

Introduction to SciNet

SciNet HPC Consortium
Compute Canada

November 7, 2012

Don't Panic



Part I

ABOUT SCINET

SciNet is ...

... a consortium for High-Performance Computing consisting of researchers at U. of T. and its associated hospitals.



There are 7 consortia in Canada that provide HPC resources to Canadian academic researchers and their collaborators.



SciNet is ...

- ... where you meet other users at monthly SciNet User Group meetings.
Every 2nd Wednesday of the Month.
1 or more TechTalks (wiki.scinethpc.ca/wiki/index.php/SNUG_TechTalks)
And pizza!

- ... where you go for SciNet Developer Seminars

SciNet people

Analysts

- Jonathan Dursi
- Scott Northrup
- Ramses van Zon
- Daniel Gruner (CTO)

Hardware and sysadmin

- Jaime Pinto
- Joseph Chen
- Jason Chong
- Ching-Hsing Yu
- Neil Knecht
- Leslie Groer
- Chris Loken (CTO)

And

- Technical director Prof. Richard Peltier
- Business manager Teresa Henriques
- Project coordinator Jillian Dempsey

Part II

SCINET SYSTEMS

Compute Resources at SciNet

General Purpose Cluster (GPC)



http://wiki.scinethpc.ca/wiki/index.php/GPC_Quickstart

Compute Resources at SciNet

General Purpose Cluster (GPC)

- 3864 nodes with 8 Intel x86-64 cores @ 2.53Hz
- 30,912 cores
- 328 TFlops
- 16 GB RAM per node (~14GB for user jobs)
- 16 threads per node
- Operating system: CentOS 6
- Interconnect: InfiniBand
- #16 on the June 2009 *TOP500* (Now at #66)
- #1 in Canada

http://wiki.scinethpc.ca/wiki/index.php/GPC_Quickstart

Other Compute Resources at SciNet

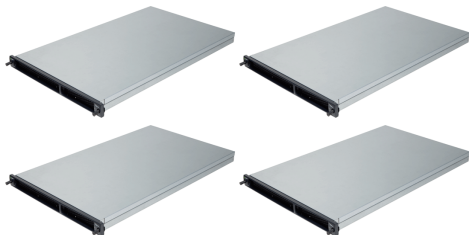
Tightly Coupled System (TCS)



Power 7 Linux Cluster (P7)



GPU Devel Nodes (ARC)



Blue Gene/Q



Storage Resources at SciNet



Disk space

- 1.4 PB of storage in 1790 drives
- Two controllers each delivering 4-5 GB/s (r/w)
- *Shared* file system GPFS on all systems
- Your files go in `/home/g/group/user` and `/scratch/g/group/user`.

Storage Resources at SciNet



Disk space

- 1.4 PB of storage in 1790 drives
- Two controllers each delivering 4-5 GB/s (r/w)
- *Shared* file system GPFS on all systems
- Your files go in `/home/g/group/user` and `/scratch/g/group/user`.

Storage space

- HPSS: Tape-backed storage expansion solution.

Access by allocation

<http://wiki.scinethpc.ca/wiki/index.php/HPSS>

Storage Limits at SciNet

location	quota	block-size	time-limit	backup	devel	comp
/home	10GB	256kB	perpetual	yes	rw	ro
/scratch	20TB/1M	4MB	3 months	no	rw	rw

Storage Limits at SciNet

location	quota	block-size	time-limit	backup	devel	comp
/home	10GB	256kB	perpetual	yes	rw	ro
/scratch	20TB/1M	4MB	3 months	no	rw	rw

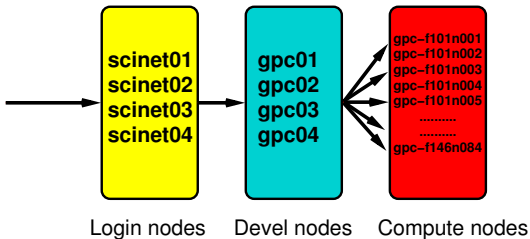
- Compute nodes do not have local hard drives.
- /home and /scratch are both part of the GPFS file system.
- GPFS is a high-performance file system which provides rapid reads and writes to large data sets in parallel from many nodes.
- Performs poorly accessing data sets which consist of many, small files.
- Avoid reading and writing lots of small amounts of data to disk.
- Many small files on the system would waste space and would be slower to access, read and write.

http://wiki.scinethpc.ca/wiki/index.php/Data_Management

Part III

USING SCINET

Using SciNet



1. Access systems: login.scinet.utoronto.ca

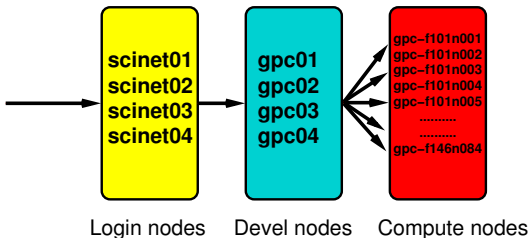
First ssh to login (not part of clusters):

```
ssh -l <username> login.scinet.utoronto.ca [-X]
```

The login nodes are gateways:

- only to be used for small data transfer
- and to proceed logging into one of the devel nodes.

Using SciNet



2. Go to the right cluster: ssh to the devel nodes

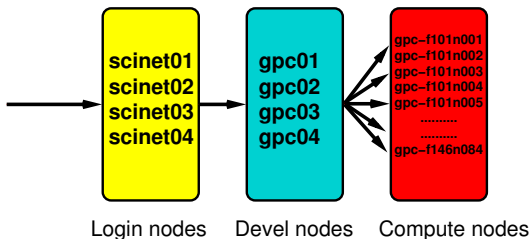
GPC: gpc01, gpc02, gpc03, gpc04

These are aliases for longer node names E.g.

```
ssh gpc03 -X
```

gets you to the gpc development node named gpc-f103n084.

Using SciNet



3. Submit a job to the compute nodes

E.g.

```
cd $SCRATCH/jobdir
qsub jobscript.sh
```

Wait, didn't we skip some steps?

Software and Libraries (1)

Once you log into devel nodes, what software is already installed?

- Other than essentials, all software installed using `module` commands.
- sets environment variables (`LD_LIBRARY_PATH`, `PATH`, ...)
- Allows multiple, conflicting versions of package to be available.
- More on *Software and Libraries* page of wiki.

http://wiki.scinethpc.ca/wiki/index.php/Software_and_Libraries

Software and Libraries (1)

Once you log into devel nodes, what software is already installed?

- Other than essentials, all software installed using module commands.
- sets environment variables (LD_LIBRARY_PATH, PATH, ...)
- Allows multiple, conflicting versions of package to be available.
- More on *Software and Libraries* page of wiki.

```
gpc-f103n084-$ module avail
```

```
-----/scinet/gpc/Modules6/Modules/versions-----  
3.2.8 3.2.9  
-----/scinet/gpc/Modules6/Modules/3.2.9/modulefiles  
dot modules use.own  
module-cvs use.deprecated  
module-info use.experimental  
-----/scinet/gpc/Modules6/Modules/modulefiles-----  
ImageMagick/6.6.7(default)  
R/2.13.1(default)  
R/2.14.1  
ROOT/5.30.03(default)  
ROOT/5.32.00  
Xlibraries/X11-64  
abyss/1.3.2  
adios/131-openmpi-gcc(default)  
amber/10.0.30  
antlr/2.7.7  
autoconf/2.68  
automake/1.11.2  
blast/2.2.23+  
...
```

http://wiki.scinethpc.ca/wiki/index.php/Software_and_Libraries

Software and Libraries (2)

<code>module load <module-name></code>	use particular software
<code>module purge</code>	remove currently loaded modules
<code>module avail</code>	list available software packages
<code>module list</code>	list loaded modules
<code>module help <module-name></code>	describe a module

- Load frequently used modules in `.bashrc` in home directory.
- Load run-specific modules inside the job script.
- Short name gives default (e.g. `intel` → `intel/12.1.3`)

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

```
gpc-f103n084$
```


Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

```
gpc-f103n084$ module purge
```

```
gpc-f103n084$
```

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

```
gpc-f103n084$ module purge
```

```
gpc-f103n084$ module load python
```

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

```
gpc-f103n084$ module purge
```

```
gpc-f103n084$ module load python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'gcc/4.7.0 gcc/4.6.1 gcc/4.4.6'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq gcc
```

```
gpc-f103n084$
```

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

```
gpc-f103n084$ module purge
```

```
gpc-f103n084$ module load python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'gcc/4.7.0 gcc/4.6.1 gcc/4.4.6'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq gcc
```

```
gpc-f103n084$ module load gcc python
```

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

```
gpc-f103n084$ module purge
```

```
gpc-f103n084$ module load python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'gcc/4.7.0 gcc/4.6.1 gcc/4.4.6'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq gcc
```

```
gpc-f103n084$ module load gcc python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'intel/12.1.5 intel/12.1.3 intel/12.1.2 intel/12.1'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq intel
```

```
gpc-f103n084$
```

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

```
gpc-f103n084$ module purge
```

```
gpc-f103n084$ module load python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'gcc/4.7.0 gcc/4.6.1 gcc/4.4.6'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq gcc
```

```
gpc-f103n084$ module load gcc python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'intel/12.1.5 intel/12.1.3 intel/12.1.2 intel/12.1'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq intel
```

```
gpc-f103n084$ module load gcc intel python
```

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

```
gpc-f103n084$ module purge
```

```
gpc-f103n084$ module load python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'gcc/4.7.0 gcc/4.6.1 gcc/4.4.6'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq gcc
```

```
gpc-f103n084$ module load gcc python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'intel/12.1.5 intel/12.1.3 intel/12.1.2 intel/12.1'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq intel
```

```
gpc-f103n084$ module load gcc intel python
```

```
gpc-f103n084$
```

Software and Libraries (3)

Dependencies

- Modules sometimes require other modules to be loaded first.
- Module will let you know if you didn't.

Example

```
gpc-f103n084$ module purge
```

```
gpc-f103n084$ module load python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'gcc/4.7.0 gcc/4.6.1 gcc/4.4.6'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq gcc
```

```
gpc-f103n084$ module load gcc python
```

```
python/2.7.2(11):ERROR:151: Module 'python/2.7.2' depends on one of  
the module(s) 'intel/12.1.5 intel/12.1.3 intel/12.1.2 intel/12.1'
```

```
python/2.6.2(11):ERROR:102: Tcl command execution failed:  prereq intel
```

```
gpc-f103n084$ module load gcc intel python
```

```
gpc-f103n084$ module list
```

Currently Loaded Modulefiles:

1) gcc/4.6.1

2) intel/12.1.3

3) python/2.7.2

Software and Libraries (4)

Commercial Software?

- SciNet has an extremely large and broad user base (~ 1000 users)
- Only commercial software that can benefit everyone:
Compilers, math libraries and debugger.
- No Matlab, Gaussian, IDL, . . .
(but Octave)
- Can help you to install software for which you have a license.

Compiling on SciNet (1): GPC

- From `login.scinet.utoronto.ca`, ssh to one of the four devel nodes.

```
ssh gpc04 [-X]
```

or

```
gpcdev [-X]
```

- We recommend Intel compilers, which are

```
icc, icpc, ifort
```

for C, C++, and Fortran, respectively (from the module `intel`)

- Optimize code for the GPC machine using of at least

```
-O3 -xhost
```

- Add `-openmp` to the command line for OpenMP

- Compile MPI code with `mpif77/mpif90/mpicc/mpicxx`.

- 1 OpenMPI, in module `openmpi` (v1.4.4)
- 2 Intel MPI, in module `intelmpl` (v4.0.3)

Compiling on SciNet (2): library modules

- To compile code that uses that a library from a module, add

```
-I${SCINET_[shortmodulename]_INC}
```

- To link, add

```
-L${SCINET_[shortmodulename]_LIB}
```

Compiling on SciNet (3): library modules

Example

```
scinet04$ ssh gpc03
gpc-f103n084$ module list
No Modulefiles Currently Loaded.
gpc-f103n084$ pwd
/home/s/scinet/rzon
gpc-f103n084$ ls
main.c module.c
gpc-f103n084$ module load intel gsl
gpc-f103n084$ module list
Currently Loaded Modulefiles:
1) intel/12.1.3 2) gsl/1.13-intel
gpc-f103n084$ icc -c -O3 -xHost -o main.o main.c
gpc-f103n084$ icc -c -O3 -xHost -I$SCINET_GSL_INC -o module.o module.c
gpc-f103n084$ icc -o main module.o main.o -L$SCINET_GSL_LIB -lgsl -mkl
gpc-f103n084$ ./main
```

Testing (1)

Why test?

- Computations are run by **Moab**, our **job scheduler**:
 - Not interactive.
 - Gets queued, i.e. does not run immediately.

So one cannot catch obvious bugs quickly.

- You need to test your job's requirements and scaling behaviour, so you know what to request from the scheduler.
- To avoid surprises, start runs on a small scale and work your way up to larger scales.

Testing (2)

How to test

- Computations do not run on the devel nodes, but through the **job scheduler**.
- But small test jobs can be run on the devel nodes. Rule of thumb: couple of minutes, taking at most about 1-2GB of memory.
- You can run the **ddt** debugger (or gdb or ddd) on the devel nodes.
- Short tests that do not fit on a development node, or for which you need a dedicated node, can be run through the **interactive debug queue** (more later).

Submitting jobs

- To run a job, you must submit to a batch queue.
- You submit jobs from a devel node in the form of a script
- Scheduling is by node. **You need to use all 8 cores on the node!**
- Must run from the scratch directory (**home=read-only**)
- Copy essential results out after runs have finished.

Submitting jobs

Limits

- Group based limits:
possible for your colleagues to exhaust group limits
- Talk to us first to run massively parallel jobs (> 2048 cores)
- While their resources last, jobs will run at a higher priority than others for groups with an allocation.

GPC queues

queue	min.time	max.time	max jobs	max cores
batch	15m	48h	32/1000	256/8000
debug		2h/30m	1	16/64
largemem	15m	48h	1	16

GPC queues

- Submit to these queues from a GPC devel node with

```
qsub [options] <script>
```

- Common options (usually in script):

-l: specifies requested nodes and time, e.g.

```
-l nodes=2:ppn=8,walltime=1:00:00
```

-q: specifies the queue, e.g.

```
-q batch
```

```
-q debug
```

```
-q largemem
```

-I specifies that you want an interactive session.

-X specifies that you want X forwarded.

GPC queues

- The largemem queue is exceptional, in that it provides access to two nodes (only) that have 16 processors and 128GB of ram.
- There is no queue for serial jobs, so if you have serial jobs, **YOU** will have to bunch together 8 of them to use the node's full power. GNU Parallel[†] can help you with that.

[†] O. Tange (2011): GNU Parallel - The Command-Line Power Tool, *login: The USENIX Magazine*, February 2011:42-47.

GPC job script example

```
#!/bin/bash
#PBS -l nodes=1:ppn=8
#PBS -l walltime=1:00:00
#PBS -N simple-omp-job
cd $PBS_O_WORKDIR
export OMP_NUM_THREADS=8
./openmp_example > output
```

```
$ qsub scriptname.pbs
```

GPC queues

Monitoring

Once the job is incorporated into the queue (this takes a minute),
Command you can use:

- `showq` to show the queue;
- `showstart JOBID` to get an estimate for when it will run
- `checkjob JOBID` to get information on the job (quite verbose).
- `canceljob JOBID` to cancel the job.

GPC queues

Jobs can be in one of three states:

- 1 **Running:** great, all is well (at least from a scheduler point-of-view).
- 2 **Idle:** waiting in the queue for resource. All is well (at least from a scheduler point-of-view).
- 3 **Blocked:** too many jobs submitted at the same time; scheduler will not consider these until some of the idle ones have been scheduled.

More serious errors (such as asking for a node with 16 cores), will lead to a rejection, and you will get an error with a more-or-less cryptic explanation. See the FAQ on the wiki for some typical cases and their explanation.

Example 1 (GPC)

gpc-f101n084-s

Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi  
gpc-f101n084-$
```


Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi
```

```
gpc-f101n084-$ mpif90 -O3 -xhost mycode.f90 -o mycode
```

```
gpc-f101n084-$
```

Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi
gpc-f101n084-$ mpif90 -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example1
gpc-f101n084-$
```

Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi
gpc-f101n084-$ mpif90 -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example1
gpc-f101n084-$ cp mycode $SCRATCH/example1
gpc-f101n084-$
```

Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi
gpc-f101n084-$ mpif90 -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example1
gpc-f101n084-$ cp mycode $SCRATCH/example1
gpc-f101n084-$ cd $SCRATCH/example1
gpc-f101n084-$
```

Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi
gpc-f101n084-$ mpif90 -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example1
gpc-f101n084-$ cp mycode $SCRATCH/example1
gpc-f101n084-$ cd $SCRATCH/example1
gpc-f101n084-$ cat > myjob.pbs
    #!/bin/bash
    #PBS -l nodes=8:ppn=8,walltime=1:00:00
    #PBS -N JobName
    cd $PBS_O_WORKDIR
    module load intel openmpi
    mpirun -np 64 ./mycode > out
gpc-f101n084-$
```

Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi
gpc-f101n084-$ mpif90 -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example1
gpc-f101n084-$ cp mycode $SCRATCH/example1
gpc-f101n084-$ cd $SCRATCH/example1
gpc-f101n084-$ cat > myjob.pbs
    #!/bin/bash
    #PBS -l nodes=8:ppn=8,walltime=1:00:00
    #PBS -N JobName
    cd $PBS_O_WORKDIR
    module load intel openmpi
    mpirun -np 64 ./mycode > out
gpc-f101n084-$ qsub myjob.pbs
    2961983.gpc-sched
gpc-f101n084-$
```

Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi
gpc-f101n084-$ mpif90 -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example1
gpc-f101n084-$ cp mycode $SCRATCH/example1
gpc-f101n084-$ cd $SCRATCH/example1
gpc-f101n084-$ cat > myjob.pbs
    #!/bin/bash
    #PBS -l nodes=8:ppn=8,walltime=1:00:00
    #PBS -N JobName
    cd $PBS_O_WORKDIR
    module load intel openmpi
    mpirun -np 64 ./mycode > out
gpc-f101n084-$ qsub myjob.pbs
    2961983.gpc-sched
gpc-f101n084-$ qstat    (or checkjob 2961983, or showq -u $USER)
```

Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi
gpc-f101n084-$ mpif90 -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example1
gpc-f101n084-$ cp mycode $SCRATCH/example1
gpc-f101n084-$ cd $SCRATCH/example1
gpc-f101n084-$ cat > myjob.pbs
#!/bin/bash
#PBS -l nodes=8:ppn=8,walltime=1:00:00
#PBS -N JobName
cd $PBS_O_WORKDIR
module load intel openmpi
mpirun -np 64 ./mycode > out
gpc-f101n084-$ qsub myjob.pbs
2961983.gpc-sched
gpc-f101n084-$ qstat (or checkjob 2961983, or showq -u $USER)
Job id          Name          User Time Use S Queue
-----
2961983.gpc-sched JobName      rzon          0 Q batch
gpc-f101n084-$
```


Example 1 (GPC)

```
gpc-f101n084-$ module load intel openmpi
gpc-f101n084-$ mpif90 -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example1
gpc-f101n084-$ cp mycode $SCRATCH/example1
gpc-f101n084-$ cd $SCRATCH/example1
gpc-f101n084-$ cat > myjob.pbs
#!/bin/bash
#PBS -l nodes=8:ppn=8,walltime=1:00:00
#PBS -N JobName
cd $PBS_O_WORKDIR
module load intel openmpi
mpirun -np 64 ./mycode > out
gpc-f101n084-$ qsub myjob.pbs
2961983.gpc-sched
gpc-f101n084-$ qstat (or checkjob 2961983, or showq -u $USER)
Job id          Name          User Time Use S Queue
-----
2961983.gpc-sched JobName      rzon          0 Q batch
gpc-f101n084-$ ls
JobName.e2961983 JobName.o2961983 mycode myjob.pbs
out
```

Example 2 (GPC)

gpc-f101n084-\$

Example 2 (GPC)

```
gpc-f101n084-$ module load intel  
gpc-f101n084-$
```

Example 2 (GPC)

```
gpc-f101n084-$ module load intel
```

```
gpc-f101n084-$ ifort -O3 -xhost mycode.f90 -o mycode
```

```
gpc-f101n084-$
```

Example 2 (GPC)

```
gpc-f101n084-$ module load intel
```

```
gpc-f101n084-$ ifort -O3 -xhost mycode.f90 -o mycode
```

```
gpc-f101n084-$ mkdir $SCRATCH/example2
```

```
gpc-f101n084-$
```

Example 2 (GPC)

```
gpc-f101n084-$ module load intel
gpc-f101n084-$ ifort -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example2
gpc-f101n084-$ cp mycode $SCRATCH/example2
gpc-f101n084-$
```

Example 2 (GPC)

```
gpc-f101n084-$ module load intel
gpc-f101n084-$ ifort -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example2
gpc-f101n084-$ cp mycode $SCRATCH/example2
gpc-f101n084-$ cd $SCRATCH/example2
gpc-f101n084-$
```

Example 2 (GPC)

```
gpc-f101n084-$ module load intel
gpc-f101n084-$ ifort -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example2
gpc-f101n084-$ cp mycode $SCRATCH/example2
gpc-f101n084-$ cd $SCRATCH/example2
gpc-f101n084-$ cat > joblist.txt
    mkdir run1; cd run1; ../mycode 1 > out
    mkdir run2; cd run2; ../mycode 2 > out
    mkdir run3; cd run3; ../mycode 3 > out
    ...
    mkdir run64; cd run64; ../mycode 64 > out
gpc-f101n084-$
```


Example 2 (GPC)

```
gpc-f101n084-$ module load intel
gpc-f101n084-$ ifort -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example2
gpc-f101n084-$ cp mycode $SCRATCH/example2
gpc-f101n084-$ cd $SCRATCH/example2
gpc-f101n084-$ cat > joblist.txt
    mkdir run1; cd run1; ../mycode 1 > out
    mkdir run2; cd run2; ../mycode 2 > out
    mkdir run3; cd run3; ../mycode 3 > out
    ...
    mkdir run64; cd run64; ../mycode 64 > out
gpc-f101n084-$ cat > myjob.pbs
    #!/bin/bash
    #PBS -l nodes=1:ppn=8,walltime=24:00:00
    #PBS -N ASerialJob
    cd $PBS_O_WORKDIR
    module load intel gnu-parallel
    parallel -j 8 < joblist.txt
gpc-f101n084-$
```

Example 2 (GPC)

```
gpc-f101n084-$ module load intel
gpc-f101n084-$ ifort -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example2
gpc-f101n084-$ cp mycode $SCRATCH/example2
gpc-f101n084-$ cd $SCRATCH/example2
gpc-f101n084-$ cat > joblist.txt
    mkdir run1; cd run1; ../mycode 1 > out
    mkdir run2; cd run2; ../mycode 2 > out
    mkdir run3; cd run3; ../mycode 3 > out
    ...
    mkdir run64; cd run64; ../mycode 64 > out
gpc-f101n084-$ cat > myjob.pbs
    #!/bin/bash
    #PBS -l nodes=1:ppn=8,walltime=24:00:00
    #PBS -N ASerialJob
    cd $PBS_O_WORKDIR
    module load intel gnu-parallel
    parallel -j 8 < joblist.txt
gpc-f101n084-$ qsub myjob.pbs
    2961985.gpc-sched
gpc-f101n084-$
```

Example 2 (GPC)

```
gpc-f101n084-$ module load intel
gpc-f101n084-$ ifort -O3 -xhost mycode.f90 -o mycode
gpc-f101n084-$ mkdir $SCRATCH/example2
gpc-f101n084-$ cp mycode $SCRATCH/example2
gpc-f101n084-$ cd $SCRATCH/example2
gpc-f101n084-$ cat > joblist.txt
    mkdir run1; cd run1; ../mycode 1 > out
    mkdir run2; cd run2; ../mycode 2 > out
    mkdir run3; cd run3; ../mycode 3 > out
    ...
    mkdir run64; cd run64; ../mycode 64 > out
gpc-f101n084-$ cat > myjob.pbs
    #!/bin/bash
    #PBS -l nodes=1:ppn=8,walltime=24:00:00
    #PBS -N ASerialJob
    cd $PBS_O_WORKDIR
    module load intel gnu-parallel
    parallel -j 8 < joblist.txt
gpc-f101n084-$ qsub myjob.pbs
    2961985.gpc-sched
gpc-f101n084-$ ls
    ASerialJob.e2961985  ASerialJob.o2961985  joblist.txt  mycode
    myjob.pbs            run1/                 run2/        run3/
```

Part IV

DATA MANAGEMENT TIPS

I/O strategies

- Do not read and write lots of small amounts of data to disk.
Reading data in from one 4MB file can be enormously faster than from 100 40KB files.
- Write data out in binary. Faster and takes less space.
- Each process writing to a file of its own is not scalable.
A directory gets locked by the first process accessing it, so the other processes have to wait for it.
- If you must read and write a lot to disk, use ramdisk if possible.
The ramdisk can be accessed using `/dev/shm/` and is currently set to 11GB max.
- Copy back from ramdisk at end of run.

Moving large data

Moving less than 10GB through the login nodes

- Only login nodes visible from outside SciNet (1Gb/s link).
- Use scp or rsync.
- but datamover1 node is faster.

Moving more than 10GB through the datamover1 node

- Should be done from the datamover1 node (10Gb/s link).
- From any SciNet node, ssh to datamover1.
- Transfers must originate from datamover1.
- Your machine must be reachable from the outside.

Moving data to HPSS

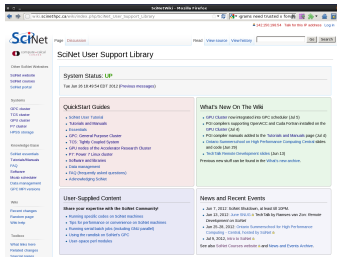
- HPSS is a tape-based storage solution.
- Available to groups with a special allocation > 5TB.

Part V

USEFUL SITES

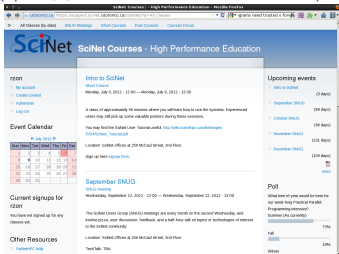
www.scinethpc.ca

wiki.scinethpc.ca



https://support.scinet.utoronto.ca/courses

https://portal.scinet.utoronto.ca



SciNet usage reports
 Change password, default allocation, maillist subscriptions

