Assignment 2 - Broadcast, Reduce and Allreduce

Assignment 2 - Broadcast, Reduce and Allreduce

Overview

Write your own implementations of mpi_reduce, mpi_broadcast and MPI_Allreduce. Implement the 2LogP fan-out broadcast & fan-in reduce and the logP butterfly allreduce algorithms. Compare the performance of your implementations with the built-in MPI_Allreduce function, as well as a simple all-to-one (looped) implementation.

Development

First, draw node communication diagrams to show how data is transmitted in a 4-node network for each of the following 6 communication patterns:

- a low-to-high broadcast and a high-to-low broadcast
- a low-to-high reduce and a high-to-low reduce
- a low-to-high butterfly allreduce and a high-to-low butterfly allreduce

Using your choice of binary arithmetic (masks and shifts), or simple algebra, write rules to determine which nodes send and receive at each stage of the operation.

Software Development

Write custom broadcast, reduce, and allreduce routines that implement the following techniques:

- Write custom routines that use MPI_Send and MPI_Recv to perform a tree staged **fan-out broadcast** and **fan-in reduce**.

For these routines:

- Your routine should support the usual MPI parameters, including a pointer to the data, the number of elements, and the communicator.
- Your routine should also accept a parameter specifying whether it should use a high-to-low or a low-to-high bit traversal sequence.
- For simplicity, you may hardcode the operation and data type. That is, you may make your reduce operation sum only MPI_DOUBLEs.
- You may hardcode the rank of your choice as the target of the reduce and the source of the broadcast, but they should be the same for both operations. (Don't reduce to rank 0 and broadcast from rank np-1).

- Write a custom allreduce routine that performs an allreduce by calling your custom reduce followed by your custom broadcast. Your routine should support specifying the bit traversal method for the reduce and the broadcast independently.
- Write a custom allreduce routine using the **log-P butterfly algorithm** (see PPMPI page 77). Again, support both high-to-low and low-to-high bit traversal sequences.
- Finally, write a custom allreduce routine that uses an all-to-one reduce and an one-to-all broadcast just using a simple loop. That is, have every processor send to one processor, which receives using a loop, and then broadcast to each using MPI_Send from a loop. (This is a trivial allreduce for comparison purposes.)

Testing and Analysis

Comment on the correctness of your implementation and its performance.

- Provide the illustrations showing the bit traversal steps performed for low-to-high and high-to low reduce, broadcast, and butterfly allreduce operations for a 4-node or 8-node network. (Make sure you have 6 figures that are clearly labeled.)
- Provide sufficient output showing that your program is performing the correct send and receive operations for the custom tree reduce, the tree broadcast, and the butterfly allreduce for both bit traversal directions. (You should show 6 sets of output.)

Perform tests examining the performance of your algorithms.

- Instrument your program with MPI_Wtime and a sufficient number of outer loop operations to produce timing data for your implementations with various processor counts and data sizes. When timing, use at least 10 "warm-up" iterations to make sure that the MPI library & buffers are pre-allocated.
- Examine how increasing the amount of data and increasing the processor count impacts each algorithm. Produce plots showing the following data:
    - For fixed np=32 and np=128, provide a graph showing allreduce time by increasing message size, from 1 double (8 bytes) to 1 MB, for each of the allreduce variants as well as the time for the equivalent MPI_Allreduce function. (Remember, 1 double = 8 bytes.)
    - Then, test your implementation in the latency dominated regime by running a 1 double allreduce for np=2,4,8,16,32.
- How do the implementations perform in the latency dominated regime?
- Does your plot of time vs. np (for 1 double) look similar to a known curve? Try using a regression function to estimate the formula of the line.
- How would knowing the physical topology influence your algorithm design?

In your discussion, please make sure to cover the following points:

- How does the performance of your implementations compare with the built-in MPI_Allreduce?
- Does the bit traversal order matter for your tests?

- Does it make sense to match up the custom reduce and broadcast routines with the same bit traversal methods or the opposite? (That is... should a low-to-high reduce be paired with a low-to-high broadcast or a high-to-low broadcast? Does it matter?)

Extra Credit

Formulate your code such that any datatype can be specified, any operation may be specified (MPI_MIN, MPI_MAX, MPI_SUM, etc.) and any root may be specified for the tree-based algorithms. These modifications must be done without any extra communication.

Assignment Submission

Provide the requested diagrams, plots and analysis in a report or research paper format (a sample LaTeX paper format will be provided). Generate a code print-out using a code beautifier (vim/emacs built-in, GNU indent, etc). Create a compressed tar file username-hw2.tar.gz file of your homework directory, including all code and makefiles (but no object files or executables) and submit it to the moodle. Please print a double-sided copy of the report and code and bring to lab following the due date.