Parallel Computing
Machines and Accounts

Jesper Larsson Träff, Angelos Papatriantafyllou
{traff,papatriantafyllou}@par.tuwien.ac.at
Vienna University of Technology
Parallel Computing Group

Sprechstunde: By email appointment
TU Wien parallel computing shared-memory node

4xAMD “magny cours” 12-core Opteron 6168 processors, at 1.9GHz, total number of cores 48, 128GByte main memory

NUMA-type shared-memory system

Name: saturn.par.tuwien.ac.at
NUMA-type shared-memory system

- Per core L1 cache: 128KB
- Per core L2 cache 512KB
- Shared L3 cache 12288KB
Intended use:

1. Develop&debug in lab/own PC
2. Transfer to saturn (scp, ssh)
3. Run, test, debug, run, benchmark, ...
4. Transfer results back, write-up at home, submit...

Available software: emacs, vi, gcc, gdb – standard Linux tools

Pthreads:
gcc -Wall -o myprogpt -O3 -pthread myprogpt.c

OpenMP:
gcc -Wall -fopenmp -o myprogomp -O3 myprogomp.c

Cilk:
/opt/cilk-5.4.6/bin/cilkc -o mycilk -O3 mycilk.cilkc
BE CAREFUL AND CONSIDERATE:

• Do not allocate too much memory(!!) - node may crash/hang
• Make sure processes are killed before exiting, no zombies

• Don't swamp filesystem with excessive output files

• So far no queing system: be respectful to others (use „top“), don't block system for too long with intensive calculations, try to coordinate.

• In January: will install some form of reservation system (perhaps just TUWEL)
TU Wien parallel computing distributed-memory system

36x2xAMD "magny cours" 8-core Opteron 6134 processors at 2.3GHz, total number of cores 576, 32GByte main memory per node, total memory 1152GBytes.

Distributed memory system

Name: jupiter.par.tuwien.ac.at
But reality is (much) more complex

• Similar NUMA-type shared memory nodes as saturn. But higher clocked (2.3GHz vs. 1.9GHz)
• InfiniBand communication network

• 36 “nodes”, each with 16 cores - use no more than 16 MPI processes per node

Distributed memory system

Infiniband QDR switch MT4036
Intended use:

1. Develop & debug on own PC/lab
2. Transfer to jupiter (scp, ssh), jupiter0 is the login node
3. Run & benchmark
4. Transfer results back, write report

Currently installed: NEC mpi, OpenMPI, mvapich

bash
PATH=$PATH:/opt/NEC mpi/gcc/inst/bin64

tcsh
setenv PATH $PATH:/opt/NEC mpi/gcc/inst/bin64
Compile with
mpicc -o mpiprog -O3 <other gcc options> mpiprog.c
mpicc -help

Execute with
mpirun -np <number of procs on node> mpiprog
mpirun -host jupiter1 -np 5 -host jupiter2 -np 5 ...
mpiprog
mpirun -help

Nodes are named “jupiter0”, “jupiter1”, … “jupiter35”
Helpful (tcsh): define explicitly list of nodes, these nodes can be referenced with their index in list 0, 1, 2, ...

```bash
setenv MPILX_NODELIST 'jupiter0,jupiter1,...'
mpirun -node 0-19 -nnp 4 mpiprog
```

bash

```bash
MPILX_NODELIST=‘jupiter0,jupiter1,jupiter2,...’
export MPILX_NODELIST
```
BE CAREFUL AND CONSIDERATE:

- Start on single node, (perhaps other than jupiter0), debug with up to 16 MPI processes
- Gradually extend number of nodes
- After use: look for dead processes and kill them

MPI demon sometimes leaves undead processes. Please kill!

```bash
kill -9
on local node

ckill <processname>
kills on all notes; see

man ckill
```
• Don't swamp the file system (remove old log files, do not create excessive result files)

• So far no queueing system - simultaneous use will affect (timing) results

• In January: will install some form of reservation system (perhaps just TUWEL)
mpich, OpenMPI libraries/compilers also installed (not recommended, but perhaps interesting for comparison):

See in

/opt/mpich2
/opt/openmpi
In case of major problems...

Think about what you are doing! Don’t swamp the systems

If/when it should happen that:
• Nodes down/unavailable
• MPI/OpenMP/Cilk compilers unavailable
• File system full (by somebody elses doing...)
• ...

... then email us:

papatriantafyllou@par.tuwien.ac.at
levonyak@par.tuwien.ac.at (system administrator)
traff@par.tuwien.ac.at
Getting access, getting account

Environment (ssh):

**GNU/Linux, Mac OS X**
- Make sure the OpenSSH client is installed
- Open a terminal/shell

**Windows**
- Install Cygwin from [http://cygwin.com/](http://cygwin.com/)
- Select the OpenSSH client during installation
- Open a Cygwin terminal
Generate SSH key pair (public, private) using 4K Bits

Enter in a shell:
ssh-keygen -t rsa -b 4096

You will see the following output:
Generating public/private rsa key pair.
Enter file in which to save the key (~/.ssh/id_rsa): (just press <return>)
Enter passphrase (empty for no passphrase): (enter a secure passphrase - do not leave this empty!)
Enter same passphrase again:
Your identification has been saved in ~/.ssh/id_rsa.
Your public key has been saved in ~/.ssh/id_rsa.pub.
The key fingerprint is:

Note: Keys with less than 4K bits will not be accepted
Submit SSH public key (~/.ssh/rsa_id.pub) via TUWEL (there is an assignment for submitting the key). DEADLINE 9.11.2015
Submit SSH public key (~/.ssh/rsa_id.pub) via TUWEL (there is an assignment for submitting the key). DEADLINE 9.11.2015

You will be notified when the account has been created

Some rules:

• Do not submit the public key to us by email - only TUWEL

• NEVER share your private key with anyone

• Use a good passphrase to protect your private key. Do not leave empty
Using the systems: ssh, scp

To log in:
• ssh jupiter
• ssh saturn

To transfer files:
• scp file jupiter: (to jupiter)
• scp saturn:file . (from saturn)

This should work, provided the right SSH configuration (~/.ssh/config)
Host saturn
  HostName saturn.par.tuwien.ac.at
  Port 22
  User username
  IdentityFile ~/.ssh/id_rsa
  ProxyCommand ssh -p 1022 -q -W %h:%p \%
  TCPKeepAlive yes
  ServerAliveInterval 180
  ServerAliveCountMax 2

Host jupiter
  HostName jupiter.par.tuwien.ac.at
  Port 22
  User username
  IdentityFile ~/.ssh/id_rsa
  ProxyCommand ssh -p 1022 -q -W %h:%p \%
  TCPKeepAlive yes
  ServerAliveInterval 180
  ServerAliveCountMax 2
Security and integrity of our systems

No unauthorized accesses, no intruders, no hacking

Please inform us ([levonyak@par.tuwien.ac.at](mailto:levonyak@par.tuwien.ac.at), [traff@par.tuwien.ac.at](mailto:traff@par.tuwien.ac.at)) if you suspect that someone could have had access to your SSH private key, or if you detect any other security threat
Follow common sense:

• Choose a secure and unique passphrase for your SSH private key

• Never share your/any SSH private keys with others

• Always log out when you do not use the system

• Never let anyone else use your account

• Do not use our systems (sатурн, jupiter) for anything else than the exercises for this course