CS6230 Programming Assignment 1 Report

Che Chang, uNID: u1302280

October 22, 2022

1 Problem 1

If we wrote:

```
// result: 0.6 GFlops
#pragma omp parallel private(i)
for (i = 0; i < numElems; i++) {
    dest[i] = src[i] + 1;
}
```

Every one of the threads will run the entire for-loop (executing the loop **nthreads** times), instead of work-sharing. On the contrary, if we wrote:

```
// result: 2.3 GFlops
#pragma omp parallel for private(i)
for (i = 0; i < numElems; i++) {
    dest[i] = src[i] + 1;
}
```

This is a directive instructing threads to share work in this single loop. Hence the performance difference observed when we run both versions.

2 Problem 3

2.1 Explanation 3-(a)

2.1.1 trimm-ijk

1. placing *#pragma omp for* on i-loop: work-shared as expected, performance boosted.
2. placing *#pragma omp for* on j-loop and k-loop: j, k has dependencies on outer iterators, so the 2 inner loops are not parallelizable.

2.1.2 trimm-kij

1. placing *#pragma omp for* on k-loop: produces wrong result, due to array A and array B performing reads with k as index, causing race conditions

2. placing *#pragma omp for* on i-loop and j-loop: iterators i and j are shared by threads, and threads update these iterators in an nondeterministic way, causing undefined behavior (loops not finishing).

2.2 Explanation 3-(b)

2.2.1 trimm-ijk

1. boost performance with *#pragma omp for* on i-loop.
2. since the iteration length are not equal, I thought it would be better to adapt dynamic scheduling.

2.2.2 trimm-kij

1. outer-most loop is not parallelizable, so I parallelized the i-loop
2. I also made the iterators private variables to each thread, so they don't conflict.

3 Performance Report

3.0.1 hist_par.c

```
+ echo 'Histogram trial 1'
Histogram trial 1
+ ./hist
+ tee -a kingspeak_hist..log
Size of Data Array = 10000000
Reference sequential code performance in GigaOps Min: 1.15; Max:
    1.16
Max Threads (from omp_get_max_threads) = 32
Error when executing on 12 threads; 8 differences found
Performance (Best & Worst) of parallelized version: GFLOPS ||
    Speedup on 1/2/4/8/10/12/14/15/31 threads
Best Performance (GFLOPS || Speedup): 1.16 2.30 4.44 8.79 11.16
    7.97 8.41 8.86 2.04  || 1.00 1.98 3.82 7.56 9.60 6.86 7.23 7.63
     1.75
Worst Performance (GFLOPS || Speedup): 1.16 2.25 4.22 6.73 8.90
    7.00 7.35 0.30 0.11  || 1.00 1.94 3.63 5.79 7.66 6.03 6.33 0.25
     0.10
+ echo 'Histogram trial 2'
Histogram trial 2
+ ./hist
+ tee -a kingspeak_hist..log
Size of Data Array = 10000000
Reference sequential code performance in GigaOps Min: 1.15; Max:
    1.16
Max Threads (from omp_get_max_threads) = 32
Performance (Best & Worst) of parallelized version: GFLOPS ||
    Speedup on 1/2/4/8/10/12/14/15/31 threads
```

```
Best Performance (GFLOPS || Speedup): 1.16 2.30 4.47 8.75 11.10
    7.84 9.04 9.74 3.32  || 1.00 1.98 3.86 7.54 9.57 6.75 7.79 8.39
     2.86
Worst Performance (GFLOPS || Speedup): 1.16 2.25 4.17 6.66 8.65
    6.89 7.94 8.38 0.12  || 1.00 1.94 3.59 5.74 7.45 5.94 6.84 7.22
     0.10
+ echo 'Histogram trial 3'
Histogram trial 3
+ ./hist
+ tee -a kingspeak_hist..log
Size of Data Array = 10000000
Reference sequential code performance in GigaOps Min: 1.16; Max:
    1.16
Max Threads (from omp_get_max_threads) = 32
Error when executing on 15 threads; 16 differences found
Performance (Best & Worst) of parallelized version: GFLOPS ||
    Speedup on 1/2/4/8/10/12/14/15/31 threads
Best Performance (GFLOPS || Speedup): 1.16 2.31 4.49 8.74 11.34
    8.00 9.03 9.69 2.04  || 1.00 1.99 3.86 7.52 9.76 6.88 7.77 8.34
     1.76
Worst Performance (GFLOPS || Speedup): 1.15 2.24 4.22 6.69 7.65
    6.89 7.92 1.10 0.12  || 0.99 1.93 3.63 5.76 6.59 5.93 6.82 0.94
     0.10
```

3.0.2 trimm_ijk_par.c

```
Matrix Size = 1000; NTrials=5
Reference sequential code performance in GFLOPS Min: 2.45; Max:
    2.46
Max Threads (from omp_get_max_threads) = 16
Performance (Best & Worst) of parallelized version: GFLOPS ||
    Speedup on 1/2/4/7/15 threads
Best Performance (GFLOPS || Speedup): 2.39 4.76 9.37 16.29 27.51
    || 0.97 1.93 3.81 6.63 11.19
Worst Performance (GFLOPS || Speedup): 2.38 4.64 9.30 16.03 26.85
    || 0.97 1.89 3.78 6.52 10.92
```

3.0.3 trimm_kij_par.c

```
Matrix Size = 1000; NTrials=5
Reference sequential code performance in GFLOPS Min: 13.26; Max:
    13.36
Max Threads (from omp_get_max_threads) = 16
Performance (Best & Worst) of parallelized version: GFLOPS ||
    Speedup on 1/2/4/7/15 threads
Best Performance (GFLOPS || Speedup): 12.97 25.43 50.35 94.77
    101.96  || 0.97 1.90 3.77 7.09 7.63
Worst Performance (GFLOPS || Speedup): 12.55 25.28 49.58 89.34
    96.63  || 0.94 1.89 3.71 6.69 7.23
```

3.0.4 msort_par.c

```
+ echo 'Merge Sort trial 1'
Merge Sort trial 1
+ ./msort
+ tee -a kingspeak_msort..log
List Size = 10000001
Min/Max sequential Sort Rate: 26.4/29.0 Mega-Elements/Second
Max Threads (from omp_get_max_threads) = 32
Best & Worst Performance of parallelized version: Mega-Elts/second || Speedup on 1/2/4/8/10/12/14/15/31 threads
Best Performance (Mega-Elts/second || Speedup): 29.1 54.5 54.4 54.0 54.3 54.2 54.1 54.2 53.6  || 1.0 1.9 1.9 1.9 1.9 1.9 1.9 1.9 1.8
Worst Performance (Mega-Elts/second || Speedup): 28.2 54.1 54.2 53.4 53.7 54.0 53.7 53.3 50.8  || 1.0 1.9 1.9 1.8 1.9 1.9 1.9 1.8 1.7
+ echo 'Merge Sort trial 2'
Merge Sort trial 2
+ ./msort
+ tee -a kingspeak_msort..log
List Size = 10000001
Min/Max sequential Sort Rate: 27.7/29.4 Mega-Elements/Second
Max Threads (from omp_get_max_threads) = 32
Best & Worst Performance of parallelized version: Mega-Elts/second || Speedup on 1/2/4/8/10/12/14/15/31 threads
Best Performance (Mega-Elts/second || Speedup): 29.3 54.6 54.6 54.5 54.0 54.2 54.1 54.1 53.8  || 1.0 1.9 1.9 1.9 1.8 1.8 1.8 1.8 1.8
Worst Performance (Mega-Elts/second || Speedup): 29.3 54.6 54.5 54.2 51.3 48.5 54.0 51.7 43.5  || 1.0 1.9 1.9 1.8 1.7 1.6 1.8 1.8 1.5
+ echo 'Merge Sort trial 3'
Merge Sort trial 3
+ ./msort
+ tee -a kingspeak_msort..log
List Size = 10000001
Min/Max sequential Sort Rate: 27.2/29.4 Mega-Elements/Second
Max Threads (from omp_get_max_threads) = 32
Best & Worst Performance of parallelized version: Mega-Elts/second || Speedup on 1/2/4/8/10/12/14/15/31 threads
Best Performance (Mega-Elts/second || Speedup): 29.4 54.6 54.6 54.5 54.5 54.0 54.0 54.0 54.2  || 1.0 1.9 1.9 1.9 1.9 1.8 1.8 1.8 1.8
Worst Performance (Mega-Elts/second || Speedup): 29.3 54.5 54.5 54.5 54.3 49.9 53.3 53.2 53.2  || 1.0 1.9 1.9 1.9 1.9 1.7 1.8 1.8 1.8
```