MPI Quick Reference: Compiling/Running

M. D. Jones, Ph.D.

Center for Computational Research
University at Buffalo
State University of New York

High Performance Computing I, 2009

Background

This document covers the essentials of compiling and running MPI applications on the CCR platforms. It does not cover MPI programming itself, nor debugging, etc.

Modules Software Management System

There are a large number of available software packages on the CCR systems, particularly the Linux clusters. To help maintain this often confusing environment, the **modules** package is used to add and remove these packages from your default environment (many of the packages conflict in terms of their names, libraries, etc., so the default is a minimally populated environment).

The module Command

module command syntax:

```
-bash-2.05b$ module help

  Modules Release 3.1.6 (Copyright GNU GPL v2 1991):
  Available Commands and Usage:
        + add|load              modulefile [modulefile ...]
        + rm|unload             modulefile [modulefile ...]
        + switch|swap           modulefile1 modulefile2
        + display|show          modulefile [modulefile ...]
        + avail                 [modulefile [modulefile ...]]
        + use [-a|--append]     dir [dir ...]
        + unuse                 dir [dir ...]
        + update
        + purge
        + list
        + clear
        + help                  [modulefile [modulefile ...]]
        + whatis                [modulefile [modulefile ...]]
        + apropos|keyword       string
        + initadd               modulefile [modulefile ...]
        + initprepend           modulefile [modulefile ...]
        + initrm                modulefile [modulefile ...]
        + initswitch            modulefile1 modulefile2
        + initlist
        + initclear
```

Using module in Batch

If you change shells in your batch script you may need to explicitly load the modules environment:

tcsh :

```
source $MODULESHOME/init/tcsh
```

bash :

```
. ${MODULESHOME}/init/bash
```

Objective: Construct a very elementary MPI program to do the usual "Hello World" problem, i.e. have each process print out its rank in the communicator.

in C

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

int main(int argc,char **argv)
{
    int myid, nprocs;
    int  namelen,mpiv,mpisubv;
    char processor_name[MPI_MAX_PROCESSOR_NAME];

    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&nprocs);
    MPI_Comm_rank(MPI_COMM_WORLD,&myid);
    MPI_Get_processor_name(processor_name,&namelen);

    printf("Process %d of %d on %s\n",myid, nprocs, processor_name);
    if (myid == 0) {
       MPI_Get_version(&mpiv,&mpisubv);
       printf("MPI Version: %d.%d\n",mpiv,mpisubv);
    }
    MPI_Finalize();
    return 0;
}
```

Running interactively on the Altix

Compiling (with gcc) and running interactively on lennon:

```
[jonesm@lennon ~]$ gcc -o ex1 ex1.c -lmpi
[jonesm@lennon ~]$ mpirun -np 4 ./ex1
Process 0 of 4 on lennon.ccr.buffalo.edu
MPI Version: 1.2
Process 1 of 4 on lennon.ccr.buffalo.edu
Process 3 of 4 on lennon.ccr.buffalo.edu
Process 2 of 4 on lennon.ccr.buffalo.edu
[jonesm@lennon ~]$
```

The Altix uses SGI's MPI implementation, and does not need compiler wrappers (i.e. no mpicc, mpicxx, mpif90, ...)

Running in Batch on the Altix

Very simple batch script:

```
[jonesm@lennon ~/d_debug]$ cat subQ
#PBS -S /bin/tcsh
#PBS -m e
#PBS -l nodes=1:ppn=4
#PBS -l walltime=00:10:00
#PBS -M jonesm@ccr.buffalo.edu
#PBS -o subQ.out
#PBS -j oe
#PBS -N subQ
#PBS
source $MODULESHOME/init/tcsh
set NN = `cat $PBS_NODEFILE | wc -l`
module list
cd $PBS_O_WORKDIR
mpirun -np $NN ./ex1
# remove scratch directory
\rm -r $PBSTMPDIR
```

and the output:

```
[jonesm@lennon ~/d_debug]$ cat subQ.out
############################PBS PROLOGUE BEGINS#######################
PBS prologue script run on host mccartney at Wed Sep 26 13:31:32 EDT 2007
############################PBS PROLOGUE ENDS#########################
Warning: no access to tty (Bad file descriptor).
Thus no job control in this shell.
PBSTMPDIR is /scratch/4585.lennon.ccr.buffalo.edu
PBSTMPDIR is /scratch/4585.lennon.ccr.buffalo.edu
Currently Loaded Modulefiles:
  1) null       2) modules    3) use.own
PBSTMPDIR is /scratch/4585.lennon.ccr.buffalo.edu
Process 0 of 4 on mccartney.ccr.buffalo.edu
MPI Version: 1.2
Process 1 of 4 on mccartney.ccr.buffalo.edu
Process 2 of 4 on mccartney.ccr.buffalo.edu
Process 3 of 4 on mccartney.ccr.buffalo.edu
PBS epilogue script run on host mccartney at Wed Sep 26 13:31:33 EDT 2007
###########PBS EPILOGUE BEGINS#######################
PBS epilogue script run on host mccartney at
Your scratch directory
        /scratch/4585.lennon.ccr.buffalo.edu
is marked for removal after a set period of time (several days) ...
Please cleanup your scratch files!
Clean up processes
############################PBS EPILOGUE->ENDS########################
```

Summary of MPI Modules on U2: MPICH

The variety of MPICH modules on U2 can get a bit confusing (type "module avail mpich" to see them all). They break down something like this:

mpich/*compiler*/*network*/*version*

where:

compiler is usually one of **gnu**, **intel-X** (X is the version of the compiler), or **pgi-X**. The compiler wrappers, **mpif77, mpif90, mpicc, mpicxx** then wrap around these base compilers (so you load the module for your desired compiler, then compile with one of **mpif77**/**mpif90**/**mpicc**/**mpicxx**).

network is one of **ch_p4** (gigabit ethernet) or **ch_mx** (Myrinet). For Myrinet batch jobs, make sure that also request the **GM** nodes property (e.g. **-lnodes=2:GM:ppn=2**).

version can be used to further distinguish a particular version of `MPICH`. If in doubt, omit it, and it should give you a reasonable default.

Running in Batch on U2 ...

Compilation (using gcc over ethernet) and batch run (using interactive mode) on U2:

```
[jonesm@bono ~]$ module load mpich/gcc-3.4.6/ch_p4
[jonesm@bono ~]$ mpicc -o ex1 ex1.c
[jonesm@bono ~]$ qsub -I -lnodes=2:ppn=2,walltime=00:10:00 -q debug
qsub: waiting for job 547181.bono.ccr.buffalo.edu to start
qsub: job 547181.bono.ccr.buffalo.edu ready

##############################PBS Prologue##################################
PBS prologue script run on host c15n32 at Wed Aug 22 12:52:23 EDT 2007
PBSTMPDIR is /scratch/547181.bono.ccr.buffalo.edu
```

```
[c15n32:~]$ module load mpich/gcc-3.4.6/ch_p4
[c15n32:~]$ cd $PBS_O_WORKDIR
[c15n32:~/d_debug]$ mpiexec ./ex1
Process 3 of 4 on c15n31.ccr.buffalo.edu
Process 0 of 4 on c15n32.ccr.buffalo.edu
MPI Version: 1.2
Process 2 of 4 on c15n31.ccr.buffalo.edu
Process 1 of 4 on c15n32.ccr.buffalo.edu
```

Using same example code, now compile for myrinet using the intel compiler, and run using a batch script:

```
[bono:~/d_debug]$ module avail mpich

------------------------------------------------------ /util/Modules/3.1.6/modulefiles ------------------------------------------------------
mpich/gcc-3.4.6/ch_mx/1.2.7..4   mpich/intel-9/ch_p4/1.2.7p1       mpich/pgi-7.0/ch_p4/1.2.7p1
mpich/gcc-3.4.6/ch_p4/1.2.7p1    mpich/intel-9/ch_p4/1.2.7p1-fix   mpich/pgi-7.1/ch_mx/1.2.7..4
mpich/intel-10.0/ch_mx/1.2.7..4  mpich/intel-9.1/ch_mx/1.2.7..4    mpich/pgi-7.1/ch_p4/1.2.7p1
mpich/intel-10.0/ch_p4/1.2.7p1   mpich/intel-9.1/ch_p4/1.2.7p1     mpich2/gcc-3.4.6/1.0.7
mpich/intel-10.1/ch_mx/1.2.7..4  mpich/pgi-6.2/ch_mx/1.2.7..4      mpich2/gcc-3.4.6/1.0.7-pvfs2
mpich/intel-10.1/ch_mx/1.2.7..7  mpich/pgi-6.2/ch_p4/1.2.7p1
mpich/intel-10.1/ch_p4/1.2.7p1   mpich/pgi-7.0/ch_mx/1.2.7..4

[bono:~/d_debug]$ module load mpich/intel-9.1/ch_mx
[bono:~/d_debug]$ mpicc -o ex1 ex1.c
```

```
[bono:~/d_debug]$ cat subQ
#PBS -S /bin/bash
#PBS -l walltime=00:10:00
#PBS -l nodes=2:GM:ppn=2
#PBS -M jonesm@ccr.buffalo.edu
#PBS -m e
#PBS -N test
#PBS -o subQ.out
#PBS -j oe
#
. ${MODULESHOME}/init/bash
module load mpich/intel-9.1/ch_mx
module list
#
# cd to directory from which job was submitted
#
cd $PBS_O_WORKDIR
which mpiexec
mpiexec ./ex1
```

```
[bono:~/d_debug]$ cat subQ.out
################################PBS Prologue###################################
PBS prologue script run on host c15n32 at Wed Aug 22 12:46:18 EDT 2007
PBSTMPDIR is /scratch/547179.bono.ccr.buffalo.edu
 'openmpi/intel-9/1.2.3' load complete.
Currently Loaded Modulefiles:
  1) null                                3) use.own
  2) modules                             4) mpich/intel-9.1/ch_mx/1.2.7..4
/util/mpich/1.2.7p1/intel-9.1/ch_mx/bin/mpiexec
Process 1 of 4 on c15n32.ccr.buffalo.edu
Process 3 of 4 on c15n31.ccr.buffalo.edu
Process 0 of 4 on c15n32.ccr.buffalo.edu
MPI Version: 1.2
Process 2 of 4 on c15n31.ccr.buffalo.edu
################################PBS EPILOGUE###################################
PBS epilogue script run on host c15n32 at Wed Aug 22 12:46:20 EDT 2007
    for 4 processes.
Cleaning up processes on node c15n32 for user jonesm
Removing /scratch/547179.bono.ccr.buffalo.edu on node c15n32
Cleaning up processes on node c15n31 for user jonesm
Removing /scratch/547179.bono.ccr.buffalo.edu on node c15n31
################################PBS EPILOGUE###################################
```

OpenMPI on U2

OpenMPI is an MPI implementation that fully supports MPI-1 and MPI-2 (note that MPICH has support for MPI-1.2 plus some bits from MPI-2, like I/O and the `C++` bindings). It is freely available:

`www.open-mpi.org`

and has some nice features, including run-time support for different networks (i.e. one executable that you can run over Ethernet or Myrinet).

OpenMPI Modules on U2

```
[bono:~]$ module avail openmpi

------------------ /util/Modules/3.1.6/modulefiles -------------------------------------
openmpi/gcc-3.4.6/1.2.5          openmpi/intel-10.1/1.2.5        openmpi/pgi-7.1/1.2.5
openmpi/gcc-3.4.6/1.2.6          openmpi/intel-10.1/1.2.6        openmpi/pgi-7.1/1.2.6
openmpi/gcc-3.4.6/1.2.6-pvfs2    openmpi/intel-9.1/1.2.5
openmpi/gcc-4.1.2/1.2.5          openmpi/pgi-6/1.2.3
```

Note that there are just different compilers and OpenMPI versions to worry about here.

Running an OpenMPI Code on Myrinet

First, build the code:

```
[bono:~/d_debug]$ module load openmpi
 'openmpi/gcc-3.4.6/1.2.6-pvfs2' load complete.
[bono:~/d_debug]$ mpicc -o ex1 ex1.c
[bono:~/d_debug]$ qsub -q debug -lnodes=2:GM:ppn=2,walltime=00:10:00 -I
qsub: waiting for job 1034243.bono.ccr.buffalo.edu to start
qsub: job 1034243.bono.ccr.buffalo.edu ready

################################PBS Prologue################################
PBS prologue script run on host c14n32 at Fri Sep 5 17:06:13 EDT 2008
PBSTMPDIR is /scratch/1034243.bono.ccr.buffalo.edu
[c14n32:~]$ cd $PBS_O_WORKDIR
[c14n32:~/d_debug]$ module load openmpi
 'openmpi/gcc-3.4.6/1.2.6-pvfs2' load complete.
[c14n32:~/d_debug]$ export NPROCS=`cat $PBS_NODEFILE | wc -l`
[c14n32:~/d_debug]$ mpiexec -np $NPROCS ./ex1
warning:regcache incompatible with malloc
warning:regcache incompatible with malloc
warning:regcache incompatible with malloc
warning:regcache incompatible with malloc
Process 0 of 4 on c14n32.ccr.buffalo.edu
MPI Version: 2.0
Process 1 of 4 on c14n32.ccr.buffalo.edu
Process 2 of 4 on c14n31.ccr.buffalo.edu
Process 3 of 4 on c14n31.ccr.buffalo.edu
```

U2: Running an OpenMPI Code on Ethernet

By default OpenMPI is smart enough to pick the faster (Myrinet) network, if it present on all the nodes in your job, so to force a run over Ethernet:

- Request U2 nodes that do not have Myrinet using the **CPU30** property:

```
[bono:~/d_debug]$ cat subQ
#PBS -S /bin/bash
#PBS -l walltime=00:10:00
#PBS -l nodes=2:CPU30:ppn=2
#PBS -M jonesm@ccr.buffalo.edu
#PBS -m e
#PBS -N test
#PBS -o subQ.out
#PBS -j oe
. ${MODULESHOME}/init/bash
module load openmpi/gnu
module list
# cd to directory from which job was submitted
cd $PBS_O_WORKDIR
which mpiexec
NPROCS=`cat $PBS_NODEFILE | wc -l `
mpiexec -np $NPROCS ./ex1
```

```
[bono:~/d_debug]$ cat subQ.out
################################PBS Prologue###################################
PBS prologue script run on host c27n29 at Wed Aug 22 13:39:34 EDT 2007
PBSTMPDIR is /scratch/547216.bono.ccr.buffalo.edu
 'openmpi/gnu/1.2.3' load complete.
Currently Loaded Modulefiles:
  1) null                  3) use.own
  2) modules               4) openmpi/gnu/1.2.3
/util/openmpi/1.2.3/gcc-3.4.6/bin/mpiexec
[c27n28:19692] mca: base: component_find: unable to open mtl mx: file not found (ignored)
[c27n29:28056] mca: base: component_find: unable to open mtl mx: file not found (ignored)
[c27n28:19691] mca: base: component_find: unable to open mtl mx: file not found (ignored)
[c27n29:28055] mca: base: component_find: unable to open mtl mx: file not found (ignored)
[c27n28:19691] mca: base: component_find: unable to open btl mx: file not found (ignored)
[c27n28:19692] mca: base: component_find: unable to open btl mx: file not found (ignored)
[c27n29:28055] mca: base: component_find: unable to open btl mx: file not found (ignored)
[c27n29:28056] mca: base: component_find: unable to open btl mx: file not found (ignored)
Process 1 of 4 on c27n29.ccr.buffalo.edu
Process 0 of 4 on c27n29.ccr.buffalo.edu
MPI Version: 2.0
Process 2 of 4 on c27n28.ccr.buffalo.edu
Process 3 of 4 on c27n28.ccr.buffalo.edu
################################PBS EPILOGUE###################################
PBS epilogue script run on host c27n29 at Wed Aug 22 13:39:37 EDT 2007
    for 4 processes.
Cleaning up processes on node c27n29 for user jonesm
Removing /scratch/547216.bono.ccr.buffalo.edu on node c27n29
Cleaning up processes on node c27n28 for user jonesm
Removing /scratch/547216.bono.ccr.buffalo.edu on node c27n28
################################PBS EPILOGUE###################################
```

U2: Intel MPI

There are several commercial implementations of MPI, `Intel` and `HP` currently being the most prominent (IBM, Sun, SGI, etc. all have their own variants, but usually are only supported on their own hardware). CCR has a license for `Intel` MPI, and it has some nice features:

- Support for multiple networks (Infiniband, Myrinet, TCP/IP)
- Part of the ScaLAPACK support in the Intel MKL
- MPI-2 features (one-sided, dynamic tasks, I/O with limited filesystem support)
- CPU pinning/process affinity

Build the code with the appropriate wrappers:

```
[bono:~/d_mpi-samples]$ ml intel-mpi
[bono:~/d_mpi-samples]$ mls
Currently Loaded Modulefiles:
  1) null                    3) use.own                 5) intel/11.0
  2) modules                 4) java/j2sdk/1.6.0_13     6) intel-mpi/3.2
[bono:~/d_mpi-samples]$ mpiicc -o hello hello.c
```

Unfortunately Intel MPI lacks any integration with PBS/Torque, and instead relies on "daemons" to initiate MPI tasks.

```
1  #PBS -S /bin/bash
2  #PBS -q debug
3  #PBS -l walltime=01:00:00
4  #PBS -l nodes=2:ppn=2
5  #PBS -M jonesm@ccr.buffalo.edu
6  #PBS -m e
7  #PBS -N test
8  #PBS -o subQ.out
9  #PBS -j oe
10 #
11 # Note the above directives can be commented out using an
12 # additional "#" (as in the debug queue line above)
13 #
14 # Initialize modules environment, load modules needed
15 #  by your code.
16 #
17 . ${MODULESHOME}/init/bash
18 module purge
19 module load modules
20 module load use.own
21 module load intel-mpi
22 #
23 # cd to directory from which job was submitted
24 #
25 cd $PBS_O_WORKDIR
```

```
26 #
27 # Intel MPI needs mpd 'daemons' launched (no integration with PBS,
28 #  so you have to tell it where to run).
29 # You can find description of all Intel MPI parameters in the
30 # Intel MPI Reference Manual.
31 # See <intel mpi installdir>/doc/Reference_manual.pdf
32 #
33 export I_MPI_DEBUG=100
34 NPROCS=`cat $PBS_NODEFILE | wc -l`
35 NODES=`cat $PBS_NODEFILE | uniq`
36 NNODES=`cat $PBS_NODEFILE | uniq | wc -l`
37 mpdboot -n $NNODES -f $PBS_NODEFILE -v
38 mpdtrace
39 mpiexec -np $NPROCS ./hello
40 mpdallexit
```

```
[bono:~/d_mpi-samples]$ cat subQ.out
c14n30 has 2 cores/processors.
###############################PBS Prologue###################################
PBS prologue script run on host c14n30 at Fri Sep  4 11:59:37 EDT 2009
Job 1533704.bono.ccr.buffalo.edu has requested 2 cores/processors per node.
Job 1533704.bono.ccr.buffalo.edu is running dedicated on c14n30.  Local scratch will ...
PBSTMPDIR is /scratch/1533704.bono.ccr.buffalo.edu
running mpdallexit on c14n30
LAUNCHED mpd on c14n30  via
RUNNING: mpd on c14n30
LAUNCHED mpd on c14n29  via  c14n30
RUNNING: mpd on c14n29
c14n30
c14n29
[1] I_MPI_init_dat_regestry_info(): [0] I_MPI_init_dat_regestry_info(): trying to load ...
[0] my_dlopen(): trying to dlopen: libdat.so
[0] MPI startup(): cannot open dynamic library libdat.so
.....
.....
[1] MPI startup(): DAPL provider <NULL on rank 0:c14n30 differs from <NULL string> on ...
[2] MPI startup(): DAPL provider <NULL on rank 0:c14n30 differs from <NULL string> on ...
[3] MPI startup(): DAPL provider <NULL on rank 0:c14n30 differs from <NULL string> on ...
[0] MPI startup(): shared memory and socket data transfer modes
[1] MPI startup(): shared memory and socket data transfer modes
[2] MPI startup(): shared memory and socket data transfer modes
[3] MPI startup(): shared memory and socket data transfer modes
[2] MPI Startup(): process is pinned to CPU00 on node c14n29.ccr.buffalo.edu
[0] MPI startup(): Intel(R) MPI Library, Version 3.2  Build 20080917
[0] MPI startup(): Copyright (C) 2003-2008 Intel Corporation.  All rights reserved.
```

```
30 [0] MPI Startup(): process is pinned to CPU00 on node c14n30.ccr.buffalo.edu
31 [3] MPI Startup(): process is pinned to CPU01 on node c14n29.ccr.buffalo.edu
32 Process 3 of 4 on c14n29
33 Process 2 of 4 on c14n29
34 [1] MPI Startup(): process is pinned to CPU01 on node c14n30.ccr.buffalo.edu
35 Process 1 of 4 on c14n30
36 [0] Rank    Pid      Pin cpu Node name
37 [0] 0       22950    0       c14n30.ccr.buffalo.edu
38 [0] 1       22949    1       c14n30.ccr.buffalo.edu
39 [0] 2       17647    0       c14n29.ccr.buffalo.edu
40 [0] 3       17648    1       c14n29.ccr.buffalo.edu
41 [0] Init(): I_MPI_DEBUG=100
42 [0] Init(): LD_LIBRARY_PATH=/util/java/jdk1.6.0_13/lib:/util/intel/Compiler/11.0...
43 [0] Init(): MPICH_INTERFACE_HOSTNAME=10.33.14.30
44 Process 0 of 4 on c14n30
45 MPI Version: 2.0
46 ################################PBS EPILOGUE#################################
47 PBS epilogue script run on host c14n30 at Fri Sep 4 11:59:45 EDT 2009
48     for 4 processes.
49 Cleaning up processes on node c14n30 for user jonesm
50 Removing /scratch/1533704.bono.ccr.buffalo.edu on node c14n30
51 Cleaning up processes on node c14n29 for user jonesm
52 Removing /scratch/1533704.bono.ccr.buffalo.edu on node c14n29
53 ################################PBS EPILOGUE#################################
```

Summary - MPI at CCR

- Use `modules` environment manager to choose your MPI flavor
- Still recommend `MPICH/MPICH-MX` on the clusters, unless you need MPI-2 features (`OpenMPI` or `Intel MPI`)
- Be careful with task launching - use `mpiexec` whenever possible
- Ensure that your MPI processes end up where you want - use `ps` and `jobvis` to check (also use MPI_Get_processor_name in your code).