CS 5500 Homework 4

Sally Devitry (A01980316)

Approach

In this assignment, we were tasked with adding numbers from each process together and having the data in each process at the end in the following five ways:

a) Use MPI_Allreduce to find the sum immediately.

b) Use only MPI_Gather and MPI_Bcast to communicate among processes.

c) Use only MPI_Send and MPI_Recv to send all the integers to rank 0, where the sum is calculated and the result is send back to each process.

d) Use only MPI_Send and MPI_Recv in a ring interconnection topology.

e) Use only MPI_Send and MPI_Recv in a hypercube topology.

Implementation

My program defines 5 functions, each achieving one of the above approaches. Task a was the simplest and definitely the most straightforward way of adding the numbers. Task b has process 0 gather all of the data from the other processes, using MPI_Gather, and then sends the total back to all the processes using MPI_Bcast. This one was tricky at first because it was difficult for me to understand that MPI_Gather needs no sends to function correctly. Task c was similar to task b but rather than the simplicity of using MPI_Gather, we used primitive sends and receives. Exercise c was a good display that pretty much anything can be achieved with only sends and receives.

Tasks d and e were much more complicated to grasp. Using the ring and cube functions provided by Professor Watson was helpful, but there was a lot of debugging that occurred before I got the ring and hypercube to work correctly. My ring function always passes its data to the right side neighbor and does that n times, until every process has the total. The hypercube uses 'cmath's log2 function to find 'm', and does that many passes along varying axes provided by the cube function.

Some example output:

For 4 processes - `mpirun -np 4 -oversubscribe ./a.out`
ALLREDUCE- rank 0: data is equal to 6
ALLREDUCE- rank 1: data is equal to 6
ALLREDUCE- rank 2: data is equal to 6

ALLREDUCE- rank 3: data is equal to 6
GATHER- rank 2: data is equal to 6
GATHER- rank 3: data is equal to 6
GATHER- rank 1: data is equal to 6
GATHER- rank 0: data is equal to 6
RING- rank 0: data is equal to 6
RING- rank 1: data is equal to 6
RING- rank 2: data is equal to 6
RING- rank 3: data is equal to 6
HYPERCUBE- rank 1: data is equal to 6
HYPERCUBE- rank 2: data is equal to 6
HYPERCUBE- rank 0: data is equal to 6
HYPERCUBE- rank 3: data is equal to 6
SENDRECV- rank 0: data is equal to 6
SENDRECV- rank 3: data is equal to 6
SENDRECV- rank 1: data is equal to 6
SENDRECV- rank 2: data is equal to 6

For 8 processes - `mpirun -np 8 -oversubscribe ./a.out`
ALLREDUCE- rank 3: data is equal to 28
ALLREDUCE- rank 7: data is equal to 28
ALLREDUCE- rank 1: data is equal to 28
ALLREDUCE- rank 2: data is equal to 28
ALLREDUCE- rank 5: data is equal to 28
ALLREDUCE- rank 0: data is equal to 28
ALLREDUCE- rank 6: data is equal to 28
ALLREDUCE- rank 4: data is equal to 28
GATHER- rank 5: data is equal to 28
GATHER- rank 1: data is equal to 28
GATHER- rank 6: data is equal to 28
GATHER- rank 7: data is equal to 28
GATHER- rank 2: data is equal to 28
GATHER- rank 3: data is equal to 28
GATHER- rank 4: data is equal to 28
GATHER- rank 0: data is equal to 28
RING- rank 0: data is equal to 28
RING- rank 1: data is equal to 28
RING- rank 2: data is equal to 28
RING- rank 3: data is equal to 28
RING- rank 4: data is equal to 28
RING- rank 5: data is equal to 28
RING- rank 6: data is equal to 28
RING- rank 7: data is equal to 28
HYPERCUBE- rank 4: data is equal to 28
HYPERCUBE- rank 1: data is equal to 28
HYPERCUBE- rank 2: data is equal to 28
HYPERCUBE- rank 3: data is equal to 28
HYPERCUBE- rank 5: data is equal to 28
HYPERCUBE- rank 6: data is equal to 28
HYPERCUBE- rank 7: data is equal to 28
HYPERCUBE- rank 0: data is equal to 28

SENDRECV- rank 6: data is equal to 28
SENDRECV- rank 0: data is equal to 28
SENDRECV- rank 5: data is equal to 28
SENDRECV- rank 3: data is equal to 28
SENDRECV- rank 4: data is equal to 28
SENDRECV- rank 1: data is equal to 28
SENDRECV- rank 7: data is equal to 28
SENDRECV- rank 2: data is equal to 28

Concluding Remarks

Overall, this assignment was very time consuming. I thought I understood how all of the MPI methods worked, but doing this assignment helped me see a lot of the subtleties I was missing behind parallel programming. The hypercube function was pretty cool to learn about and understand better through this assignment.