Table of Contents

1. HW3, P1: Point-to-Point Communications - Ping-Pong
2. HW3, P2: Collective Comms - Ping-Ping
3. HW3, P3: Point-to-Point Comms - Ring
4. HW3, P4: Collective Comms - Exchange
5. General Instructions
6. Reporting Results
HW3, P1: Point-to-Point Comms - Ping-Pong

Analyze and modify the *ping-pong.c* program written by Pacheco and provided on tuckoo (see /cs596/hw3).

- Text code *mpi-ping-pong.c*:
  - Uses point-to-point comm routines, MPI_Send and MPI_Recv.
  - Two PE’s pass a message packet back and forth to each other.
  - Packet Size is gradually increased until maximum bandwidth is reached.

- All tests should be run for the following conditions:
  - Number of PEs: 2
  - Packet Sizes \(= 10^n\), where \(n = [1, 2, \ldots, N_{\text{max}}]\).
  - Test for integer, float and double data types

- Report:
  - Elapsed time and statistics (see below under General Instructions).
  - Calculate Bandwidth (see below under General Instructions).
  - Table of data for each data type: Packet Size, BW, Timing, & Stats
  - Plot of the BW as a function of Packet Size for the 3 data types.
  - Determine \(N_{\text{max}}\)
Test code *mpi-ping-pong.c*:
- Modify *ping-pong.c* code to use collective MPI_SendRecv
- Two PE's pass a message packet back and forth to each other.
- Each PE gets a message from its' lower neighbor and sends the MSG to its' upper neighbor.

All tests should be run for the following conditions:
- Number of PEs: 2
- Packet Sizes = $10^n$, where $n = [1, 2, \ldots, N_{max}]$.
- Test for integer, float and double data types

Report:
- Elapsed time and statistics (see below under General Instructions).
- Calculate Bandwidth (see below under General Instructions).
- Table of data: Packet Size, BW, Timing, & Stats
- Plot of the BW as a function of Packet Size for the 3 data types.
- Determine $N_{max}$
- Compare to *ping-pong.c*
HW3, P3: Point-to-Point Comms - Ring

- Test code *mpi-ring.c*
- Uses the point-to-point comm routines, MPI_Send and MPI_Recv.
- $P_0$ creates the initial message data array.
- Each $P_0$ waits for data from its lower neighbor, and then passes that message on to its higher neighbor.
- All tests should be run for the following conditions:
  - Number of PEs: 2, 4, 8, 16
  - Packet Sizes = $10^n$, where $n = [1, 2, \ldots, N_{\text{max}}]$. What is $N_{\text{max}}$?
  - Test for double data types
- Report:
  - Elapsed time and statistics (see below under General Instructions).
  - Calculate Bandwidth (see below under General Instructions).
  - Table of data: Packet Size, BW, Timing, & Stats
  - Plot of the BW as a function of Packet Size and number of nodes
HW3, P4: Collective Comms - Exchange

- Test code `mpi-exchange.c`
  - Use collective communication routine, e.g. MPI_SendRecv.
  - Each $P_i$ creates its' own local message data array.
  - $P_i$ gets a message from lower (LO) neighbor, $P_{i-1}$, and sends its' local array to its' upper (HI) neighbor, $P_{i+1}$.
  - $P_i$ gets a message from upper (HI) neighbor, $P_{i+1}$, and sends its' local array to its lower (LO) neighbor, $P_{i-1}$.

- All tests should be run for the following conditions:
  - Number of PEs: 2, 4, 8, 16
  - Packet Sizes $= 10^n$, where $n = [1, 2, \ldots, N_{\text{max}}]$. What is $N_{\text{max}}$?
  - Test for double data types

- Report:
  - Elapsed time and statistics (see below under General Instructions).
  - Calculate Bandwidth (see below under General Instructions).
  - Table of data: Packet Size, BW, Timing, & Stats
  - Plot of the BW as a function of Packet Size and number of nodes
  - Compare to `ring.c`
Description: This homework involves measuring the performance of MPI Communications.

- Due: Tuesday, 10/23/14; written report due at start of class
- All code must be run on the student cluster
- For each problem, please create a homework directory for each problem in your home directory:
  /home/605/accountname/hw/hw3/p1
  /home/605/accountname/hw/hw3/p2
  ...
- You may work with copies of source codes, located on the student cluster in the directory /596/hw3.
- See /596/examples for batch script examples.
- To request a specific node or a group of nodes, use the following line in your batch script:
  #PBS -l nodes=node1:ppn=2 (one specific node)
  #PBS -l nodes=node3:ppn=1+node4:ppn=1 (internode communication)
HW3: Experimental Approach

For all experiments, do the following:

- Modify all inputs to come from either the command line, or from a file.
- Vary inputs as specified for each problem.
- Add/use error checking in the code where needed; use proper/clean exits.
- Add timing diagnostics where needed.
- Report statistics on the timings (see below).
- Include tables and plots of relevant results.
Calculating BW:

- BW units should be Giga Bytes per second = GByte/sec
- Estimate packet size per send or recv
- number of sends or recvs you are counting
- are you calculating BITS/sec, or BYTES/second? Convert packet size accordingly

Example estimation: Ping-pong:

\[
BW[a/b] \approx \frac{(#\text{exchanges}) \times \text{packetSize[floats]} \times \text{size[1 float]}}{\text{rawTime[\mu sec]}}
\]

\[
\approx 2 \times \frac{[\text{exchanges}] \times 10^6[\text{floats}] \times 32[\text{bits/float}]}{3 \times 10^{-3}[\text{seconds}]}
\]

\[
\approx 21 \times 10^9 \frac{\text{bits}}{\text{second}} \times \frac{1\text{Byte}}{8\text{bits}}
\]

\[
\approx 2.67 \times 10^9 \frac{\text{GByes}}{\text{second}}
\]
Run times on any computer are not reproducible, hence, it is important to analyze the distribution of a codes’ run times, and not just take one measurement.

- Standard statistical variables used to describe the distribution of the data include:
  - Max/Min (maximum/minimum values)
  - Mean (average value)
  - Median (central value)
  - Variance (variance)
  - StandardDeviation ($\sigma$) of the timings.

- To test your codes:
  - Run and time critical blocks
  - Vary key parameters (packet or problem sizes, number of processors, etc.).
  - Calculate the statistics at run-time.

- Refs:
  - http://edl.nova.edu/secure/stats/
Due: 10/23/14, at start of class.

Turn in a typewritten report, describing your solution and results; include:

- a discussion of what you did, why, and what you measured, etc.
- relevant code snippets and copies of batch scripts.
- tables of results with notation about the data and what it means.
- plots of results with notation about the data and what it means.
Reporting Results

For each of the problems, decide whether or not the questions below apply, and try to answer them.

- For the different model/run conditions, what is the saturation bandwidth and latency?
- Are your timings the same for integers and double/floating point numbers?
- How do $t_{startup}$ and $t_{bw}$ change as a function of message size, number of PEs, application (ring, ping, exchange, etc.)?
- Do your results match what is expected for tuckoo?
- Are they reproducible?
- What nodes did you end up using? Did the memory and CPU speed have an impact?
- Did the network affect your results?
- What other conditions on the machine affected your observed performance (others online, particular nodes, etc.)?