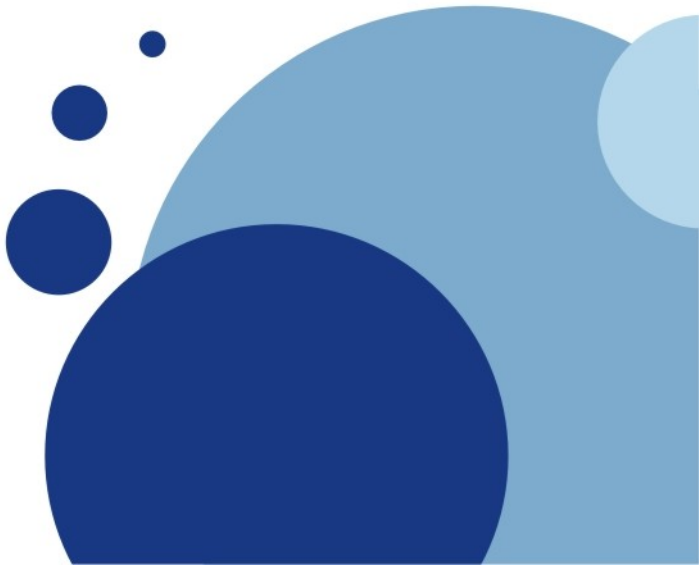
Advanced MPI

metu-ceng
ts@TayfunSen.com
28 November 2008



Outline

- Quick Introduction to MPI
- Collective Communication
- OpenMPI
- MPI using C++: Boost Libraries
- User Defined Types
- Tools of Trade & Debugging
- References
- Q & A

Quick Intro to MPI

- MPI is a ***standard*** with many implementations (libraries)
- ***OpenMPI*** which evolved from lam-mpi/mpich and ***MVAPICH*** are bigger ones
- Basically a message passing ***API***
- Some basic requirements: ***high performing, scalable*** and ***portable***.

Collective Comm.

- Point to point communication has been talked in previous seminars
- ***communicator***: groups processes so you can communicate with all at the same time (custom groups)
- ***synchronization***, ***data sharing***, ***reduce*** operations are all examples

Collective Comm.

- Demo time!

Example collective communication programs on localhost

Connecting to NAR – best guide is the web site – hpc.ceng.metu.edu.tr

Which MPI to use, how to set up environment?

Using man pages

OpenMPI

- Both ***OpenMPI*** (Beware: **not to be confused with OpenMP**) and ***MVAPICH*** are good implementations, installed on NAR – choose according to your taste
- No proper documentation at the moment, use the ***FAQ page***
- Can also do ***MPMD***:

```
$ mpirun -np 2 a.out : -np 2 b.out
```

Ranks 0 to 3 on different progs.

OpenMPI

- OpenMPI is highly configurable
- MCA (modular component architecture) parameters for run-time tuning
- Check following commands for these parameters:

```
$ ompi_info --param all all
$ ompi_info --param btl tcp
```

Shows parameters and their explanations. Check as needed.

OpenMPI

- `$ mpirun --mca btl self,sm,gm ...`
- Above says use only loopback communication, shared memory or Myrinet/GM for this run
- While communicating on the same machine, shared memory is used automatically
- TCP is used by default if it is available, for node2node communication. Check http://www.open-mpi.org/faq/?category=tuning for more info

MPI with some Boost

- Boost C++ libraries complete STL
- Some are to be included in the standard

MPI with some Boost

- Pretty easy to set up, install
- Some libraries are header only
- No Boost.MPI library on NAR /usr/lib/libboo* and no headers /usr/include/boost/
- Interesting libraries are Boost.MPI and Boost.Serialization
- Set up on your home directory on NAR or ask the sys admins
- Can use any underlying MPI implementation

MPI with some Boost

- Previous code becomes:

```
mpi::environment env(argc, argv);
mpi::communicator world;
std::cout << "I am process " <<
world.rank() << " of " <<
world.size() << "." << std::endl;
```

- Some examples
- Need to write a Makefile to add required libraries/header files while compiling
- Python bindings also exist

User Defined Data Types

- Best thing about Boost is it is so easier to transfer complex types
- Uses boost::serialization
- Quick and easy
- An example
- Some thoughts: what happens when there are pointers? (like when transmitting trees?)

Tools of Trade

- gfilt for intelligible template errors
- Extremely useful when using C++ to code
- For debugging, one can use gdb and for memory checking, valgrind

Tools of Trade

BEFORE

```
rtmap.cpp: In function `int main()':
rtmap.cpp:19: invalid conversion from `int' to `
   std::_Rb_tree_node<std::pair<const int, double> >*'
rtmap.cpp:19:   initializing argument 1 of `std::_Rb_tree_iterator<_Val, _Ref,
   _Ptr>::_Rb_tree_iterator(std::_Rb_tree_node<_Val>*) [with _Val =
   std::pair<const int, double>, _Ref = std::pair<const int, double>&, _Ptr =
   std::pair<const int, double>*]'
rtmap.cpp:20: invalid conversion from `int' to `
   std::_Rb_tree_node<std::pair<const int, double> >*'
rtmap.cpp:20:   initializing argument 1 of `std::_Rb_tree_iterator<_Val, _Ref,
   _Ptr>::_Rb_tree_iterator(std::_Rb_tree_node<_Val>*) [with _Val =
   std::pair<const int, double>, _Ref = std::pair<const int, double>&, _Ptr =
   std::pair<const int, double>*]'
```

AFTER

```
 *** {BD Software Proxy c++ for gcc v3.01} STL Message Decryption is ON! ***
rtmap.cpp: In function `int main()':
rtmap.cpp:19: invalid conversion from `int' to `iter'
rtmap.cpp:19:   initializing argument 1 of `iter(iter)'
rtmap.cpp:20: invalid conversion from `int' to `iter'
rtmap.cpp:20:   initializing argument 1 of `iter(iter)'
stl_tree.h: In member function `void map<int,double>::insert_unique(_II, _II)':
   [STL Decryptor: Suppressed 1 more STL standard header message]
rtmap.cpp:21:   instantiated from here
stl_tree.h:1161: invalid type argument of `unary *'
```

Debugging

- Parallel debuggers? Totalview? DDT? **Expensive,** costs $$$
- Too simple to use a conventional debugger. GDB is your best friend.
- If using Boost libraries, C++ or more complex structures like trees: Use a graphical debugger DDD, xxgdb, ... choose your favourite.

Debugging

- Demo time! Using DDD to debug parallel programs.

Debugging

- What else? Valgrind for detecting memory leaks, and possible core dumps. Just use
- # mpirun -np 2 valgrind –leak-check=full ./main
- Beware! Extremely slow while running with valgrind

Some left out bits

- There are more advanced topics such as Parallel I/O, RDMA etc.
- Different hardware, network characteristics enable different methods
- Myrinet, OpenFabrics, Quadrics...
- Interconnections becoming much more complex

References & Notes

- The presentation template is from: http://www.presentationhelper.co.uk/free-open-
- Hardware information of NAR retrieved from http://hpc.ceng.metu.edu.tr/system/hardware/
- A great tutorial on MPI can be found @ NCSA http://webct.ncsa.uiuc.edu:8900/public/MPI/
- Boost homepage is at: http://www.boost.org/
- OpenMPI home page can be found: http://www.open-mpi.org/
- This presentation can be obtained from slideshare at: http://www.slideshare.net/tayfun/
- gfilt can be obtained from http://www.bdsoft.com/tools/stlfilt.html

The End

Thanks For Your Time.

Any Questions

?