Andrew Stephenson

CS 484

Lab 3

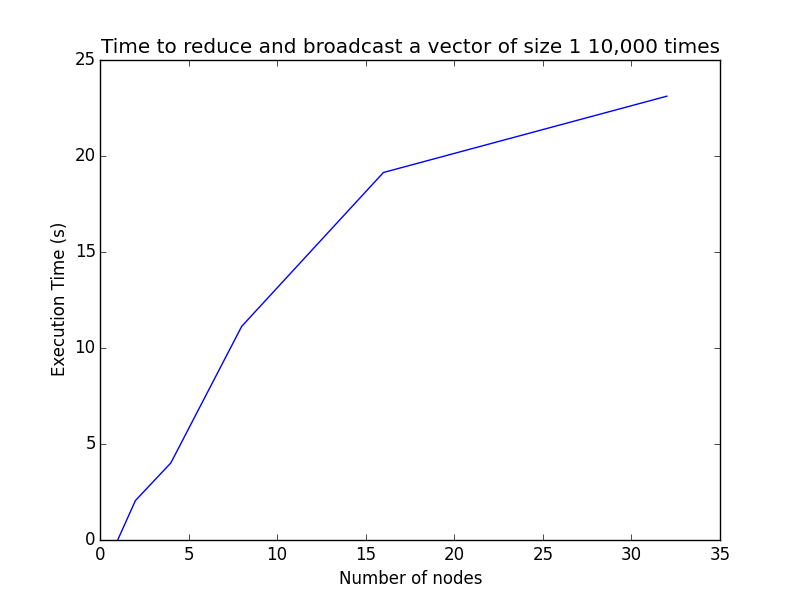
1. Correctness of the algorithm

In order to show that my implementation of vector broadcast and vector max reduce worked, I modified the benchmark code that was given to us so that it ran a reduction using both my code, and the MPI code, and then broadcast that reduced vector using both my broadcast code and the MPI code. If at any point, the vectors returned by my code were different from the MPI code, an error message was generated. I ran this program with several different numbers of nodes (1, 2, 4, 8, 16, and 32), as well as a few different vector sizes (8, 16, and 64). Each time, the program ran successfully with no error messages, meaning that the implementation successfully matched the MPI implementation’s results.

The code that demonstrates the correctness of my algorithm can be found in lab3\_validate.c

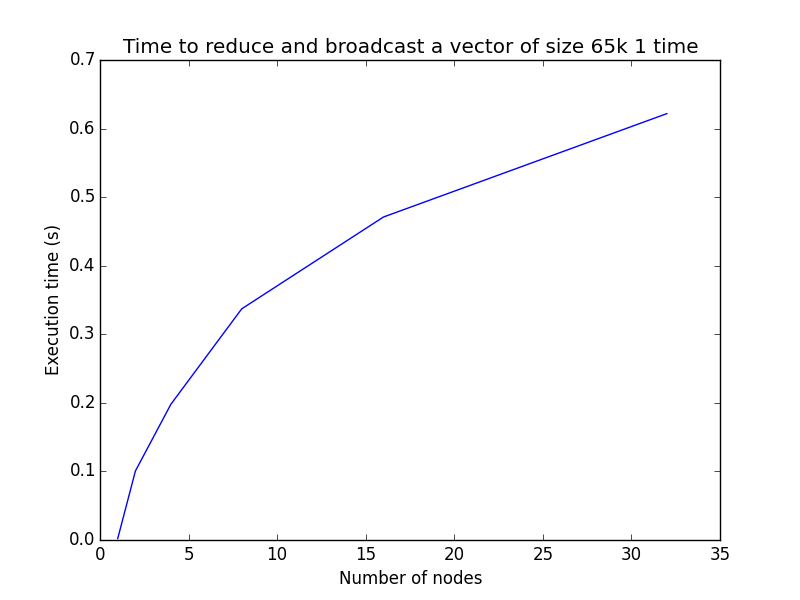
1. Experiments

I modified the provided benchmark code to use my vector max reduce and vector broadcast code instead of the MPI code. I then ran the benchmark code using various numbers of nodes with vector size 1. The results of this experiment can be seen in the graph below.



As can be seen from the graph, the execution time went up more or less linearly as the number of nodes increased. This is unsurprising, as the hypercube algorithm allows the cost of communication between the machines to scale well as additional machines are added.

I then modified the source code again so that a vector size of 65k was used. Because of the much larger size of the vector, performing a full 10,000 iterations became impractical. Therefore, instead of using 10,000 iterations like the original benchmark did, I also modified the code to only run 1 iteration. The results of running the experiment again with these modifications with varying node sizes are depicted below in the graph.



Again, the communication costs associated with higher numbers of nodes appear to scale more or less linearly. The hypercube algorithm allows us to communicate between more nodes without adding too much more to the communication costs.

1. Bandwidth calculations

The maximum bandwidth for each machine can be calculated using the formula

Bandwidth = total\_data\_transferred/execution\_time

These calculations are shown below for each of the varying number of nodes used in the experiments.

Bandwidth1 node = 65536 / 0.0021 = 31,207,619.048 bytes per second

Bandwidth2 node = 65536 / 0.1007 = 650,804.3694 bytes per second

Bandwidth4 node = 65536 / 0.1979 = 331,157.15008 bytes per second

Bandwidth8 node = 65536 / 0.3371 = 194,411.15396 bytes per second

Bandwidth16 node = 65536 / 0.4710 = 139,142.25053 bytes per second

Bandwidth32 node = 65536 / 0.6218 = 105,397.23384 bytes per second