SHARCNET Literacy:
User I Introduction

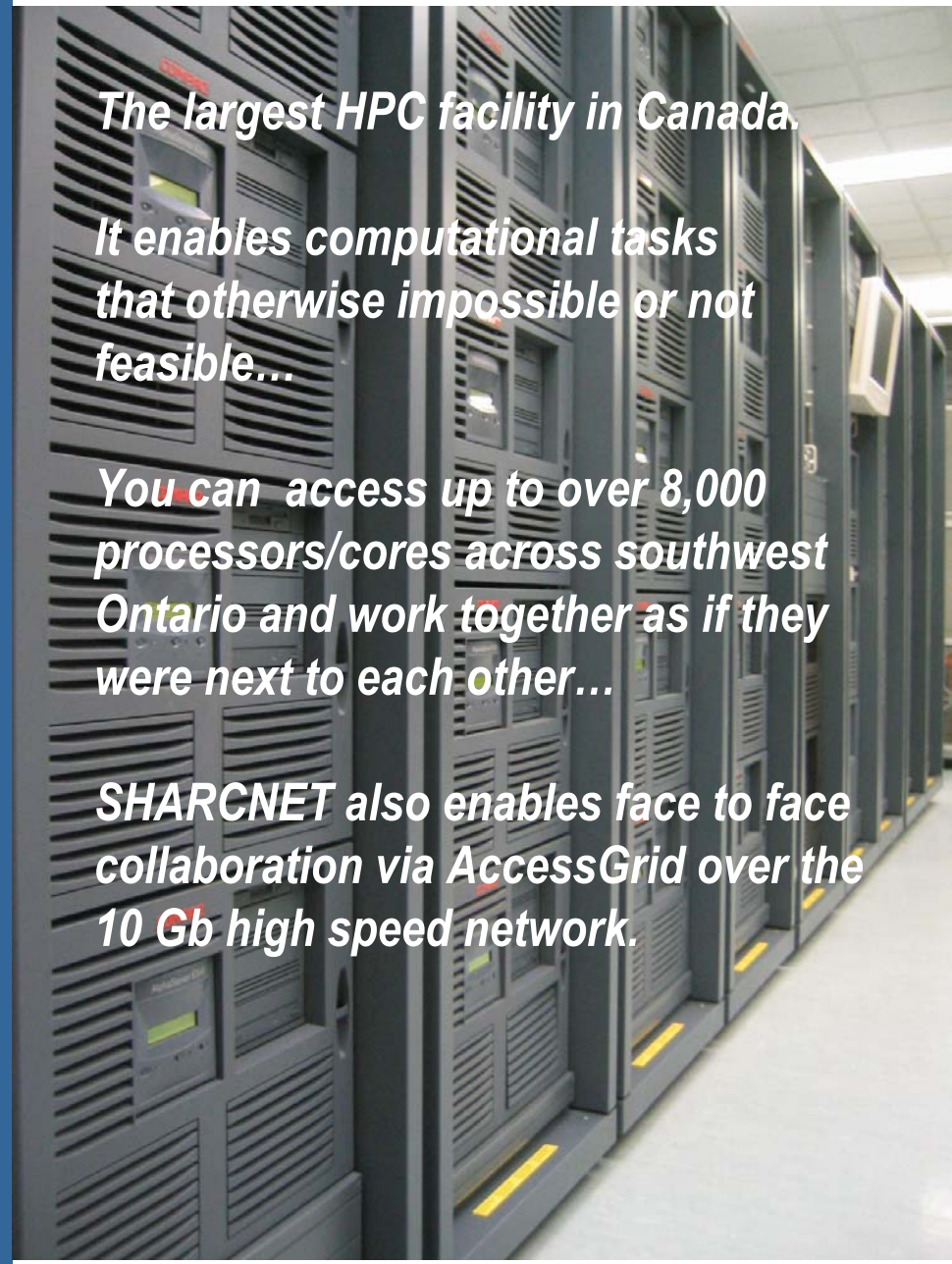
Supercomputing Environment at Your Institution and Beyond

Via AccessGrid, April 23, 2007

Agenda

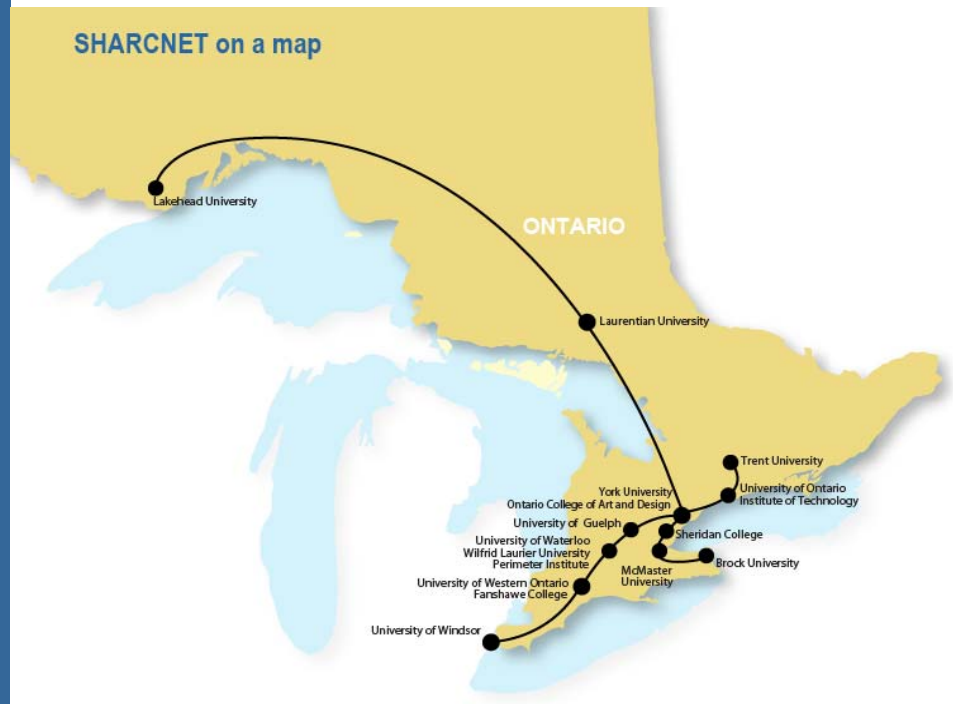
- What is SHARCNET
- Where to find Information
- SHARCNET essentials
- Support
- Parallel Computation using MPI
- Parallel and Concurrent Computation using OpenMP

SHARCNET™

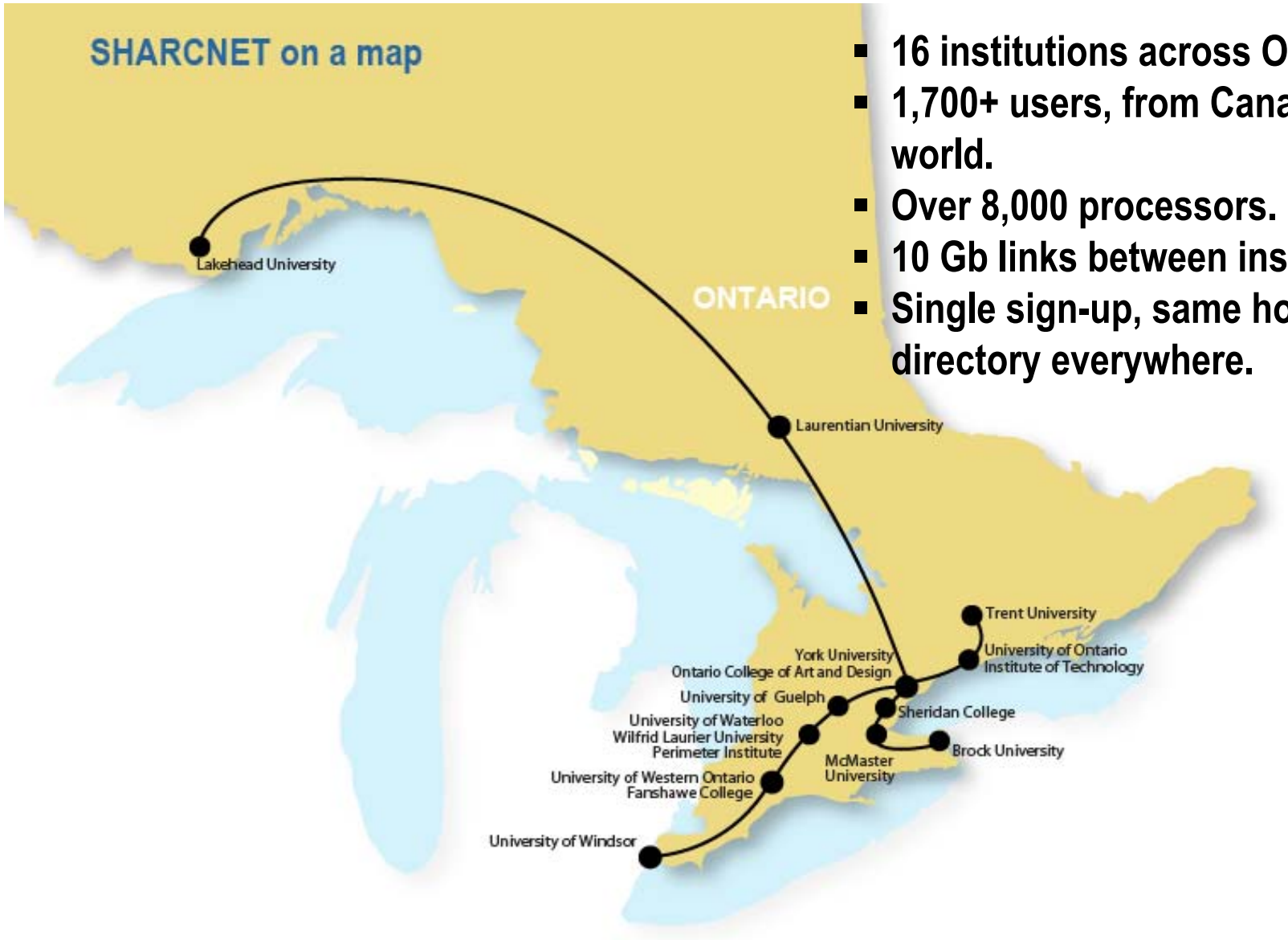


What is SHARCNET

- A consortium
- A cluster of clusters of high performance, networked supercomputer systems
- Visions and missions



SHARCNET Is A Supercomputing Consortium…



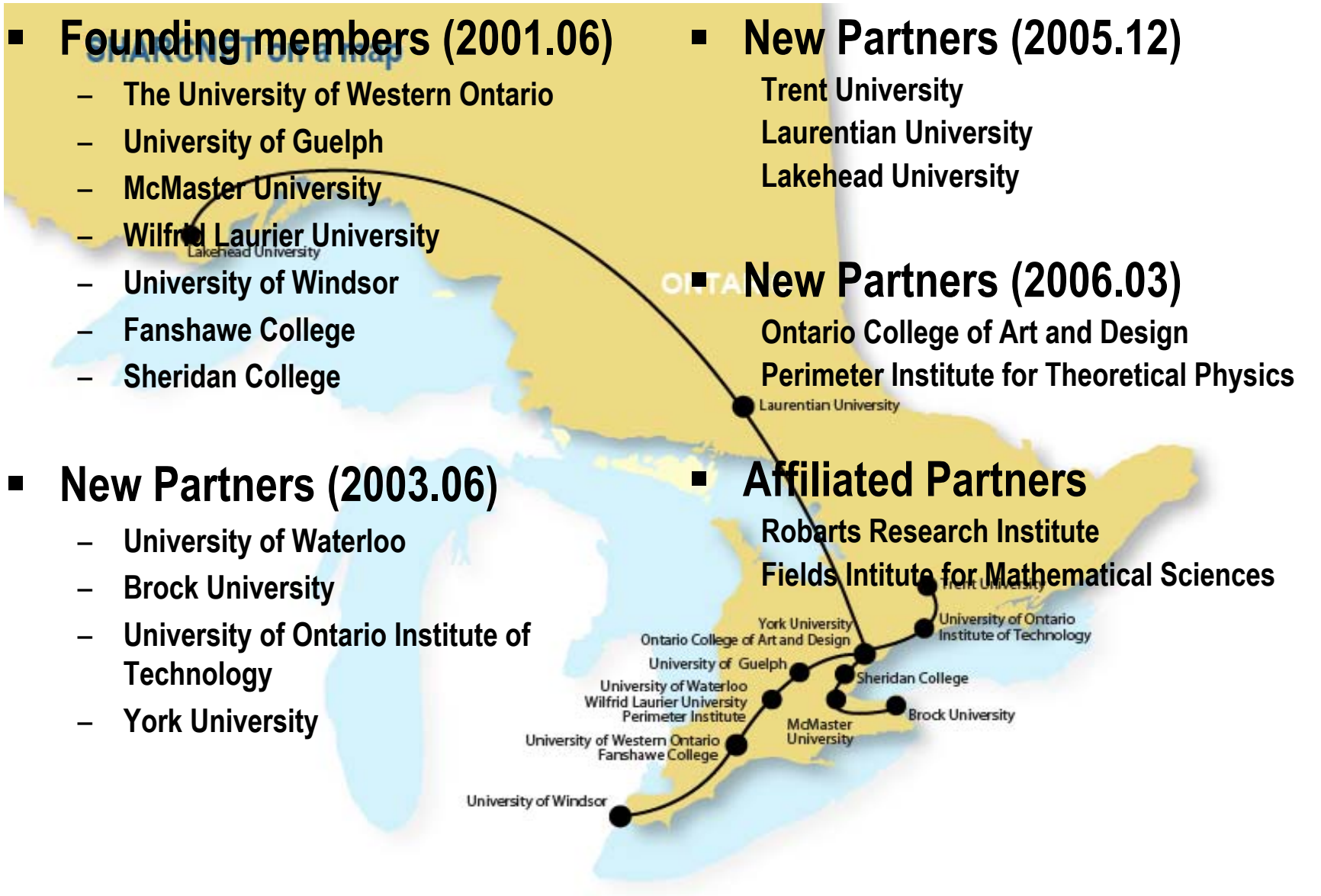
- 16 institutions across Ontario.
- 1,700+ users, from Canada and world.
- Over 8,000 processors.
- 10 Gb links between institutions.
- Single sign-up, same home directory everywhere.

Vision, Mission and Goals

- **The SHARCNET Vision**
  - *To become a world leading, academic high-performance computing consortium enabling forefront research and innovation*.
- **The SHARCNET Mission**
  - *To promote and facilitate the use of high performance computational techniques among researchers in all fields and to create a new generation of computationally-aware individuals*.
- **General Goals**
  - provision of otherwise unattainable compute resources
  - reduce time to science
  - remote collaboration

Founding Members, Partners since 2001

- **Founding members (2001.06)**
  - The University of Western Ontario
  - University of Guelph
  - McMaster University
  - Wilfrid Laurier University
  - University of Windsor
  - Fanshawe College
  - Sheridan College
- **New Partners (2003.06)**
  - University of Waterloo
  - Brock University
  - University of Ontario Institute of Technology
  - York University
- **New Partners (2005.12)**
  - Trent University
  - Laurentian University
  - Lakehead University
- **New Partners (2006.03)**
  - Ontario College of Art and Design
  - Perimeter Institute for Theoretical Physics
- **Affiliated Partners**
  - Robarts Research Institute
  - Fields Intitute for Mathematical Sciences



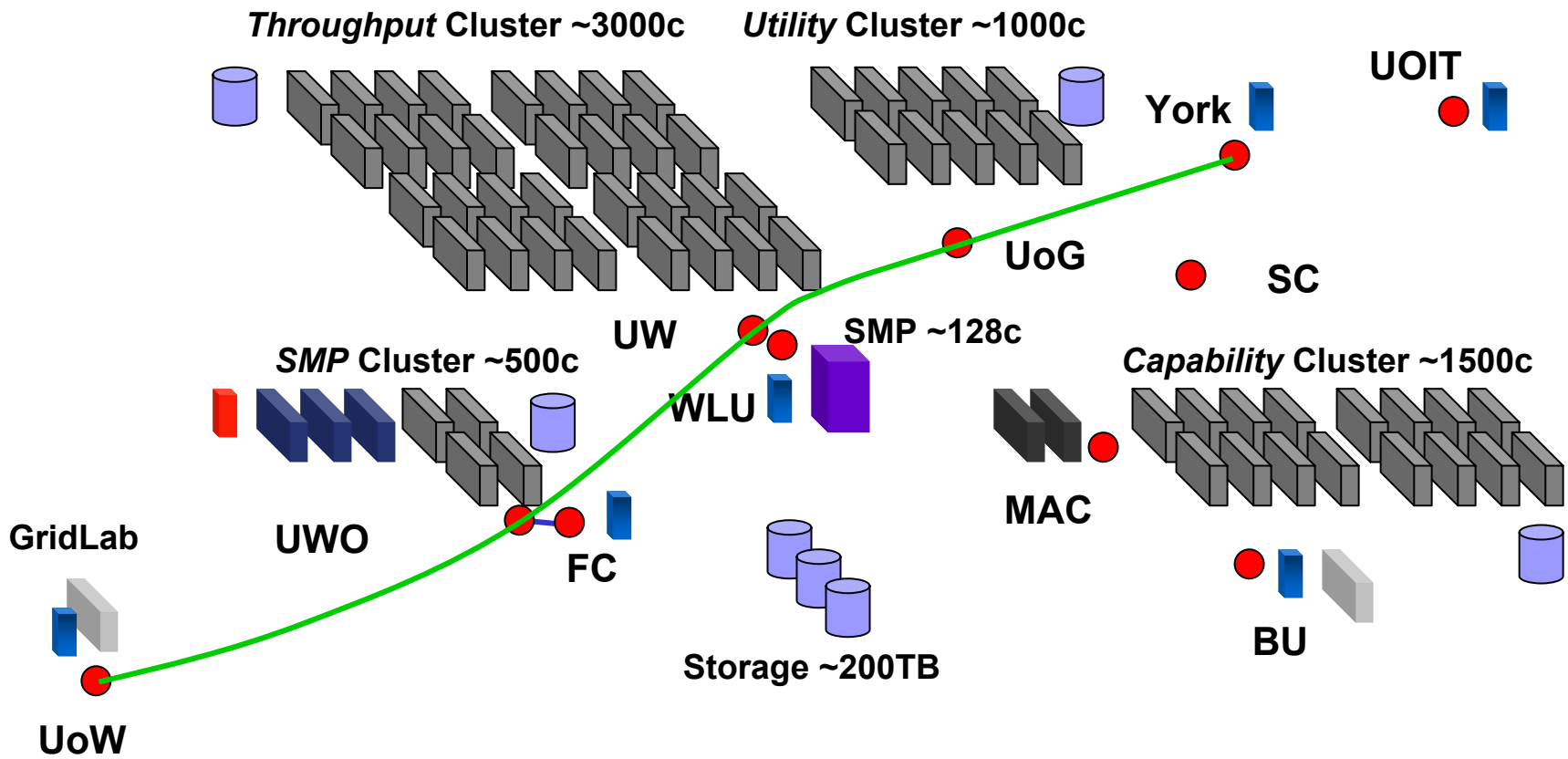
Industry and Government Partners

- **Industry**
  - Hewlett Packard
  - SGI
  - Quadrics Supercomputing World
  - Platform Computing
  - Nortel Networks
  - Bell Canada

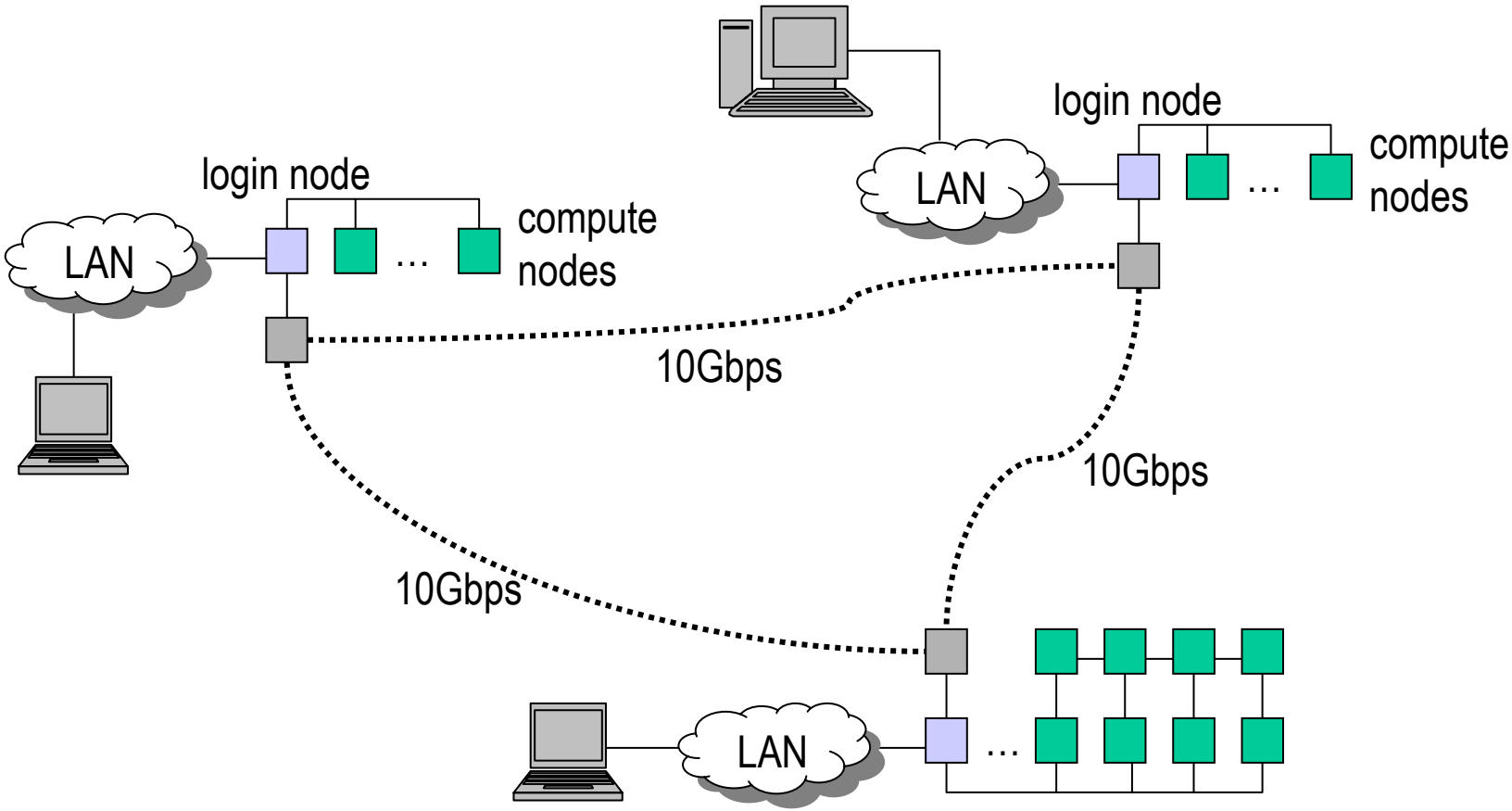
- **Government**
  - Canada Foundation for Innovation
  - Ontario Innovation Trust
  - Ontario R&D Challenge Fund
  - Optical Regional Advanced Network of Ontario (ORANO)

Resources at A Glance

- Over 8,000 processors deployed on 11 sites since 2005.



Cluster of Clusters



SHARCNET Facilities

- **Computers**
  - Clusters (**distributed memory**): for parallel and serial programs
    - Super fast interconnect
    - Fast interconnect
    - Fast interconnect and SMP nodes
    - Serial farm
  - Symmetric multiprocessing (SMP) systems (**shared memory**): for parallel, threaded applications.
- **Visualization Clusters**
  - Being deployed at some institutions.
- **Access Grid Rooms** (Multi-media)
  - Video conference, cross site workshops, etc.

SHARCNET Basics

- **FREE to academic researchers**

- **Compute-Intensive Problems**
  - The resources are provided to enable HPC and are not intended as a replacement for a researcher's desktop or lab machines.
  - SHARCNET users can productively conduct HPC research on a variety of SHARCNET systems each optimally designed for specific HPC tasks

- **Academic HPC research**
  - The research can be business-related, but must be done in collaboration with an academic researcher

- **Fairness access**
  - Users have access to all systems
  - Clusters are designed for certain type of jobs
  - Job runs in batch mode (scheduling system) with fairshare

Online Resource Discovery

SHARCNET: Facilities - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

http://www.sharcnet.ca/Facilities/index.php

Customize Links Free Hotmail Windows Marketplace Windows Media Windows

Shared Hierarchical Academic Research Computing Network

SHARCNET™

Facilities > Systems Login by: jemmyhu

Main Menu
Home
About
Research
Facilities
Systems
Network
Statistics
Contributions
Software
Policies
Photo Gallery
Performance
Help
FAQs
Contact
Media
Events
Careers

Personal
My Login Page
Submit Problems
View Problems
My Account
My Profile
My Projects
My Publications
My Mailing Lists

Features
What's New
Documents
SHARCwiki

SHARCNET Facilities at a Glance

System	State	CPUs	Architecture	Nodes	Notices
bala	Online	128	Cluster/Myrinet G2	Opteron	Aug 25 2006
bruce	Online	128	Cluster/Myrinet	Opteron	Aug 24 2006
bull	Online	384	Cluster/Quadrics Elan4	Opteron	Aug 04 2006
cat	Online	160	Cluster/Gigabit Ethernet	Xeon, Opteron	Jun 14 2006
coral	Offline	60	Cluster/Quadrics (Elan 3)	Itanium2	Jun 13 2005
dolphin	Online	128	Cluster/Myrinet G2	Opteron	Sep 08 2006
goblin	Online	54	Cluster/Gigabit Ethernet	Opteron	Jul 31 2006
greatwhite	Online	456	Cluster/Quadrics (Elan 3), Gigabit Ethernet	Alpha	Sep 07 2006
gulper	Offline	44	Cluster/Myrinet	Opteron	Jul 27 2006
mako	Testing	14	Cluster/Myrinet, Gigabit Ethernet	Xeon	Aug 30 2006
megaladon	Online	128	Cluster/Myrinet G2	Opteron	Sep 06 2006
narwhal	Online	1068	Cluster/Myrinet	Opteron	Aug 30 2006
requin	Online	1536	Cluster/Quadrics Elan4	Opteron	Sep 06 2006
silky	Online	128	SMP/NUMA	Itanium2	Aug 04 2006
spinner	Offline	70	Cluster/Gigabit Ethernet	Itanium2	Aug 22 2006
tiger	Offline	128	Cluster/Myrinet G2	Opteron	Jul 25 2006
typhon	Offline	16	SMP/SMP	Alpha	Feb 09 2006
whale	Online	3072	Cluster/Gigabit Ethernet	Opteron	Sep 11 2006
wobbe	Online	208	Cluster/Myrinet, Gigabit Ethernet	Opteron	Jun 14 2006
zebra	Online	128	Cluster/Myrinet G2	Opteron	Sep 05 2006
Totals		8038			

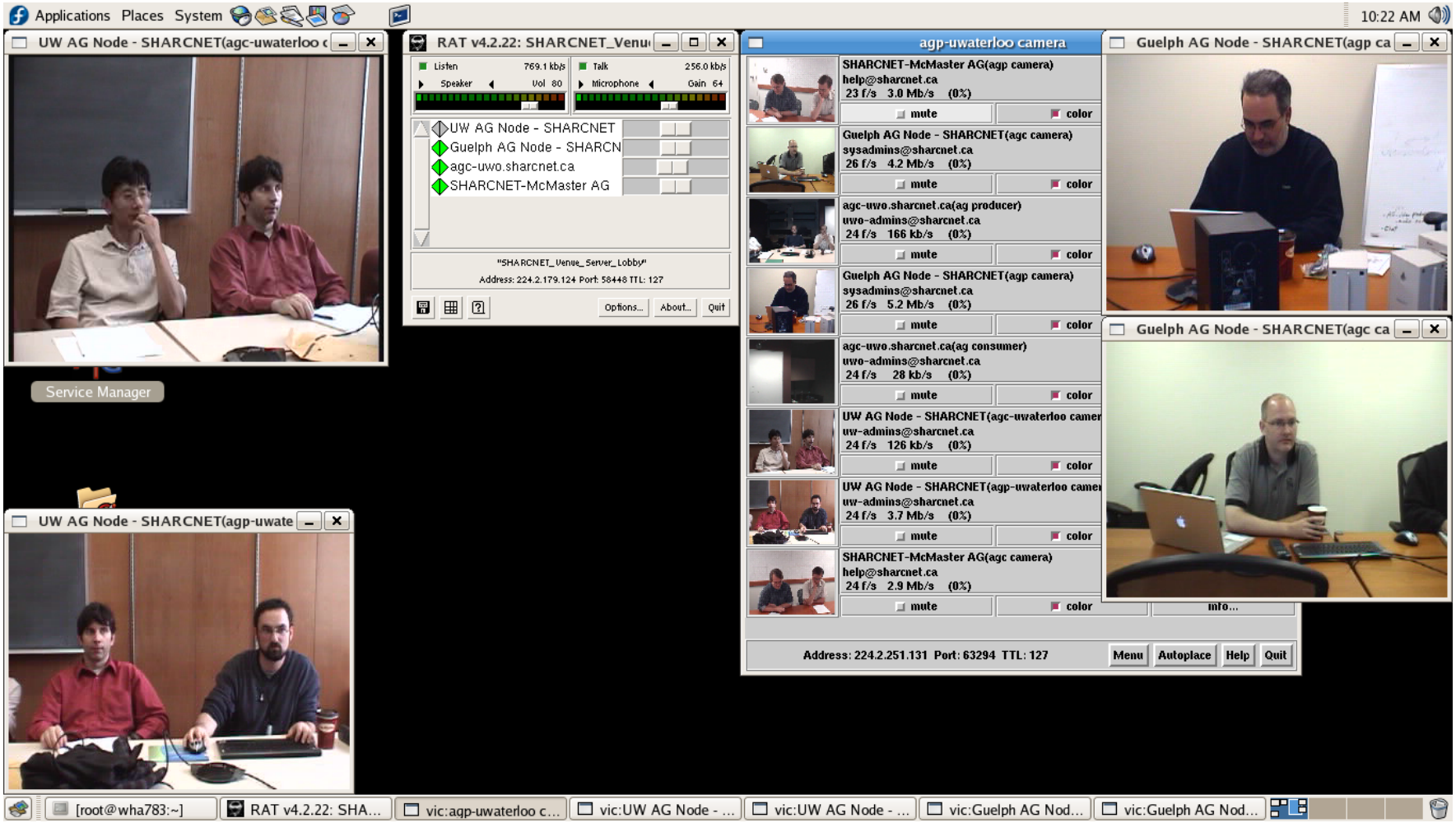
(Report a problem.)
(List systems and sites.)

Done

Examples of Systems

System	CPUs	Memory	Storage	Interconnect	Intend Use
requin (Capability)	1536	8 GB	70 TB	Quadrics	Large scale MPI
narwhal (Utility)	1068	8 GB	70 TB	Myrinet	MPI, SMP
whale (Throughput)	3072	4 GB	70 TB	GigE	Serial
bull (SMP-friendly)	384	32 GB	70 TB	Quadrics	High RAM/BW MPI & SMP
silky (SMP)	128	256 GB	4 TB	NUMAlink	OpenMP/pthreads, large memory SMP
PoPs (6x)	128	8 GB	4 TB	Myrinet	Genaral purpose

Collaboration via Access Grid



SHARCNET's Position in The World

Position on Top 500 List	Installation Site	Computer Model	Number of processors	Rmax	Rmax/Rmax(10)	Area of Installation
66 (previously 51)	RQCHP (Sherbrooke)	3.6 GHz Xeon + Infiniband	1152	6888	0.19	Academic Research
80	SHARCNET (Waterloo)	2.2 GHz Opteron + Gigabit Ethernet	3072	6015	0.17	Academic Research
83	SHARCNET (McMaster)	2.6 GHz Opteron + Quadrics	1536	5746	0.16	Academic Research
146 (previously 106)	WestGrid (UBC)	3.06 GHz Intel Xeon + Gigabit Ethernet	1680	3755	0.10	Acadmic Research
165 (previously 115)	Telco	3.2 GHz Intel Xeon + Gigabit Ethernet	1036	3755	0.10	Industrial
209	SHARCNET (Guelph)	2.2 GHz (Dual core) Opteron + Myrinet	1068 cores	3491	0.10	Academic Research
273 (previously 157)	RQCHP (Sherbrooke)	3.2 GHz Intel Xeon + Gigabit Ethernet	872	3064	0.09	Academic Research
385 (Previously 240)	Canadian Meterological Centre, Dorval	IBM p690	960	2560	0.07	Research (Weather)


CONSTRUCTION SITE
MUST BE WORN ON THIS SITE

Where to Find Information

- Account
- Resource discovery
- People

Getting An Account

- **Apply for an account online**
  - Account must be applied online
  - Students, postdoc, visiting fellows must have a **sponsor** who has an account.
- **Account approval:**
  - Faculty accounts are approved by the **site leader**
  - Students/postdoc/fellows require a faculty **sponsor**, who shall approve such accounts
  - Non-SHARCNET institution accounts are approved by the **Scientific Director**
- You will have a ***webportal*** *account* that allows you to access information/files, submit requests and manage your own profile.

One Account, Access to All Systems!

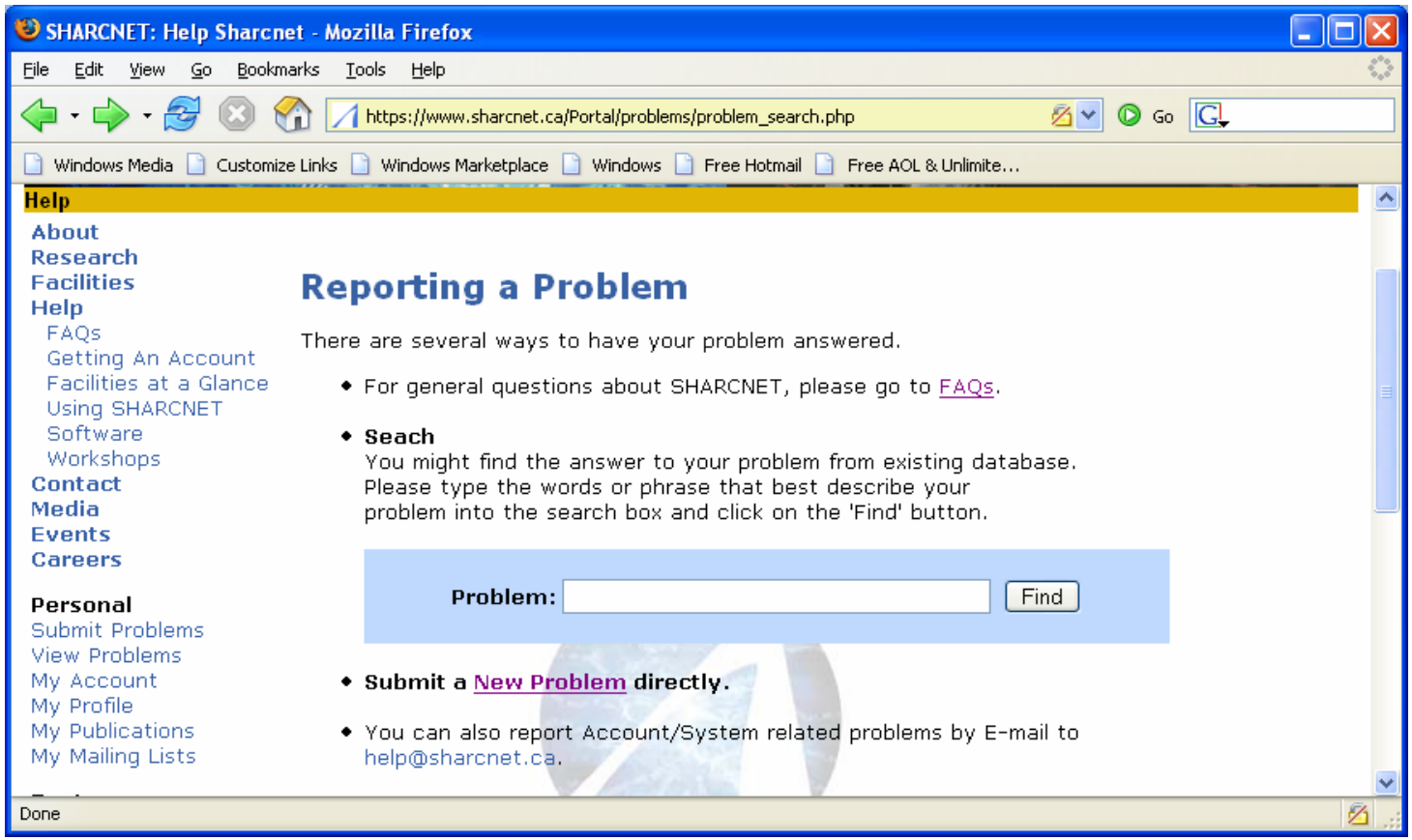
- **Login to systems**
  - Siteless login – single username/password for all systems.
  - Same user home directory across all systems.
  - Systems are designed and deployed for different purposes:
    - **parallel applications**,
    - **serial applications**, e.g. large number of **serial case runs**,
    - **threaded applications** that make use of shared memory by threads, etc
- **Login to web**
  - Discovery resources.
  - See statistics.
  - Users can change password on the web.
  - Report and keep track of problems.

Where to Look for Information

- **FAQs** are on the web. Go to SHARCNET web site
- **Weekly online seminars** on every Monday
- **Education Online** – slide, examples from past workshops are also available on the web on the **Help** page.
- Information on individual systems are available on the web on the **Facilities** page.

How to Contact Us

- E-mail us. Our contact info is listed on the **Contact** page at
- Call us.
- Use **Problem Tracking** in the web portal.



SHARCNET Essentials

- Computing environment
- Moving, editing files
- Compiling programmes
- Software and libraries
- Running programmes in batch mode - Queuing system and commonly used commands

Computing Environment

- **Systems** – Cluster and SMPs.
- **Operating systems** – Linux, Tru64, all 64-bit.
- **Languages** – Fortran, C/C++, Java, Matlab, etc.
- **Compilers**
  - SHARCNET unified compilation environment: **cc**, **c++**, **f77/90**, **mpicc**, **mpif90**.
  - Underlying compilers: **PathScale**, **PGI**, **Intel**, **Compaq**, **gcc**, **g++** and **gfortran**.
- **Key parallel development support**
  - MPI (hp-MPI, SGI MPI, Quadric MPI, MPICH, OpenMPI, etc.)
  - Pthreads, OpenMP
- **Batch scheduling**
  - **sq** - SHARCNET unified batch execution environment.
  - Underlying scheduler: **LSF** – Platform's *Load Sharing Facility*

Facilities: Intended Use

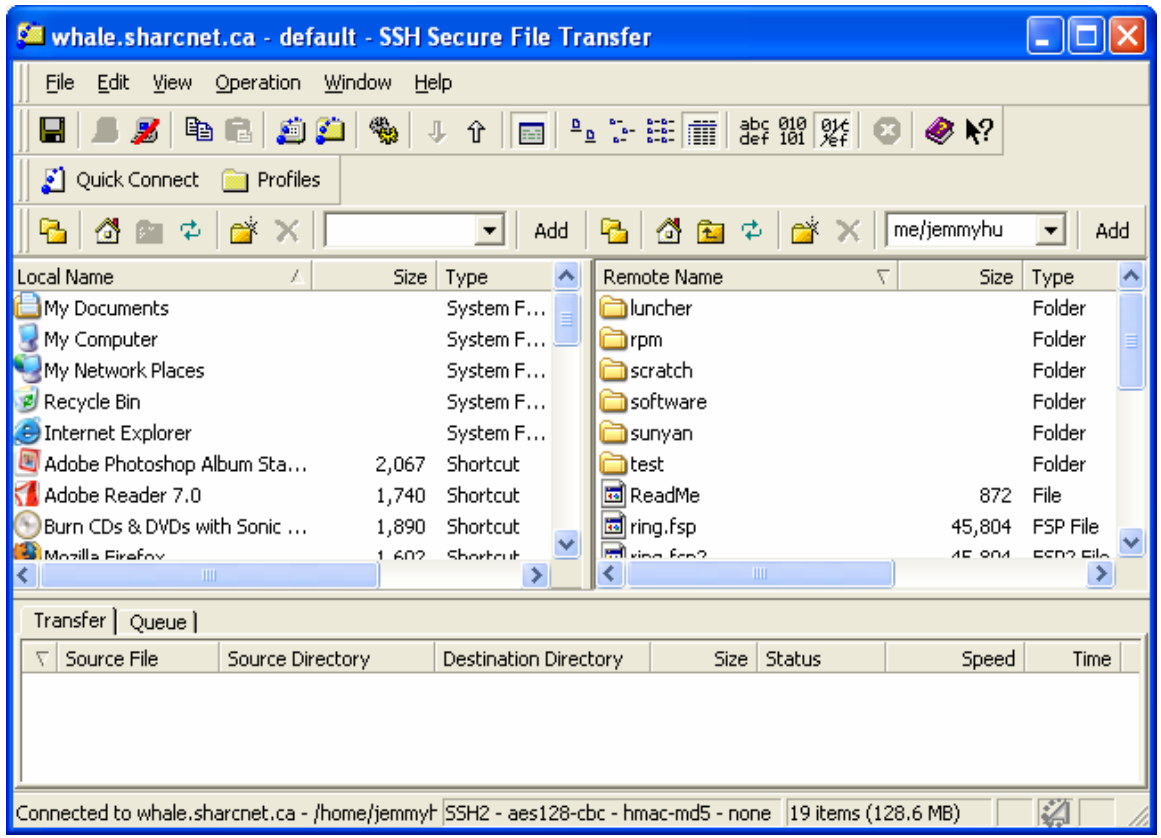
Cluster	CPUs	RAM /node	Storage	Interconnect	Intended Use
requin (Capability)	1536	8 GB	70 TB	Quadrics	Resource intensive MPI (fine grained, large mem.)
narwhal (Utility)	1068	8 GB	70 TB	Myrinet (GM)	MPI, small-scale SMP
whale (Throughput)	3072	4 GB	70 TB	GigE	Serial
bull (SMP-friendly)	384	32 GB	70 TB	Quadrics	High RAM/BW MPI & small-scale SMP
silky (SMP)	128	256 GB	4 TB	NUMAlink	large memory/medium-scale SMP
bala, bruce, dolphin, megaladon, tiger, zebra	128	8 GB	4 TB	Myrinet (GM)	Genaral purpose
mako	16	2GB	200GB	Myrinet (MX)	Development/testbed

How One Typically Works

- Login to a system via SSH, you see familiar UNIX environment.
- Edit source code and/or change the input data/configuration file(s).
- Compile source code.
- Submit a program (or many) to batch queuing system.
- Check results in two days☺

File Transfer from/to Your Desktop

- UNIX
  - User **scp** or **sftp**
- Windows
  - User **putty**, or
  - **SSH Secure File Transfer/Shell**



Compiling Programmes

- SHARCNET provides a unified compiling environment that chooses the right underlying compiler, options and libraries for you! Use them always unless you know better.

Command	Language	Extension	Example
cc	C	c	**cc** code.c –o code.exe
CC, c++, cxx	C++	.C, .cc, .cpp, .cxx, c++	**CC** code.cpp –o code.exe
f77	Fortran 77	.f, .F	**f77** Fcode.f –o Fcode.exe
f90/f95	Fortran	.f90, .f95, .F90, F95	**f90** Fcode.f90 –o Fcode.exe
mpicc	C	c	**mpicc** mpicode.c –o mpicode.exe
mpiCC	C++	C++	**mpiCC** mpicode.cc –o mpicode.exe
mpif77	Fortran 77	f77	**mpif77** mpicode.f –o mpicode.exe
mpif90/mpif95	Fortran	f90/f95	**mpif90** mpicode.f90 –o mpicode.exe

Common Compiler Options

- There are minor differences between compilers (see man page for details), e.g. Pathscale:

-c	Do not link, generate object file only.
-o *file*	Write output to specified *file* instead of default.
-I*path*	Add *path* to search path for include files.
-l*library*	Search the library named liblibrary.a or liblibrary.so such as –lmpi, -lacml
-L*path*	Add *path* to search path for libraries
-g[N]	Specify the level of debugging support produced by the compiler
-g0	No debugging information for symbolic debugging is produced. This is the default.
-g2	Produce additional debugging information for symbolic debugging.
-O[n]	Optimization level n=0 to 3. Default is -**O2**.
-O0	Turns off all optimizations.
-O1	Turns on local optimizations that can be done quickly.
-O2	Turns on extensive optimization. This is the default
-O3	Turns on aggressive optimization (e.g. loop nest optimizer).
-Ofast	Equivalent to -O3 -ipa -OPT:Ofast -fno-math-errno
-pg	Generate extra code to profile information suitable for the analysis program pathprof
-Wall	Enable most warning messages.

Development Tools

- **Libraries**
  - ACML, ATLAS, CXML, ScaLAPACK, **MKL (Intel), MLIB (hp), IMSL** (VNI), etc.
  - PETSc.
  - *…requested by users.*
- **Application Packages**
  - Blast, Gromacs, NWChem, Octave, R, …
- **Commercial Packages**
  - **Gaussian** (several places, site license required).
  - **Fluent.**
- **Development tools**
  - Debugging: DDT, gdb, …
  - Profiling/Optimization: OPT, Altix toolkit (SGI), gprof,

File System Basics

- **Policy**
  - Same username/password across all systems, and web account.
  - Common home directory across SHARCNET (exceptions: wobbe, cat)
  - Common SHARCNET wide software are in /opt/sharcnet
  - /home backup
- **File system**

pool	quota	expiry	purpose
/home	200 GB	none	Source, small configuration files
/scratch	None	none	Active data files, binaries
/work	none	none	Active data files
/tmp	160 GB	10 days	Node-local scratch

  - /scratch and /work are cluster dependant, backup by users
  - **Important:** run jobs on /scratch or /work

The Batch Scheduling System

- All access to resources managed by queuing system.
- Programs are submitted to queues to run using **sqsub** command:

  sqsub –q *qname* [options] ./myprog [arg1 [,…]]
- By default results will be mailed to you afterwards. But you may choose to have all outputs be saved in a disk file with an **–o *output*** option. This is strongly encouraged.

Queues

- Specific job queues have different priorities and constraints (**bqueues** command):
  - **mpi** – for parallel jobs.
  - **serial** – for serial jobs.
  - **threaded** – for jobs that use threads.
  - **test** – for test purpose.
  - *special queues* on some systems for running special packages, such as **GAUSSIAN**, **FLUENT**.

Commonly Used Batch Commands

- **bqueues** – list available queues.
- **sqsub** – submit a program ("job") to a specific queue.
- **sqjobs** – list the status of submitted jobs.
- **sqkill** – kill a program by job ID.
- **bhist** – list history of jobs.

- **bqueues** – list available queues

```
[bge@nar316 ~]$ bqueues
QUEUE_NAME      PRIO STATUS          MAX JL/U JL/P JL/H NJOBS   PEND    RUN   SUSP
staff            150  Open:Active       -    -    -    -     0      0      0      0
test             100  Open:Active       -    -    -    -     0      0      0      0
threaded          80  Open:Active       -    -    -    -     6      0      6      0
mpi               80  Open:Active       -    -    -    -  1972    914    994     64
serial            40  Open:Active       -    -    -    -    25      0     10     15
```

- **sqsub** – submit a program (job) to a queue, which will be executed later by the scheduler.

```
[bge@nar316 ~]$ sqsub -q serial hostname
Job <136670> is submitted to queue <serial>.

[bge@nar316 ~]$ sqsub -q mpi -n 20 -o my_mpi_prog.stdout ./my_mpi_prog
Job <136671> is submitted to queue <mpi>.
```

- **sqsub –t** – submit a program (job) to the test queue, which will be started sooner (< 60 seconds), for **short**, test runs.

```
[bge@nar316 ~]$ sqsub -q serial -t hostname
Job <136670> is submitted to queue <serial>.

[bge@nar316 ~]$ sqsub -q mpi -t -n 20 -o my_mpi_prog.stdout ./my_mpi_prog
Job <136671> is submitted to queue <mpi>.
```

- **sqsub –man** – display command options

- **sqjobs** – list the status of submitted jobs.

```
[bge@nar316 ~]$ sqjobs
 jobid user queue state ncpus time command
------ ---- ----- ----- ----- ---- -------------
136671  bge   mpi     R    20   0s ./my_mpi_prog
1060 CPUs total, 30 idle, 1030 busy; 43 jobs running; 16 suspended, 12 queued.
```

- **sqkill** – kill a job in the queue that you want to stop.

```
[bge@nar316 ~]$ sqjobs
 jobid user queue state ncpus time command
------ ---- ----- ----- ----- ---- -------------
136672  bge   mpi     Q     1   0s ./my_mpi_prog
1060 CPUs total, 50 idle, 1010 busy; 42 jobs running; 16 suspended, 13 queued.


[bge@nar316 ~]$ sqkill 136672
Job <136672> is being terminated
```

- **bhist** – list jobs history

bhist [-a | -d | -p | -r | -s] [-b | -w] [-l] [-t] …

Options:

-a	displays finished and unfinished jobs (over-rides -d, -p, -s and -r)
-b	brief format; if used with -s option, shows reason why jobs were suspended
-d	only display finished jobs
-l	long format; displays additional information
-u user	display jobs submitted by specified user

- **bhist** – A snapshot of command output

```
nar317:~/pub/exercises% bhist -a
Summary of time in seconds spent in various states:
JOBID   USER    JOB_NAME  PEND    PSUSP   RUN   USUSP   SSUSP   UNKWN   TOTAL
134177  dbm     *o_mpi_c  8       0       37    0       0       0       45
134227  dbm     *o_mpi_c  10      0       10    0       0       0       20
nar317:~/pub/exercises% bhist -l 134177
Job <134177>, User <dbm>, Project <dbm>, Job Group </dbm/dbm>,
                     Command </opt/hpmpi/bin/mpirun -srun -o mpi_hello.log ./
                     mpi_hello>
Fri Sep 15 13:06:08: Submitted from host <wha780>, to Queue <test>, CWD <$HOME/
                     scratch/examples>, Notify when job ends, 4 Processors Requ
                     ested, Requested Resources <type=any>;
Fri Sep 15 13:06:16: Dispatched to 4 Hosts/Processors <4*lsfhost.localdomain>;
Fri Sep 15 13:06:16: slurm_id=318135;ncpus=4;slurm_alloc=wha2;
Fri Sep 15 13:06:16: Starting (Pid 29769);
Fri Sep 15 13:06:17: Running with execution home </home/dbm>, Execution CWD
                     </scratch/dbm/examples>, Execution Pid <29769>;
Fri Sep 15 13:06:53: Done successfully. The CPU time used is 0.3 seconds;
Fri Sep 15 13:06:57: Post job process done successfully;
Summary of time in seconds spent in various states by  Fri Sep 15 13:06:57
  PEND     PSUSP    RUN      USUSP    SSUSP    UNKWN    TOTAL
  8                 0        37       0        0        0        45
```

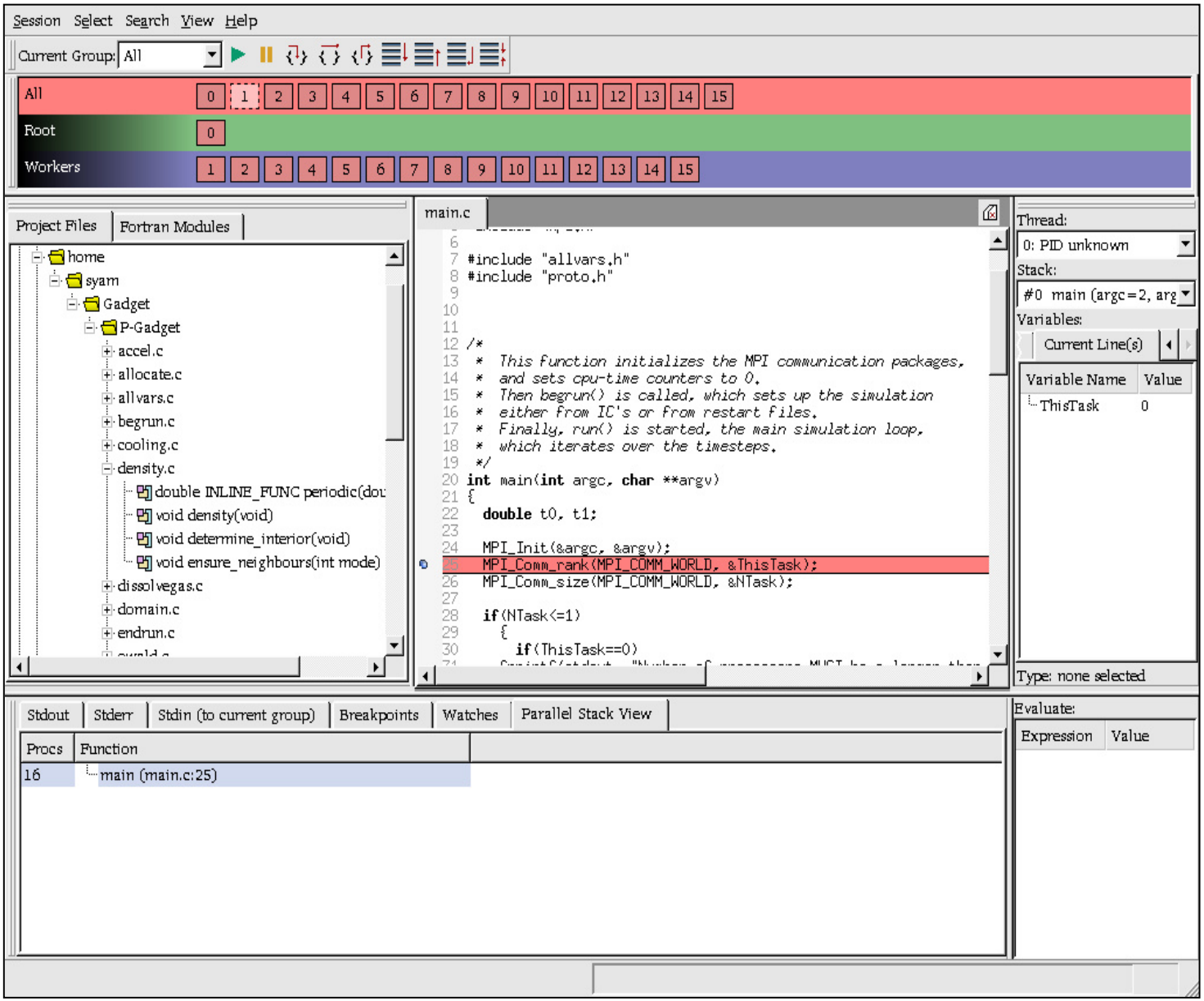
Debugging Tools: gdb, idb, DDT

- DDT is a powerful debugger for parallel programs with GUI
  - works best with MPI programs, but can also be used for threaded and serial jobs
  - supports with C, C++ and many flavours of Fortran (77, 90, 95)

- Installed on requin, narwhal, bull and six PoP clusters

- To use DDT:
  - **ddt program [arguments]**
  - then choose number of processes to run and press “Submit”
  - DDT itself involves the scheduler using the test queue
  - The debugging session starts almost immediately, but has a 1 hour time limit

DDT: A snapshot



Support

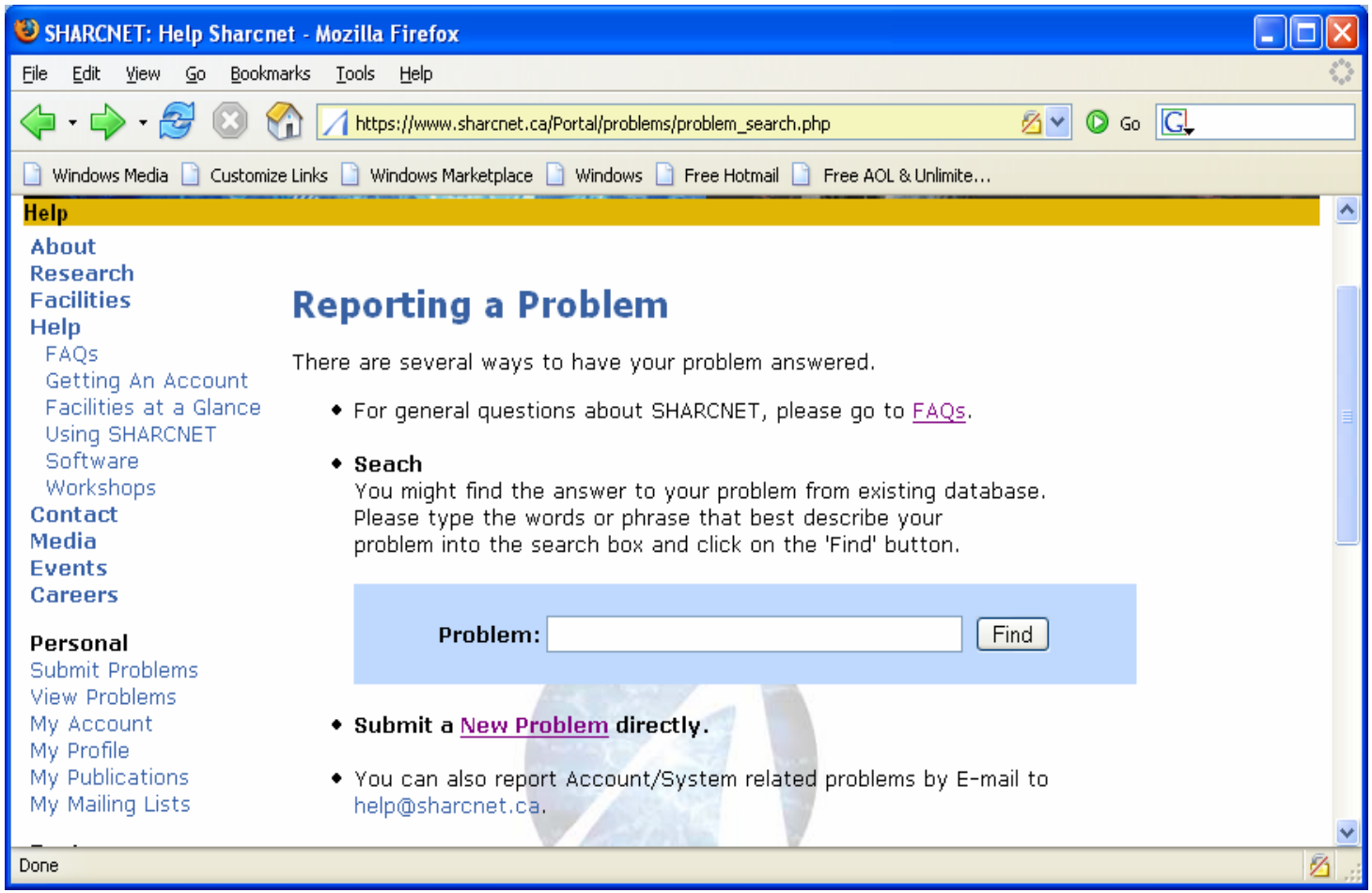
- People
- Problem tracking
- Research projects
- Education and training

SHARCNET People

- **HPC Analysts**
  - A point of contact, central resource.
  - Analysis of requirements.
  - Development support, performance analysis.
  - Training and education.
  - Research computing consultations.
- **System Administrators**
  - User accounts.
  - System software.
  - Hardware and software maintenance.
  - Research computing consultations.

Webportal: Problem Tracking

- Use **Problem Tracking** in the web portal.



Research Project Consultation

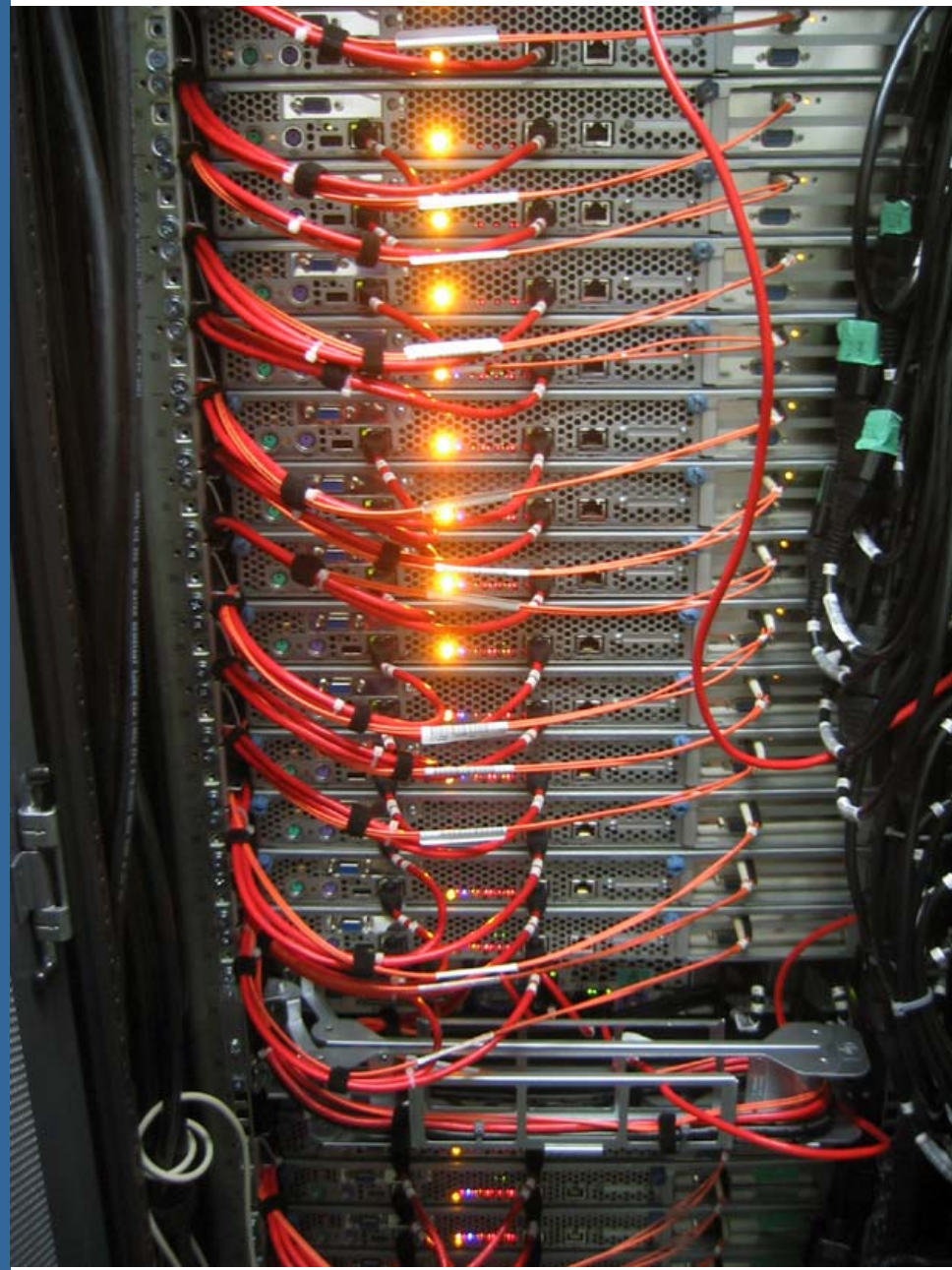
- HPTC people also do research consultation projects.

Education and Training

- SHARCNET offers different forms of education and training
  - Weekly online seminar: New users introduction and research topics;
  - Irregular and annual workshops. This year SHARCNET will hold a week long HPTC summer school;
  - Credit courses at undergraduate and graduate level.

MPI Examples

- Hello world! – *All up*
- Send/receive
- Broadcast – A collective all



Example 1: Hello world! – *all up*

```c
#include <stdio.h>
#include "mpi.h"

int main( int argc, char *argv[] )
{
   int nprocs, myrank, pnamelen;
   char pname[128];

   MPI_Init(&argc, &argv);
   MPI_Comm_size(MPI_COMM_WORLD, &nprocs);
   MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
   MPI_Get_processor_name(pname, &pnamelen);

   if (0 == myrank)
      printf("Hello world from root on %s\n", pname);
   else
      printf("Hello world from %d on %s of %d\n", \
         myrank, pname, nprocs);

   MPI_Finalize();
   return 0;
}
```

```fortran
program main
   implicit none
   include 'mpif.h'
   integer :: nprocs, myrank, ierr

   call MPI_Init(ierr)
   call MPI_Comm_size(MPI_COMM_WORLD, nprocs, ierr)
   call MPI_Comm_rank(MPI_COMM_WORLD, myrank, ierr)
   call MPI_Get_processor_name(pname, &pnamelen)

   if (0 == myrank) then
     print *, "Hello world from root on", pname
   else
     print *, "Proc", myrank, ": Hello world"
   end if

   call MPI_Finalize(ierr)
end program main
```

Example 2: P2P Communication – *send/receive*

- Main

```
#include “mpi.h”
int main( int argc, char *argv[] )
{
   MPI_Init(&argc, &argv);
   MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
   MPI_Comm_size(MPI_COMM_WORLD, &np);

   if (0 != myrank)
      server();
   else
   {
      sleep(1);              /* Wait for servers to come up */
      client();
   }

   MPI_Finalize();
   return 0;
}
```

Example 2: Echo…

- Server

```
static void server( void )
{
   char sbuf[MSGSIZE+1], rbuf[MSGSIZE+1];
   MPI_Status status;

   while (1)
   {
      MPI_Recv(rbuf, MSGSIZE, MPI_CHAR, 0, tag, MPI_COMM_WORLD, &status);
      if (strcasecmp("shutdown", rbuf) == 0) break;

      printf("Server %d: Received %d bytes (\"%s\")\n", myrank, MSGSIZE, rbuf);
      MPI_Send(rbuf, MSGSIZE, MPI_CHAR, 0, tag, MPI_COMM_WORLD);
   }

   return;
}
```

Example 2: Echo…

- Client

```
static void client( void )
{
   … …
   while (1)
   {
      printf("client>> ");
      fgets(sbuf, 256, stdin);

      for (ip = 1; ip < np; ip++)
         MPI_Send(sbuf, MSGSIZE, MPI_CHAR, ip, tag, MPI_COMM_WORLD);
      if (strcasecmp(sbuf, "shutdown") == 0) break;
      for (ip = 1; ip < np; ip++) {
         MPI_Recv(rbuf, MSGSIZE, MPI_CHAR, ip, tag, MPI_COMM_WORLD, &status);
         printf("Server %d echo: %s\n", ip, rbuf);
      }
   }
}
```

Example 3: Collective Communication – *reduce*

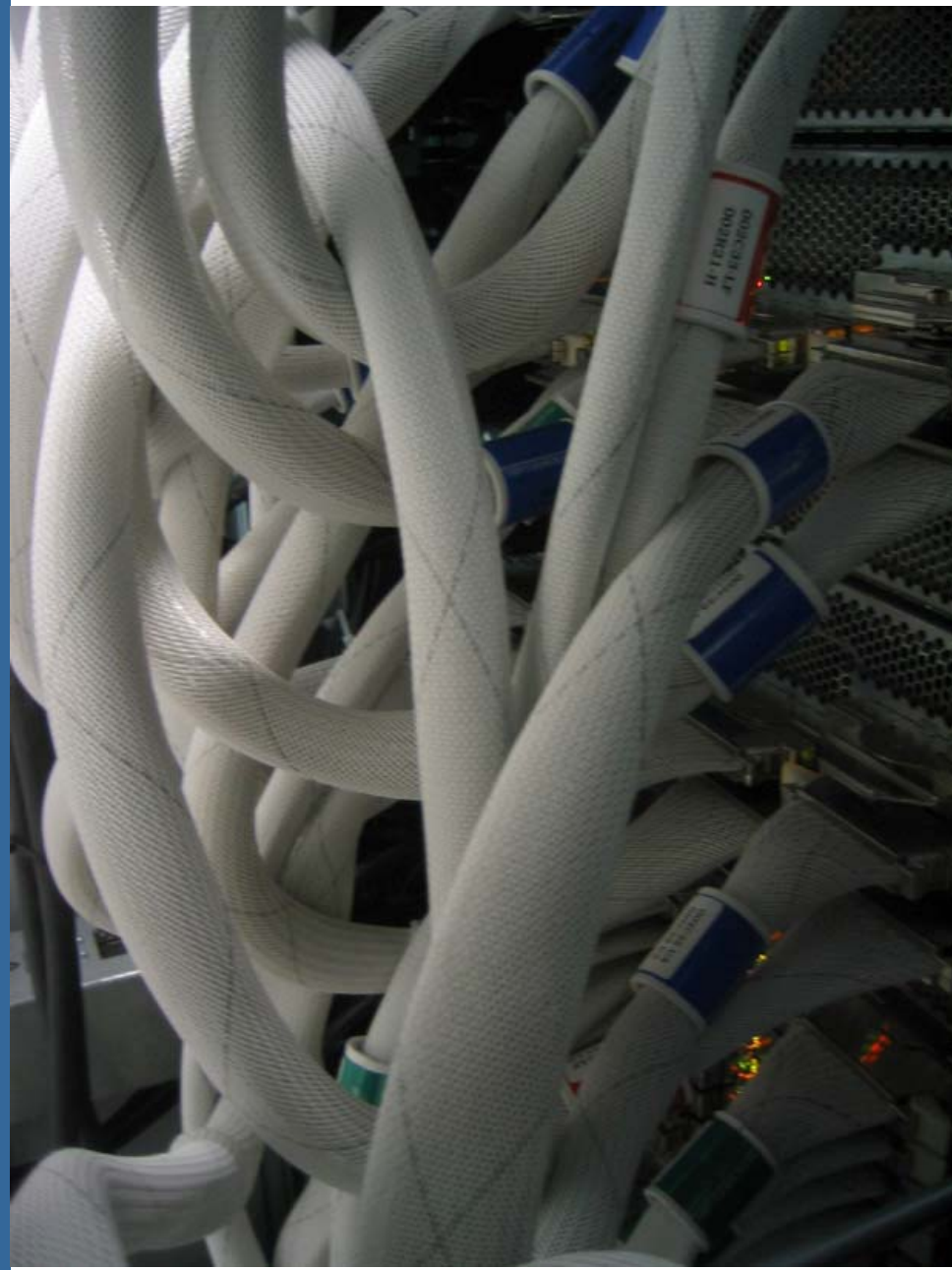
```
program mpi_avg
  … …
  iseed = myid + MPI_WTIME(ierr)
  call srand(iseed)
  nlocal = 1 + n * rand(0)
  do i = 1, nlocal
    mark = 100 * rand(0)
    print *, "(Proc ", myid, ")", mark
    myavg = (myavg * (i - 1) + mark) / i
  end do
  print *, "Average of class ", myid+1, ": ", myavg
  myavg = myavg * nlocal

  ! Reduce the local count
  call MPI_REDUCE(myavg, avg, 1, MPI_REAL8,&
    MPI_SUM, 0, MPI_COMM_WORLD, ierr)
  call MPI_REDUCE(nlocal, ntotal, 1, MPI_INTEGER,&
    MPI_SUM, 0, MPI_COMM_WORLD, ierr)

  ! Announce the final results (master only)
  if (myid .eq. 0) then
    avg = avg / ntotal
    print *, "Total number: ", ntotal
    print *, "Total average: ", avg
  end if

  call MPI_FINALIZE(ierr)
end program mpi_avg
```

References

- MPI forum, www.mpi-forum.org
- William Gropp, et al, ***Using MPI***, MIT, 1999.
- William Gropp, et al, ***Using MPI-2***, MIT, 1999.

OpenMP Examples

- Hello world! – *All up*
- Fork/join
- Concurrent tasks



Example 1: Hello world! (C)

```
#include <stdio.h>
#include <string.h>
#include "omp.h"

int main(int argc, char *argv[ ])
{
   int thd, nth = 1;

   if (argc > 1) nth = atoi(argv[1]);
   omp_set_num_threads(nth);

   #pragma omp parallel
   {
      tid = omp_get_thread_num();
      printf("Hello world! thread %d of %d\n", tid+1, nth);
   }

   return 0;
}
```

Example 1: Hello world! (Fortran)

```
      program hello
      implicit none
      integer tid, nth/1/
      integer omp_get_num_threads
      integer omp_get_thread_num
c
      read *, nth
c
      call omp_set_num_threads(nth)
c
!$omp parallel
      tid = omp_get_thread_num()
      print *, "Hello world! from thread ", thid+1, " of ", nth
!$omp end parallel
c
      stop
      end
```

Example 2: Loop parallelization

```c
#include "omp.h"

int main( int argc, char *argv[] )
{
   ... ...
   omp_set_num_threads(nt);

   #pragma omp parallel for reduction(+:s)
   for (i = 0; i < n; i++)
   {
      s += a[i];
#ifdef DEBUG
      tid = omp_get_thread_num();
      printf("Sum: %g (thread %d)\n", s, tid);
#endif
   }

   printf("Sum = %g\n", s);
   return 0;
}
```

```
      program reduce77
          ... ...
          call omp_set_num_threads(nt)
      c
      !$omp parallel do reduction(+:s)
          do 20 i = 1, n
            s = s + a(i)
      #ifdef DEBUG
            tid = omp_get_num_threads();
            print *, "Sum: ", s, " (thread ", tid, ")“
      #endif
        20 continue
      !$omp end parallel do
      c
          print *, "Sum = ", s
      c
          stop
          end
```

Example 3: Concurrency – Parallel sections

- Independent sections of code are executed concurrently

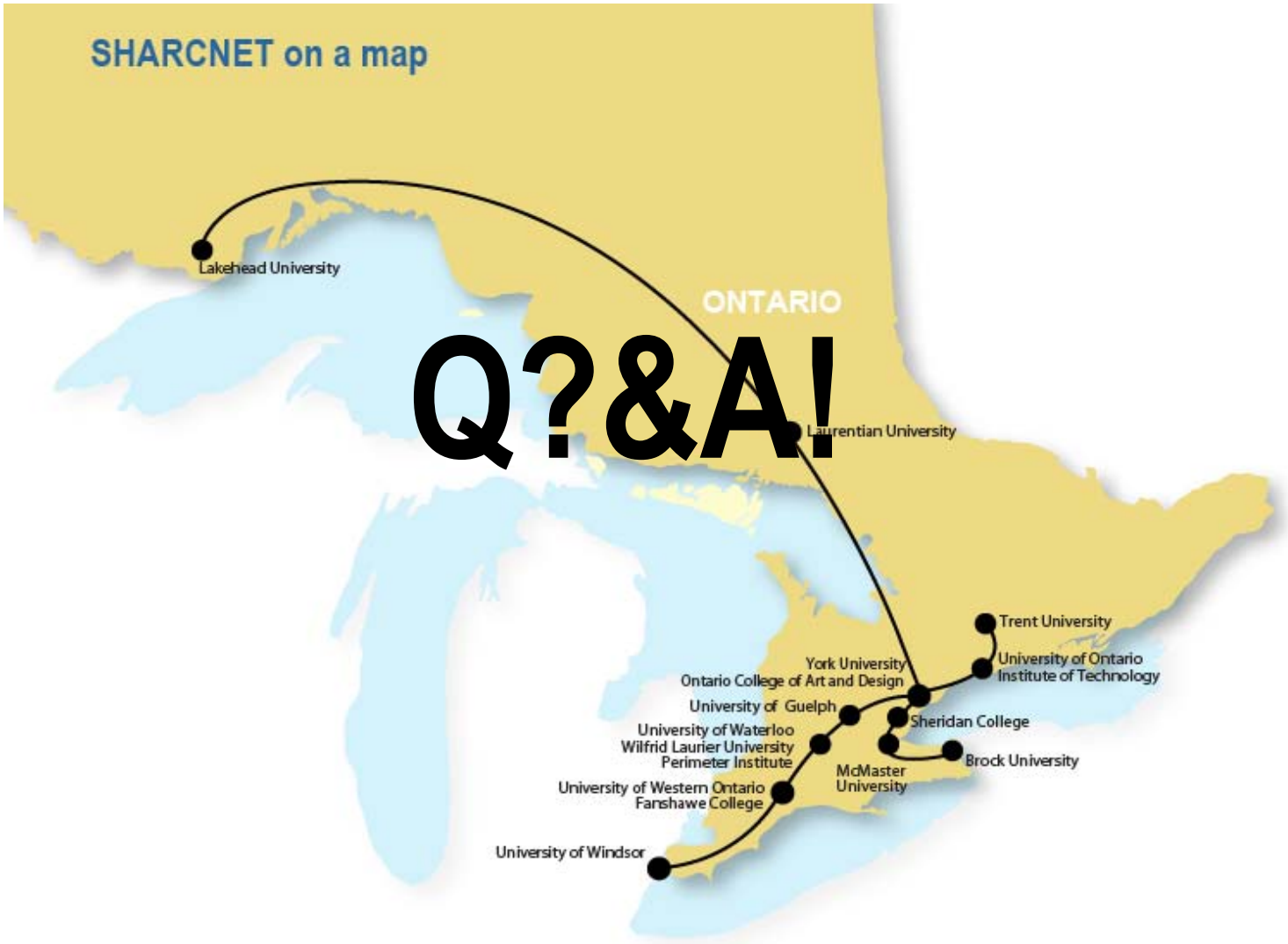
```
#pragma omp parallel sections
{
   #pragma omp section
   task1( );
   #pragma omp section
   task2( );
   #pragma omp section
   task3( );
}
```

omp barrier – no one shall proceed until all arrive at this point

- By default there is a **barrier** at the end of **omp** sections, use **nowait** clause to turn off the barrier

Reference

- OpenMP standard, www.openmp.org


SHARCNET on a map
Lakehead University
ONTARIO
Laurentian University
Trent University
University of Ontario
Institute of Technology
York University
Ontario College of Art and Design
University of Guelph
Sheridan College
University of Waterloo
Wilfrid Laurier University
Perimeter Institute
Brock University
McMaster
University
University of Western Ontario
Fanshawe College
University of Windsor

Q?&A!