Matthew Turner
CSCE435-500
Timothy Davis

Project 1 Report

Task 1

This task was simply implementing the naïve matrix multiply algorithm described in Problem 2.4 in the book.

Task 2

This task was to break down the matrix into chunks which was made easy by the SUBMATRIX routine provided. These blocks were then iterated over using three for loops similar to the method of naïve matrix multiplication and then Task 1's function was called to perform the block matrix multiplications at the innermost level of the loops. This method performs better than Task 1 because of cache locality.

Task 3

This task was to take Task 2 and add an OpenMP #pragma statement which would parallelize the loops and take advantage of the 20 cores each ada node has available to it. The trick here was determining which variables are private (indices I, J, K) and shared (matrices A, B, C).

Performance / Output

The fastest method overall was method 3, the parallel block matrix multiplication algorithm. This algorithm performed the multiplication in .393 seconds at 43.7 Gflops for a block size of 64 (peak performance point). This gave a speedup of 11.48 over the naïve sequential multiplication algorithm and a speedup of 13.49 over the block sequential multiplication algorithm.

Output given on next page.

```
echo Linux
Linux
icc -fopenmp -O3  mult.c tictoc.c -o mult -lm -ldl -lrt
./mult
omp_num_threads = 20
start simple
t 4.51446 Gflops 3.80552
blocksize 4
t 25.1775 Gflops 0.68235 err 0
t 1.71932 Gflops 9.99227 err 0 speedup 14.6439
blocksize 8
t 10.0595 Gflops 1.70782 err 0
t 0.631161 Gflops 27.2195 err 0 speedup 15.9381
blocksize 16
t 6.67139 Gflops 2.57516 err 0
t 0.490917 Gflops 34.9955 err 0 speedup 13.5896
blocksize 32
t 6.0489 Gflops 2.84017 err 0
t 0.444849 Gflops 38.6196 err 0 speedup 13.5976
blocksize 64
t 5.30528 Gflops 3.23826 err 0
t 0.393 Gflops 43.7147 err 0 speedup 13.4994
blocksize 128
t 4.59601 Gflops 3.738 err 0
t 0.423281 Gflops 40.5874 err 0 speedup 10.8581
blocksize 256
t 3.86158 Gflops 4.44892 err 0
t 0.748676 Gflops 22.947 err 0 speedup 5.15788
blocksize 512
t 3.78192 Gflops 4.54264 err 0
t 1.30738 Gflops 13.1407 err 0 speedup 2.89275
blocksize 1024
t 3.61492 Gflops 4.75248 err 0
t 2.62195 Gflops 6.55232 err 0 speedup 1.37871
blocksize 2048
t 5.19501 Gflops 3.30699 err 0
t 5.18074 Gflops 3.3161 err 0 speedup 1.00275
```