Tyler Petrochko

CPSC 525 Assignment 5

Gravitational N-Body Problem – II

Task 0

With compiler optimization set to –O0, performance was steadily at .4 GFlop/s, and with optimization set to –O3, performance was steadily at 3.7 GFlop/s. The full runs for these tests can be found in the files "task0_no_opt.txt" and "task0.txt" respectively.

Task 1

Using strength reduction, I replaced all power/division operations with a single sqrtf() call and one reciprocal calculation. This yielded steady performance at 9.8 GFlop/s. To see the data for the full run, see "task1.txt"

Task 2

Using OpenMP (and including the strength reduction improvements from Task 1), I achieved performances of 9.8 GFlop/s, 19.6 GFlop/s, 39.2 GFlop/s, and 78.4 GFlop/s (with +/- 0.1 GFlop/s variability) respectively for 1, 2, 4, and 8 threads/cores respectively.

I experimented with various directives (including SSE4 SIMD), but ultimately found that I achieved the best i-loop performance using

```
#pragma omp for schedule(static)
```

I used the following directive for the reduction (center-of-mass calculation):

```
#pragma omp parallel for private(i) (+:comx, comy, comz)
```

These directives also yielded the highest performance for Task 4. To see the full runs for these trials, see the respective files:

task2_1thread.txt

task2_2threads.txt

task2_4threads.txt

task2_8threads.txt

Task 3

Keeping the improvements from previous tasks and replacing the Array of Structures (AoS) with a Structure of Arrays (SoA), I was able to achieve performances of 11.9 GFlop/s, 23.7 GFlop/s, 47.6 GFlop/s, and 95.1 +/- 0.1 GFlop/s for 1, 2, 4, and 8 threads respectively. Then, using 8 cores, I reran the tests for N-values 2048, 4096, 8192, 16384, and 32768 and achieved throughputs of 93.1 +/- 0.1 GFlop/s, 94.3 GFlop/s, 95.0 GFlop/s, 94.3 GFlop/s, and 93.2 GFlop/s respectively. To see the full runs for these tests, see the files:

task3_1thread.txt

task3_2threads.txt

task3_4threads.txt

task3_8threads.txt

task3_8threads_2048bodies.txt

task3_8threads_4096bodies.txt

task3_8threads_8192bodies.txt

task3_8threads_16384bodies.txt

task3_8threads_32768bodies.txt

Because this is a bit dense, here is the same data in tabular format:

Threads (N = default 16,384 bodies)	Performance (GFlop/s)
1	11.9
2	23.7
4	47.6
8	95.1 +/- 0.1

Bodies (Number of threads = 8)	Performance (GFlop/s)
2048	93.1 +/- 0.1
4096	94.3
8192	95.0
16384	94.3
32768	93.2

Task 4

Using 8 threads/cores, default 16,384 bodies, and maintaining previous improvements, I was able to achieve the following performances using tiling/unrolling:

Tilesize	Performance (GFlop/s)
2	102.3 +/- 0.1
4	92.5 +/- 0.1
8	87.7 +/- 1.6
16	91.1 +/- 0.1

Because tile size of 2 performed the best by far, I tested the performance for various N-values with tile size 2:

Bodies (Tile size = 8)	Performance (GFlop/s)
2048	99.7 +/- 0.2
4096	101.9 +/- 0.1
8192	102.5 +/- 0.2
16384	102.1 +/- 0.5
32768	102.6

To see the data for these runs, see the files

task4_8threads_tilesize2.txt

task4_8threads_tilesize4.txt

task4_8threads_tilesize8.txt

task4_8threads_tilesize16.txt

task4_8threads_tilesize2_2048bodies.txt

task4_8threads_tilesize2_4096bodies.txt

task4_8threads_tilesize2_8192bodies.txt

task4_8threads_tilesize2_16384bodies.txt

task4_8threads_tilesize2_32768bodies.txt

Overall, it seemed like there was more variation in Task 4 than Task 3, but not necessarily to a statistically significant point.

Build Info

To build the files nbody1, nbody2, nbody3, and nbody4, use the commands

```
$ module load Langs/Intel/15; module load Langs/Intel/15 MPI/OpenMPI/1.8.6-intel15

$ make
```

Running is self-explanatory; the four files can be run using the following (you can substitute any reasonable value for OMP_NUM_THREADS)

```
$ ./nbody1
$ OMP_NUM_THREADS=1 ./nbody2
$ OMP_NUM_THREADS=1 ./nbody3
$ OMP_NUM_THREADS=1 ./nbody4
```

To change the tile size in Task 4, you can alter the line

```
#define TILESIZE 2
```

… to any reasonable value.

Run Script

No build script is necessary for this assignment; just follow the steps outlined in the previous section. To run everything, I've provided the script "run.sh" which can be invoked by

```
$ bash run.sh
```

This leaves out a few instances (different tile sizes) but this can be easily changed by modifying nbody4.c. I ran all these tests after the following qsub command:

```
$ qsub -I -l procs=8,tpn=2,mem=34gb,walltime=15:00 -q fas_devel
```