Charles Smith
cs1645
Homework 7

1) why MPI?

MPI is designed to make parallel processes work easier. Processes can run on multiple nodes and communicate to each other through already implemented and easy to use functions, abstracting away the actual implementation of the networking.

2) Speedup

runtime of 1 million particles vs process count

mpi processes	run 1	run 2	run 3	average	speedup	efficiency
1	13936	13940	13936	13937.33333	1	1
63	1164.218161	1138.81383	1177.390627	1160.140873	12.01348359	0.1906902157
127	571.386213	580.884362	563.050975	571.77385	24.37560468	0.1919338951
255	295.108226	287.710403	285.768795	289.5291413	48.13792929	0.1887761933
511	143.960569	139.514244	138.430321	140.6350447	99.10284713	0.1939390355
1023	69.521361	66.722833	67.352977	67.86572367	205.3663113	0.2007490824

With an input of 1 million the serial runtime is very unreasonable. It was almost 4 hours on a stampede node. Parallelizing this with 63 processes shows a speedup of ~12, on which the program still took lots of time, but was much more reasonable. On 1023 cores, the program took only a little of a minute.

3) Efficiency

Looking at the table above, it can be seen that in parallel the efficiency stayed around 0.2. This is not a very Ideal number, as it shows that adding more processors will help the program, but many will be needed for a significant speedup.

4) Bottlenecks

A program like this has many bottlenecks. One being the time spent on network communications. Since the nodes are not local to each other, the time spent to transfer information from one node to another is nontrivial. Another would be the MPI overheads. Creating and distributing jobs to different nodes takes time. Finally extra steps had to be added at the end to gather the final data. This is additional time that is not required in the serial program.