Message passing and MPI
Assignments and problems
Graham Fagg
CS-594 lecture 2 Spring 2003

Rules
• Two weeks to complete assignments and problems. Hand in by midday 3rd Feb03.
• If you have problems and do not ask for help (until 10 minutes before the deadline) beware!
• Hand-in written work either on paper or Email to me fagg@cs.utk.edu
• Code is to be tarred with makefiles, output etc and a MD5 signature sent to me:
  – Md5 mywork.tar & mail fagg@cs.utk.edu
  – Code is to run on hydra or torc machines. If I cannot verify it by remaking it and running it then I will assume it does not!
  – Broken code with comments gets more points than non working non commented code.
  – A short description of design is always needed.

Part A
Correctness and buffering?
• Proc 0                    Proc 1
  • MPI_Send (data, size.. 1) MPI_Send (data, size.. 0..)
  • MPI_Recv (indata, insize..1..) MPI_Recv (indata, insize.. 0..)
Above is a head to head send. This might or might not work depending on
the system, MPI implementation or other factors such as timing.
1. Write a paragraph on why the above is an incorrect [non-
deterministic] MPI code.
2. Re-write the above code in 3 different ways to make it work (hint,
there are 4 simple ways)
Extra under MPICH at what data size does the above break?

Part B
3 ring codes
1. Write a simple code that sends integer numbers around a ring of 4
   nodes (from node 0->1->2->3->0).
2. Alter the above code so that every node ‘swaps’ their bottom most
   edge of a 2D [random] matrix with the top edge of the next node
   around a ring. After the code has run each node should have
different data in the top and bottom edges of their matrix.
   • Hint, it could be two rings in opposite directions or a real swap!
3. Do the same but sending right and left columns of data (i.e. in C non
   contiguous data). You should show both packed data and a derived
data type. (Which is faster and why?)
4. Question. In 2&3 we have ‘n’ random matrices. How do we keep
   them random (which is important for Monte Carlo simulations) on a
   parallel machine? Why would it not be random across all nodes?
   • Hint, look-up Knuths semi-numeric algorithms vol2 and leap-frog generators

Part C
Collecting with collectives
1. Write a SPMD code with 1 master and 3 slaves that
   calculates pi using code handed out in class.
   • Use p2p calls.
   • Master sends out slices
   • Slaves calculate integral slices
   • Master sums to get pi.
2. Change the above to use collective calls
   • Master sends out slices all at once
   • Master receives all slices at once
   • Master sums to get pi.
3. Change 2 so that the master does not have to sum the
   data explicitly but the collective for receiving the partial
   results does this.
Part D

Load balancing and Scalable supercomputers?

At the end of Part C you should have used two collectives. Are these scalable and do they allow load balancing for overloaded nodes?

- If non-blocking calls were used in C-1 describe in ½ a page how this could allow load balancing compared to the collectives solution in C-3.
  - Hint, look up bag of tasks computing.
- Describe in a page how you would implement a broadcast that works faster than \( O(N) \) for \( N \) nodes where \( N \) is very large.
  - Hint, what topologies would you use and why?