# Using the Machines at the PSC

**Sections**

## 1.) **Overview of Systems at the PSC**

The PSC has three main systems that you can use to run your projects on: The Intel Cluster, the Cray T3E (jaromir), and the TCSini (Lemieux terascale system).

Intel Cluster: This cluster has twenty compute nodes each with 4 processors that share 1 Gbyte of memory and a 512 Kbyte cache. Ten of the nodes have 400 Mhz Pentium II Xeon Processors, and ten have 550 Mhz Pentium III Xeon processors. All the nodes run Linux. They are connected by a 100 Mbps switch. Intranode communication use shared memory; internode communication uses the switch (i.e. message passing). MPI will handle both intranode and internode communication. This cluster is accessed through a 2 processor front end where all compilation is performed.

Cray T3E: The T3E has three types of processing elements (PEs): 8 OS PEs accessible by the system only. 23 command PEs accessible by users for editing and compiling. These nodes can also be accessed through several default batch queues. 512 application PEs consisting of Digital Alpha 64-bit microprocessors running at 450 Mhz. These can be accessed both interactively and in batch mode.

TCSini: Lemieux comprises 750 Compaq Alphaserver ES45 nodes and two separate front end nodes. Each computational node contains four 1-GHz processors and runs the Tru64 Unix operating system. A Quadrics interconnection network connects the nodes. Each node is a 4 processor SMP, with 4 Gbytes of memory.

## 2.) **Connecting**

Reminder: Changing your password on one machine does not change it on the others. Keep your user account information form with your default password until you have changed it on all machines.

Intel Cluster: ssh intel2.psc.edu –X –l *username*

Cray T3E: ssh jaromir.psc.edu –X –l *username*

TCSini: ssh lemieux.psc.edu –X –l *username*

Use *far* or *scp* to transfer files to or from any of them machines (they will not accept a connection via ftp).
For downloading:
```
        scp username@remote_machine_name:filename local_filename
```
For uploading:
```
        scp local_filename username@remote_machine_name:filename
```

3.) **Compiling**

Programs on these machines can be written using the MPI message passing library. MPICH is a free implementation of MPI and is also included on the Intel Cluster and may create code with faster execution times. Compiling in with MPICH is very similar to compiling with MPI.

Intel Cluster: This machine has several compilers available for C, C++, Fortran77, Fortran90, and Java. The C and C++ compilers are the GNU compilers and the Portland compilers. It is recommended that you use MPICH on this machine as it is optimized for its architecture. The compiler scripts are all called mpicc or mpiCC and are located in:

/usr/local/packages/*compilerName*/bin/

The header file to be used in the include directive is located in:

/usr/local/packages/*compilerName*/include/

where *compilerName* is either *mpi* or *mpich*. To compile with mpi or mpich use a command line similar to:

/usr/local/packages/*compilerName*/bin/mpicc –o progName progName.c

for C or

/usr/local/packages/compilerName/bin/mpiCC –o progName progName.cpp

for C++.

Cray T3E: A sample makefile is located at:

http://www.psc.edu/machines/cray/t3e/t3eprogenvmake.html

Edit a copy of this makefile for your purposes and use it to compile.

TCSini: Use the include directive:

*#include <mpi.h>*

For C compile with:

cc program.c –lmpi –lelan

For C++ compile with either the Compaq compiler:

cxx program.cpp –lmpi –lelan

or the GNU compiler:

c++ program.cpp –lmpi –lelan

4.) **Submitting Jobs**

Intel Cluster: Jobs are not to be run on the front end nodes. Please submit them to the parallel queue using the *qsub* command and a script. All jobs must use a minimum of 2 nodes and are limited to 450 hours of CPU time across all processors used in the job. Also, jobs do not share nodes; we are charged by node not by processor.

Create a .rhosts file in your home directory containing:

```
   intel2      yourusername
   cog20       yourusername
   cog19       yourusername
   cog18       yourusername
   cog17       yourusername
   cog16       yourusername
   cog15       yourusername
   cog14       yourusername
   cog13       yourusername
   cog12       yourusername
   cog11       yourusername
   cog10       yourusername
   cog1        yourusername
   cog2        yourusername
   cog3        yourusername
   cog4        yourusername
   cog5        yourusername
   cog6        yourusername
   cog7        yourusername
   cog8        yourusername
   cog9        yourusername
```

Also create a PBSnodefile with the only the first column (no username) in your home directory (you may fine that the .rhosts file is unnecessary). Give both of the files a file protection of 400:

```
      chmod 400 .rhosts PBSnodefile
```

Copy their template script called '/usr/skel/pbs_PGexample.sh' to your home directory and set the file protection to 755. Edit a copy of this for your jobs as follows:

Uncomment the line that defines BINARY and set it equal to the full path name of your compiled executable (no quotes).
Add the lines:

```
 P4_SOCKBUFSIZE=0x40000
 P4_GLOBMEMSIZE=33554432
```

Change the definition of the PBS_NODEFILE in the line where it checks to see if it equals "NONE" to be the one you created in your home directory.

Edit the line:

```
#PBS –l nodes=2:PIII:ppn=4
```

to indicate the number of nodes you want that the number of processors per node. Also, PIII indicates you want to use the 550 Mhz processors, and PII indicates you want to use the 400 Mhz processors.  These #PBS directives set the command line parameters for the *qsub* command.  See the *qsub* manpage for more information.

Submit the job using your script to PBS with the following command:

```
qsub my_PGexample.sh
```

where my_PGexample.sh is the name of your script.

Cray T3E:  To run a program on the T3E use the command:

```
Mpprun –nX progName
```

where X is the number of PEs you want to use.

To submit a job for batch access use the qsub command:

```
qsub jobfile
```

Batch jobs can also be submitted via ftp.  See the following link for details:

http://www.psc.edu/machines/cray/t3e/access/ftp.html

TCSini:  Use the qsub command to submit batch jobs:

```
qsub jobScript.sh
```

An example job script is:

```
#!/bin/csh
#PBS -l walltime=5:00:00
#PBS -l rmsnodes=4:16
#PBS -j oe

set echo

# execute program
prun -N ${RMS_NODES} -n ${RMS_PROCS} ./a.out
```

The first #PBS directive sets the wall clock time limit (for 5 hours).  The second directive requests 16 processors on 4 contiguous nodes (remember each node has four processors). By default prun allocates in block mode (so processes 0,1,2, and 3 in the example above would all share the same node).  This can be changed with the –m cyclic option to cause a cyclic ordering of processes on nodes.  Jobs do not share nodes.   The third directive combines the stdout and stderror into one file.

5.) **Useful References**

If you have any questions or want more details please use the following links:

**http://www.psc.edu/machines/intel/intel.html**
**http://www.psc.edu/machines/cray/t3e/t3e.html**
**http://www.psc.edu/machines/tcs/**

If you have any questions about any of the commands given please read the man pages for that command on the machine on which you wish to use it. Please direct any further questions towards course personnel.