country

- cedar.computecanada.ca
- graham.computecanada.ca
- narval.computecanada.ca
- niagara.computecanada.ca
- beluga.computecanada.ca

Cloud services

- arbutus.cloud.computecanada.ca
- cedar.cloud.computecanada.ca
- graham.cloud.computecanada.ca
- east.cloud.computecanada.ca

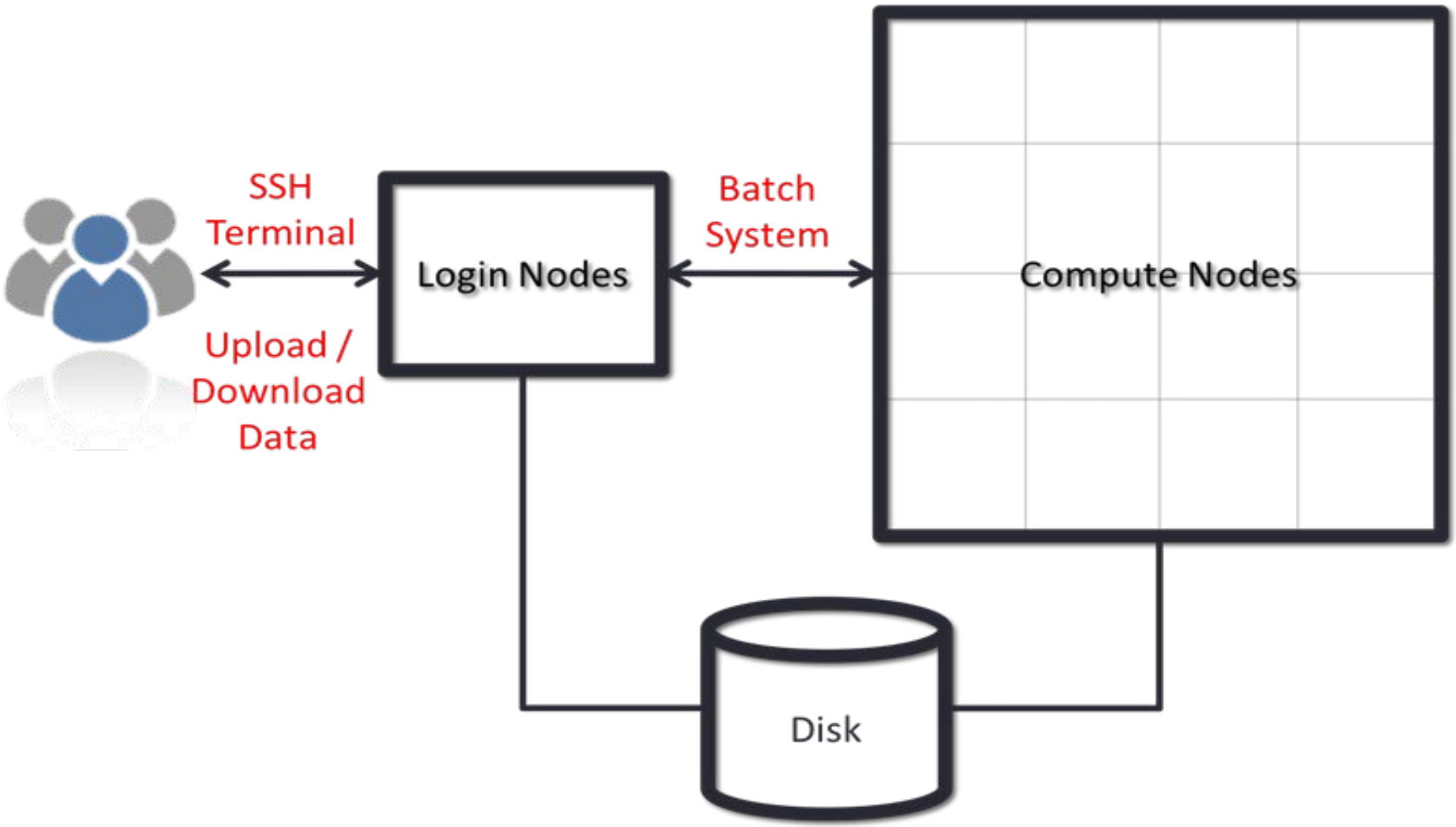
NB. Files are NOT shared across systems. You need move and copy files around as needed.

The computing environment

Cluster computing environment

- OS: 64-bit Linux - CentOS etc.
- Languages: C/C++, Fortran, R, Python, Matlab/Octave, Java, Julia, CUDA, etc.
- Access to a variety of software packages
- Parallel development support:
  - MPI, OpenMP, Pthreads, CUDA, OpenACC, OpenCL, DDT
  - C++: Language support for multithreading (since C++-11 standard)
  - Fortran: Language support for parallel programming (since 2003 standard)
  - Julia: Parallel processing constructs, shared and distributed objects
- Data science support:
  - R, Python, Julia, Spark, DASK, etc.
- Batch computing via slurm

Login nodes vs. compute nodes



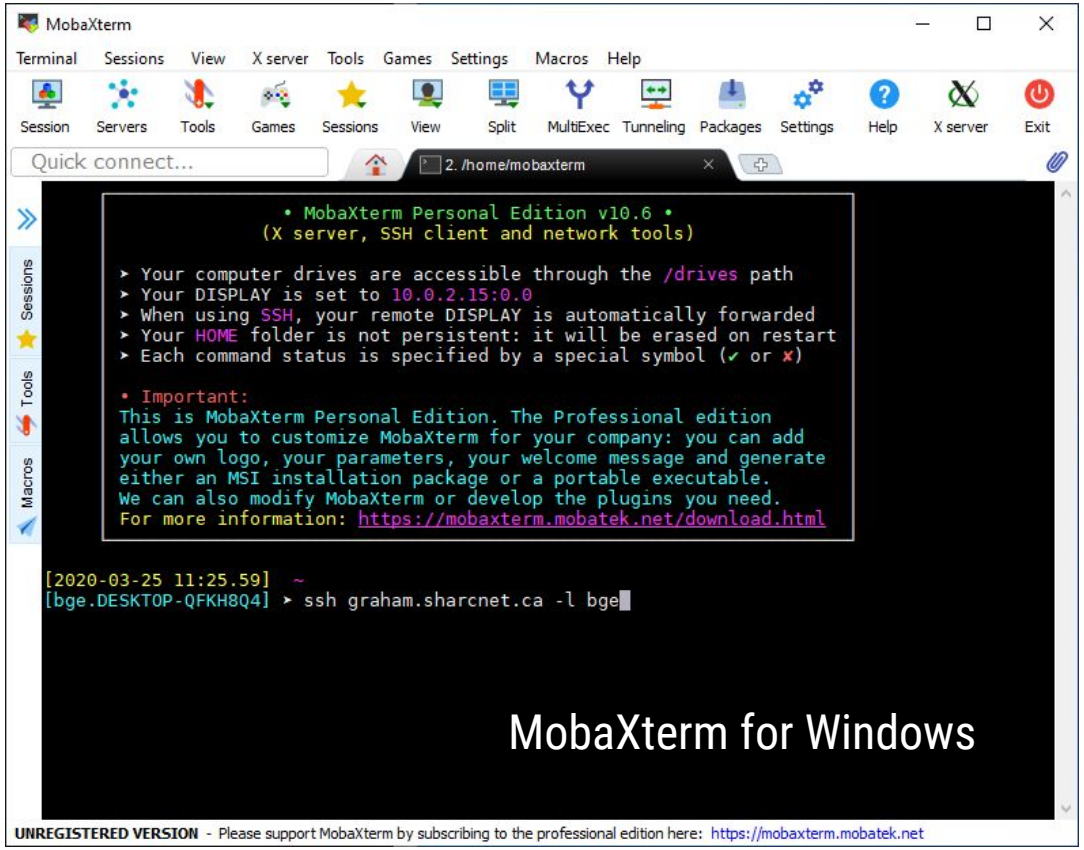
Connecting to clusters via SSH

For Windows users we recommend free software **MobaXterm**. It combines:

- SSH client (to login to systems)
- SFTP client (to copy files)
- Xwindow server (to run graphical applications)

Linux and Mac users can use command line tools ssh, scp, rsync etc.

For Mac only, running graphical applications remotely requires free software **XQuartz**.



MobaXterm for Windows

Accessing and managing files

Everyone has the access to the following file systems

- /**home**: 50G, 0.5m files; backed up regularly
- /**project**: 1T per group, 0.5m files, up to 10T per group; backed up.
- /**scratch**: 20T per user, 1m files, up to 100T; 2 months of life.
- **nearline**: to store files not currently in use, but may be needed later. 5000 files (approx. 10T) per group. NOT available on compute nodes.

*NB: Please **DO NOT** store everything, remove the files no longer in use to save space.*

Tips for project space

- Do not preserve file permissions when copying files to Project! If you do, you will likely get a “Not enough of disk space” error on Project.
- Specifically, **don’t do** the following when project is the destination, and source is not in project:

  $ cp -p

  $ mv

  $ rsync -a

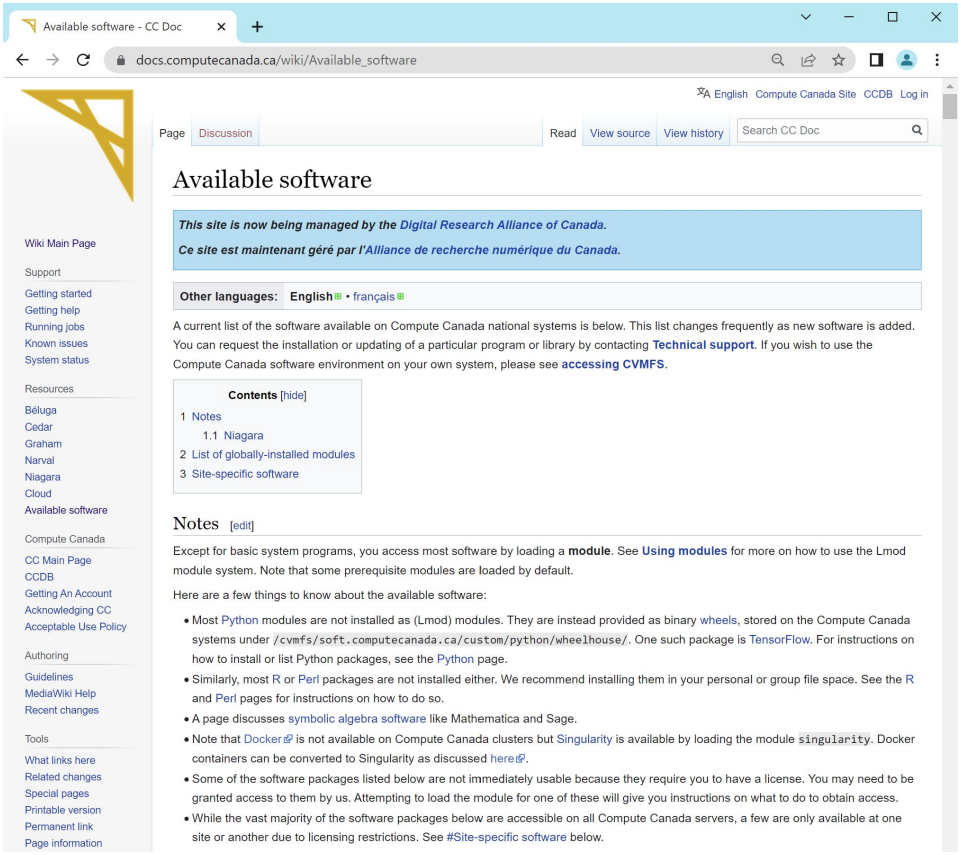
Tips for nearline space

- Quota limit is an “ingest” limit. After it has been consumed, you can add more files up to your file count limit.
- Find your file count by typing:

  find /nearline/YOUR_ACCOUNT/ | **wc** -l

- Don’t edit files in nearline! If it is on tape, your editor might hang.
- File size, some guideline:
  - At least 10G to 20G per file. Ideal file size is 100G to 500G. Up to 2T per file is acceptable.

What software packages are available?



Check for software while on a cluster

$ **module avail**

$ **module spider** *keyword*

Running jobs using a slurm script - *myjob*.sh

Submitting a serial job

```
#!/bin/bash
#SBATCH --time=00-01:00:00 # DD-HH:MM
#SBATCH --account=def-user
module load python/3.6
python simple_job.py 0 output 10
```

To see what account groups you have access to, use command **sshare -U** or **salloc** by itself

sbatch *myjob*.sh

Submitting a series of jobs

```
#!/bin/bash
#SBATCH --time=01:00
#SBATCH --account=def-user
#SBATCH --array=1-200

python simple_job.py $SLURM_ARRAY_TASK_ID output
```

META package

https://docs.computecanada.ca/wiki/META_package_for_serial_farming

(Google for “meta compute canada”)

A more universal and convenient way to manage a large number of similar jobs.

Running jobs using a slurm script - *myjob*.sh

Submitting a threaded job

```
#!/bin/bash
#SBATCH --account=def-user
#SBATCH --time=0-03:00
#SBATCH --cpus-per-task=32
#SBATCH --ntasks=1
#SBATCH --mem=20G

export \
  OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK

./myprog.exe
```

sbatch *myjob*.sh

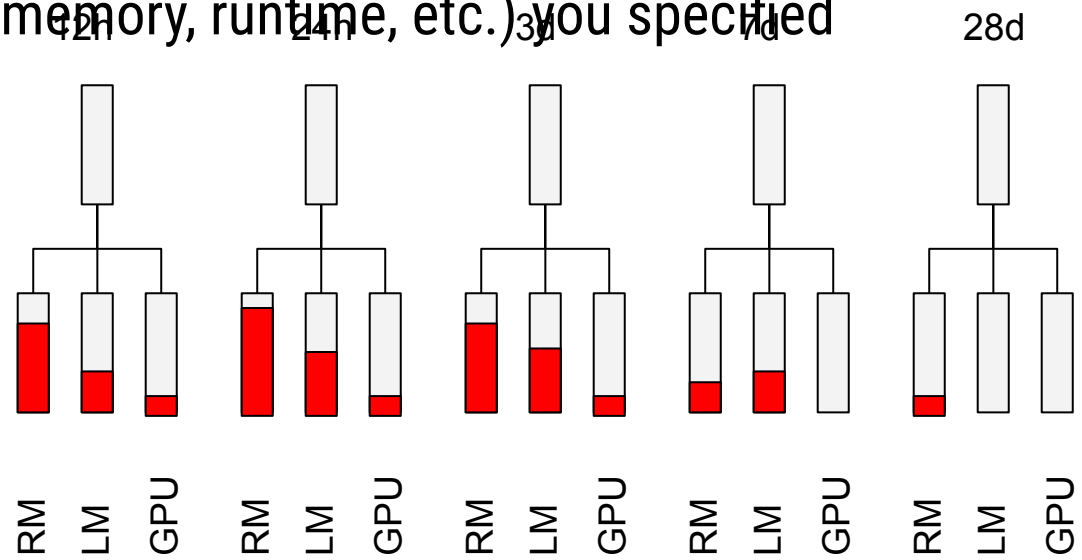
Submitting a parallel job

```
#!/bin/bash
#SBATCH --account=def-user
#SBATCH --time=5-00:00
#SBATCH --ntasks=100
#SBATCH --mem-per-cpu=4G

srun ./mympiprog.exe
```

Why my jobs don't start

On entry, each of your jobs is "placed in one of the partitions" by its attributes (cores, memory, runtime, etc.) you specified



You may use command **partition-stats** to see the system status.

What can be done about wait times

- Requesting more resources (runtime, CPU cores, memory) than what the job process requires can result in a longer queue times.

  Tip: request only what the job needs, with a bit of leeway for time and memory.

- The recent usage of an account is calculated independently on each of the Compute Canada general purpose systems (Graham, Beluga, and Cedar) and the availability of the resources varies across systems.

  Tip: use multiple systems when appropriate.

- More resources are available to full-node jobs. If your job can efficiently use multiples of 32 cpu cores (graham) it gains access to a larger set of nodes if it is submitted as a full-node job.

  Tip: use --nodes=N and --ntasks-per-node=32 sbatch arguments for full-node jobs.

- Less than 20% of all resources are available via default accounts.

  Tip: If a project needs more than the default level usage, a larger target share of the system can be obtained through the annual Resources Allocation Competition (RAC)

Common mistakes to avoid

- Do not run significant programs on login nodes, nor run programs directly on compute nodes.
- Do not specify a job run time blindly (say, 28 days), or more memory than needed for your program
- Do not create millions of tiny files, or large amounts (> GB) of uncompressed (eg. ASCII) output
- Do not let your jobs access (read/write) files frequently (more than 10 files per second from all of your running jobs).

Q&A