

Introduction to SciNet

SciNet HPC Consortium
Compute Canada

July 9, 2012

Don't Panic

Outline

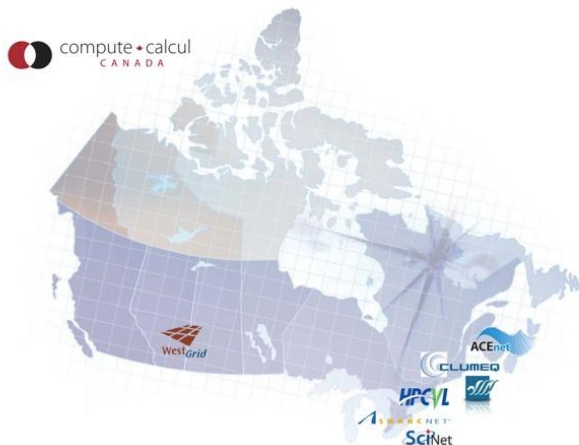
- 1 About SciNet
- 2 SciNet systems
- 3 Using SciNet
- 4 Data management

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

SciNet is ...

... home to the largest supercomputer in Canada (and other clusters)



SciNet is ...

... where you go for courses on a wide range of computational topics.

- ▶ Intro to Scientific Programming with Modern FORTRAN
- ▶ Intro to Scientific Programming with C++
- ▶ Intro to GPGPU with CUDA
- ▶ Parallel I/O
- ▶ Scientific Computing Course (for credit for physics/astrophysics grads)
- ▶ Intro to SciNet
- ▶ ...

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

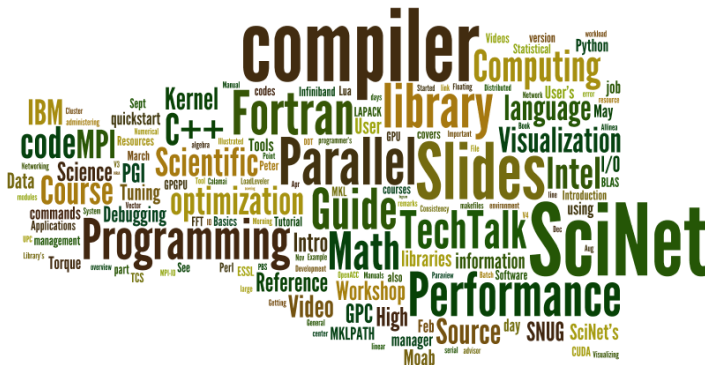
... recognized by NVIDIA as both a CUDA research and teaching centre.



... where you meet other users at monthly SciNet User Group meetings.

SciNet is ...

... where to find a wealth of online information at <http://wiki.scinethpc.ca>



SciNet is ...

... 4 technical analysts who can work directly with you to use our resources to produce good science.

- Jonathan Dursi
- Scott Northrup
- Ramses van Zon
- Daniel Gruner

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

+ 7 people that make sure everything runs smoothly.

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

How to get an account (1)

- Register with the Compute Canada Database (CCDB)
- Non-faculty: get sponsor's CCRI number
(sponsor has to have a SciNet account)
- Login to CCDB and apply for a SciNet account
(on the *Consortium Accounts* page, click *Apply* next to SciNet)
- Agree to the Acceptable Usage Policy
(e.g.: don't share account, respect others, we can monitor jobs)

How to get an account (2)

Default account

- Allows to simultaneously run 32 jobs of 48 hours wall-time.
- Use of maximally 256 cores.
- So in a year you could use over 2 million core-hours.
- 10GB of storage (plus a bunch of temporary scratch storage)

How to get an account (3)

How to get more resources on SciNet systems

- Users who will be needing more than the default amount of resources (compute cycles and/or storage) must have their PI apply for it through the yearly, competitively awarded allocations.
- Applications due in the fall of each year.
- Having an allocation also increases priority in the queue.

Part II

SCINET SYSTEMS

Resources at SciNet (1)

General Purpose Cluster (GPC)



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

Resources at SciNet (1)

General Purpose Cluster (GPC)

- 3864 nodes with 8 Intel x86-64 cores @ 2.53Hz
- 30,912 cores
- 328 TFlops
- 16 GB RAM per node (~ 14 GB for user jobs)
- 16 threads per node
- Operating system: CentOS 6
- Interconnect: InfiniBand network: $\frac{1}{4}$ DDR, $\frac{3}{4}$ QDR 5:1
- #16 on the June 2009 *TOP500* supercomputer sites (Now at #66)
- #1 in Canada

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

Resources at SciNet (2)

Tightly Coupled System (TCS)



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

Resources at SciNet (2)

Tightly Coupled System (TCS)

- 102 nodes with 32 Power-6 cores @ 4.7GHz
- 3264 cores
- 62 TFlops
- 128 GB RAM per node
- 64 threads per node
- Operating system: AIX
- Interconnect: InfiniBand
- #80 on the June 2009 *TOP500*

For access, your app should scale well to >32 procs

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

Resources at SciNet (3)

Power 7 Linux Cluster (P7)



Resources at SciNet (3)

Power 7 Linux Cluster (P7)



- 5 nodes with 32 Power-7 cores @ 3.3GHz
- 160 cores
- 4.2 TFlops
- 128 GB RAM per node
- 128 threads per node
- Operating system: BlueHat 6
- Interconnect: InfiniBand

Accessible to TCS users

http://wiki.scinethpc.ca/wiki/index.php/P7_Linux_Cluster

Resources at SciNet (4)

GPU Devel Nodes (ARC)



Resources at SciNet (4)

GPU Devel Nodes (ARC)

8 GPU devel nodes and 16 NVIDIA Tesla M2070. Per node:

- 8 Intel cores @ 2.67 GHz
- 48 GB RAM
- Operating system: CentOS 6
- Interconnect: InfiniBand DDR
- 2 $\times$ GPUs with Cuda capability 2.0 (Fermi) each with 448 Cuda cores @ 1.15GHz and 6 GB of RAM.

From CPUs: 683.52 GFlops max from 64 cores

From GPUs: 8.24 TFlops (double prec) from 7168 Cuda cores

Access upon request.

Resources at SciNet (4)

GPU Devel Nodes (ARC)

8 GPU devel nodes and 16 NVIDIA Tesla M2070. Per node:

- 8 Intel cores @ 2.67 GHz
- 48 GB RAM
- Operating system: CentOS 6
- Interconnect: InfiniBand DDR
- 2 $\times$ GPUs with Cuda capability 2.0 (Fermi) each with 448 Cuda cores @ 1.15GHz and 6 GB of RAM.

From CPUs: 683.52 GFlops max from 64 cores

From GPUs: 8.24 TFlops (double prec) from 7168 Cuda cores

Access upon request.

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

Resources at SciNet (5)

Soon: Blue Gene/Q

The BGQ is an extremely powerful and energy efficient 3rd generation IBM Supercomputer. It is expected to arrive at SciNet in early fall 2012.



Part of the Southern Ontario Smart Computing and Innovation Platform

<http://soscip.org/?q=node/1>

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

Resources at SciNet (6)

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.

Resources at SciNet (6)

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.

Restricted access

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



Resources at SciNet (6)

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.

Restricted access

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



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

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

Part III

USING SCINET

Get on the system

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, they are not part of any of the clusters and they are only to be used for small data transfer and to proceed logging into one of the devel nodes.

Get on the system

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

GPC: gpc01, gpc02, gpc03, gpc04

ARC: arc01

TCS: tcs01, tcs02

P7: p701

These are aliases for longer node names E.g.

```
ssh gpc03 -X
```

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

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:
 - GPC and ARC: Intel compilers, MKL (module intel)
 - ARC: PGI compilers (module pgi)
 - TCS and P7 IBM compilers, ESSL (modules vacpp, xlf, pe)
 - GPC, ARC, TCS and P7: DDT debugger from Allinea (module ddt)
- 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 -X gpc03
gpc-f103n084$ module list

No Modulefiles Currently Loaded.

gpc-f103n084$ pwd
/home/s/scinet/rzon

gpc-f103n084$ ls
main.c morecode.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 -O3 -xHost -o main.o main.c

gpc-f103n084$ icc -O3 -xHost -I${SCINET_GSL_INC} -o morecode.o morecode.c

gpc-f103n084$ icc -o main morecode.o main.o -L${SCINET_GSL_LIB} -lgsl -lgslcblas

gpc-f103n084$ ./main
```

Compiling on SciNet (4): TCS and P7

- ssh to a devel node

`ssh tcs01` *or* `ssh tcs02`

- Use IBM compilers: `xlc`, `xlC`, `xlF` for C, C++, and Fortran
(module load `vacpp xlF` for the latest version)
- For OpenMP, use `xlc_r`, `xlC_r`, `xlF_r`.
- For MPI, `mpicc`, `mpCC`, `mpxlf` are the mpi wrappers.
- Suggested compiler flags:

`-q64 -O3 -qhot -qarch=auto -qtune=auto`

supplemented by `-qsmp=omp` for OpenMP programs.

- On the link line we suggest using

`-q64 -bdatapsize:64k -bstacksize:64k`

also supplemented by `-qsmp=omp` for OpenMP programs.

Compiling on SciNet (4): TCS and P7

- ssh to a devel node

`ssh tcs01` *or* `ssh tcs02`

- Use IBM compilers: `xlc`, `xlC`, `xlF` for C, C++, and Fortran
(module load `vacpp xlF` for the latest version)
- For OpenMP, use `xlc_r`, `xlC_r`, `xlF_r`.
- For MPI, `mpicc`, `mpCC`, `mpxlf` are the mpi wrappers.
- Suggested compiler flags:

`-q64 -O3 -qhot -qarch=auto -qtune=auto`

supplemented by `-qsmp=omp` for OpenMP programs.

- On the link line we suggest using

`-q64 -bdatapsize:64k -bstacksize:64k`

also supplemented by `-qsmp=omp` for OpenMP programs.

Compiling on SciNet (4): TCS and P7

- ssh to a devel node

`ssh tcs01` or `ssh tcs02` , or `ssh p701`

- Use IBM compilers: `xlc`, `xlC`, `xlF` for C, C++, and Fortran (module load `vacpp xlF` for the latest version)
- For OpenMP, use `xlc_r`, `xlC_r`, `xlF_r`.
- For MPI, `mpicc`, `mpCC`, `mpxlf` are the mpi wrappers.
- Suggested compiler flags:

`-q64 -O3 -qhot -qarch=auto -qtune=auto`

supplemented by `-qsmp=omp` for OpenMP programs.

- On the link line we suggest using

`-q64 -bdatapsize:64k -bstacksize:64k`

also supplemented by `-qsmp=omp` for OpenMP programs.

Compiling on SciNet (4): TCS and P7

- ssh to a devel node

`ssh tcs01` or `ssh tcs02` , or `ssh p701`

- Use IBM compilers: `xlc`, `xlC`, `xlF` for C, C++, and Fortran
(`module load vacpp xlF` for the latest version)
- For OpenMP, use `xlc_r`, `xlC_r`, `xlF_r`.
- For MPI, `mpicc`, `mpCC`, `mpxlf` are the mpi wrappers.
- Suggested compiler flags:

`-q64 -O3 -qhot -qarch=auto -qtune=auto`

supplemented by `-qsmp=omp` for OpenMP programs.

- On the link line we suggest using

`-q64 -bdatapsize:64k -bstacksize:64k`

also supplemented by `-qsmp=omp` for OpenMP programs.

Compiling on SciNet (4): TCS and P7

- ssh to a devel node

`ssh tcs01` or `ssh tcs02` , or `ssh p701`

- Use IBM compilers: `xlc`, `xlC`, `xlF` for C, C++, and Fortran
(`module load vacpp xlF` for the latest version)
- For OpenMP, use `xlc_r`, `xlC_r`, `xlF_r`.
- For MPI, `mpicc`, `mpCC`, `mpxlf` are the mpi wrappers.
- Suggested compiler flags:

`-q64 -O3 -qhot -qarch=auto -qtune=auto`

supplemented by `-qsmp=omp` for OpenMP programs.

- On the link line we suggest using

`-q64`

also supplemented by `-qsmp=omp` for OpenMP programs.

Compiling on SciNet (5): ARC

- From `login.scinet.utoronto.ca`, ssh to devel node
`ssh arc01`
- CUDA: The NVIDIA cuda compiler is available (3.2, 4.0, 4.1, 4.2).
You can `module load cuda/<version>`
The compiler is called `nvcc`.
- Optimize for the Tesla GPUs using the compiler flags
`-arch=sm_20 -O3`
- As of version 3.0, OpenCL is included in the CUDA Toolkit.
- CUDA Fortran, OpenACC, OpenMP:
PGI Compilers are in the module `pgi/12.5`, which requires `gcc/4.4.6`.
The compilers are `pgcc`, `pgcpp`, `pgfortran`.
- Debuggers: `cuda-gdb` or `ddt` (`module load ddt`)

Submitting jobs

SciNet=shared resource

- 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!
- Best to 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	time(hrs)	max jobs	max cores
batch	48	32/1000	256/8000 (512/16000 threads)
debug	2/0.5	1	16/64 (32/128 threads)
largemem	48	1	16 (32 threads)
arc	48	4	32 cpu / 3584 gpu

GPC queues

- Submit to these queues from a GPC or ARC 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
```

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

-q: specifies the queue, e.g.

```
-q batch
```

```
-q debug
```

```
-q largemem
```

```
-q arc
```

-I specifies that you want an interactive session.

-X specifies that you want X forwarded.

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

./omp_example > output
```

```
$ qsub scriptname.pbs
```

GPC queues

- HyperThreading: Appears as if there are 16 processors rather than 8 per node. For OpenMP applications this is the default unless `OMP_NUM_THREADS` is set. For MPI, try `-np 16`.
- Always request `ppn=8`, even with hyperthreading.
- *Always test if this is beneficial and feasible!*
- Once the job is incorporated into the queue (this takes a minute), you can use: `showq` to show the queue, and job-specific commands such as `showstart`, `checkjob`, `canceljob`
- 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.

TCS/P7 queues

queue	time(hrs)	max jobs	max cores
verylong	48	2/25	64/800 (128/1600 threads)

Submitting is done with

```
llsubmit <script>
```

and `llq` shows the queue.

- The POWER processors have Simultaneous Multi Threading.
- Once a job is in the queue, you can use `llq` to show the queue, and job-specific commands such as `llcancel`, `llhold`, ...
- *Do not run serial jobs on the TCS!*
- To make jobs start sooner, reduce the `wall_clock_limit` to be closer to the estimated run time (perhaps adding about 10 % to be sure). Shorter jobs are scheduled sooner than longer ones.

Example 1 (GPC)

gpc-f101n084-\$

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

File system

SciNet $\neq$ 4000 $\times$ your pc

- Compute nodes do not contain hard drives!
- The available disk space, /home and /scratch, all part of the GPFS file system which runs over the network.
- 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.
- Don't keep many small files on the system.
They waste space, and are slower to access, read and write.

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.
- Consider using MPI-IO (part of the MPI-2 standard), (parallel) NetCDF, HDF5, or ADIOS.
- 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.
Cannot copy files from the outside world to 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.

Final tips

- Test your job's requirements and scaling behaviour.
Start runs on a small scale and work your way up to larger scales.
- Accurately specify the walltime when you submit a job.
- Avoid reading and writing lots of small amounts of data to disk.
- Do not create lots of files.
- Do not submit single serial jobs.
- Do not keep lots of files in your directory (use tar).
- Read the SciNet User Guide
http://wiki.scinethpc.ca/wiki/images/5/54/SciNet_Tutorial.pdf

Useful web sites

www.scinethpc.ca

SciNet | Science At Scale - Mozilla Firefox

www.scinethpc.ca

SciNet

Home SciNet for Me In The News SciNet Blogs About SciNet Events Other SciNet Links Search ...

SciNet and the Discovery of the Higgs Boson

“SciNet is absolutely central to make anything out of what happens,” Teuscher [a University of Toronto ATLAS Researcher] said in this [Toronto Star](#) article. Want to learn more about computation and the Higgs? This PC Advisor article has a very good overview of the massive data challenges that the worlds largest scientific experiment faces, and [...]

SciNet and the Discovery of the Higgs Boson

SciNet to Help Drive Ontario Innovation with Largest Computer

Canadian Supercomputers Assigned Homework

SciNet Image Gallery

SciNet is Canada's largest supercomputer centre, providing Canadian researchers with computational resources and expertise necessary to perform their research on scales not previously possible in Canada. SciNet powers work from the biomedical sciences and aerospace engineering to astrophysics and climate science. SciNet is part of Compute/Calcul Canada, a national infrastructure for supercomputing-powered innovation composed of seven consortia, and is funded by the Canada Foundation for Innovation, NSERC, the Government of Ontario, and the University of Toronto.

Find out more [about us](#), how to [contact us](#), or even how to start using SciNet resources.

If you're [a researcher](#), [an educator](#), or [work in industry](#), check out what SciNet can do for you!

SciNet is a member of Compute/Calcul Canada

compute • calcul CANADA

compute • calcul CANADA

Useful web sites: Wiki

wiki.scinethpc.ca

SciNetWiki - Mozilla Firefox

wiki.scinethpc.ca/wiki/index.php/SciNet_User_Support_Library

142.150.198.54 Talk for this IP address Log in

Page Discussion Read View source View history Go Search

SciNet User Support Library

System Status: **UP**

Tue Jun 26 18:49:54 EDT 2012 ([Previous messages](#))

QuickStart Guides

- [SciNet User Tutorial](#)
- [Tutorials and Manuals](#)
- [Essentials](#)
- [GPC: General Purpose Cluster](#)
- [TCS: Tightly Coupled System](#)
- [GPU nodes of the Accelerator Research Cluster](#)
- [P7: Power 7 Linux cluster](#)
- [Software and libraries](#)
- [Data management](#)
- [FAQ \(frequently asked questions\)](#)
- [Acknowledging SciNet](#)

What's New On The Wiki

- [GPU Cluster](#) now integrated into GPC scheduler (Jul 5)
- [PGI compilers](#) supporting OpenACC and Cuda Fortran installed on the [GPU Cluster](#) (Jul 4)
- [PGI compiler manuals](#) added to the [Tutorials and Manuals](#) page (Jul 4)
- [Ontario Summerschool on High Performance Computing Central](#) slides and code (Jun 29)
- [TechTalk Remote Development](#) slides (Jun 13)

Previous new stuff can be found in the [What's new archive](#).

User-Supplied Content

Share your expertise with the SciNet Community!

- [Running specific codes on SciNet machines](#)
- [Tips for performance or convenience on SciNet machines](#)
- [Running serial batch jobs \(including GNU parallel\)](#)
- [Using the ramdisk on SciNet's GPC](#)
- [User space nrd modules](#)

News and Recent Events

- Jun 7, 2012: SciNet Shutdown, at least till 10PM.
- Jun 13, 2012: [June SNUG](#) 📢 TechTalk by Ramses van Zon: [Remote Development on SciNet](#)
- Jun 25-28, 2012: [Ontario Summerschool for High Performance Computing - Central](#), hosted by SciNet 📢
- Jul 9, 2012, [Intro to SciNet](#) 📢

Other SciNet Websites

- [SciNet website](#)
- [SciNet courses](#)
- [SciNet portal](#)

Systems

- [GPC cluster](#)
- [TCS cluster](#)
- [GPU cluster](#)
- [P7 cluster](#)
- [HPSS storage](#)

Knowledge Base

- [SciNet essentials](#)
- [Tutorials/Manuals](#)
- [FAQ](#)
- [Software](#)
- [Moab scheduler](#)
- [Data management](#)
- [GPC MPI versions](#)

Wiki

- [Recent changes](#)
- [Random page](#)
- [Wiki help](#)

Toolbox

Useful web sites: Courses

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

SciNet Courses | - High Performance Education - Mozilla Firefox

SciNet Courses - High Performance Education

My account
Create content
Administer
Log out

Event Calendar

« July 2012 »

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

Current signups for rzon

You have not signed up for any classes yet.

Other Resources

Intro to SciNet

Short Course

Monday, July 9, 2012 - 12:00 — Monday, July 9, 2012 - 13:30

A class of approximately 90 minutes where you will learn how to use the systems. Experienced users may still pick up some valuable pointers during these sessions.

You may find the SciNet User Tutorial useful: http://wiki.scinethpc.ca/wiki/images/5/54/SciNet_Tutorial.pdf

Location: SciNet offices at 256 McCaul Street, 2nd Floor.

Sign up here: [signup form](#).

September SNUG

SNUG meeting

Wednesday, September 12, 2012 - 12:00 — Wednesday, September 12, 2012 - 13:00

The SciNet Users Group (SNUG) meetings are every month on the second Wednesday, and involve pizza, user discussion, feedback, and a half-hour talk on topics or technologies of interest to the SciNet community.

Location: SciNet offices at 256 McCaul Street, 2nd Floor.

Upcoming events

- Intro to SciNet (3 days)
- September SNUG (68 days)
- October SNUG (96 days)
- November SNUG (131 days)
- December SNUG (159 days)

[more](#)

Poll

What time of year would be best for our week-long Practical Parallel Programming intensive?
Summer (As currently)

73%

Fall

Net e+calcul NADA

Useful web sites: Portal

<https://portal.scinet.utoronto.ca>

SciNet usage reports

Change password, default allocation, maillist subscriptions



Exception: gpgpu mail list:

<https://support.scinet.utoronto.ca/mailman/listinfo/scinet-gpgpu>