Steven Comer

CS 1645

HW 7

14 April 2014

PARTICLES

1. This is a general strategy of the parallelization effort. Why did you choose those MPI operations to parallelize the program?

   I used MPI_Isend and MPI_recv to parallelize the program. I used MPI_Isend instead of the normal MPI_send to prevent any race conditions in the program and to prevent it from hanging. The use of parallelization in this program divides the work among all the ranks being used and balances the load evenly.

   I also used MPI_type_contiguous to create a derived type for the structs consisting of floating point numbers, which comprised the messages being passed between ranks. Alternatively, I could have use the MPI_BYTE option, but creating a derived datatype is more correct in this case.

2. This is a speedup analysis. Use 3 different interesting values of N and cores {63, 127, 255, 511, 1023}. Repeat each experiment 5 times and report the average value. Runtimes are measured in seconds.

   I used 3 values of N: 1024, 32768, and 1048576. I used the numbers of cores listed above. Each experiment was repeated 5 times and the average speedup is recorded. **See the attached data tables for more information.** Speedup was greatest for N=32768 and cores=1023.

3. This is an efficiency analysis. Use 3 different interesting values of N and cores {63, 127, 255, 511, 1023}. Repeat each experiment 5 times and report the average value. Runtimes are measured in seconds.

   I used 3 values of N: 1024, 32768, and 1048576. I used the numbers of cores listed above. Each experiment was repeated 5 times and the average efficiency is recorded. **See the attached data tables for more information.** Efficiency was greatest for N=32768 and cores=63.

4. This is a description of the performance bottlenecks. What is preventing the program from getting linear speedup?

> The MPI parallelization greatly decreases the runtime of the program. It does communication and calculation in steps in order to prevent a serial effect of communication. However, the communication still causes the largest delay. This can be seen as the efficiency per core drops with large number of cores and large N. These situations require more communication to work.

Experiment Data (each cell entry is an average of 5 trials)

t (N,cores)		1	63	127	255	511	1023
10	**1024**	0.056964	0.006401	0.006322	0.007621	0.007503	0.011139
15	**32768**	56.002921	0.044221	0.181148	0.083575	0.044073	0.032197
20	**1048576**	51,012.005879	292.217901	152.064741	78.588669	72.534432	35.326545

speedup (N,cores)		1	63	127	255	511	1023
10	**1024**	1	8.89923449	9.01043973	7.47460963	7.59216313	5.11392405
15	**32768**	1	1266.43271	309.155613	670.091786	1270.68548	1739.3832
20	**1048576**	1	174.568381	335.462419	649.101283	703.27987	1444.01344

efficiency (N,cores)		1	63	127	255	511	1023
10	**1024**	1	0.14125769	0.07094834	0.02931219	0.01485746	0.00499895
15	**32768**	1	20.1021066	2.43429617	2.62781092	2.48666434	1.70027684
20	**1048576**	1	2.77092668	2.64143637	2.54549523	1.37628155	1.41154784