Matthias Riegler mriegler@htwsaar.de

Exercise 1

Note: I ran the experiments on an Apple M1 processor therefore I limited the number of threads to 4 in order to only schedule the workload on the performance cores

Test runs

Processor: M1, Memory: 16GB

N	T(4)	T(4)	S(4)
50000	11174.59 ms	2387.90 ms	4.68
40000	5664.71 ms	1488.86 ms	3.80
25000	2198.10 ms	576.74 ms	3.81
10000	354.71 ms	93.35 ms	3.80
5000	88.31 ms	23.23 ms	3.80
1000	6.67 ms	1.69 ms	3.95
500	2.11 ms	0.64 ms	3.30
250	0.55 ms	0.24 ms	2.29
100	0.09 ms	0.12 ms	
50	0.02 ms	0.11 ms	
4	0.00 ms	0.10 ms	

Conclusions

The speedup factor remains rather constant across when the workload is large. Starting with N $<$ 500 we can see the speedup factor going down. Lower N (especially e.g. N=50 or N=4) show the overhead of the dataset split-up and initialisation of openmp. On larger subsets, a sublinear scaling can be denoted, most likely from a better cache hit performance with multiple processors or improved performance based on a dedicated control/main thread.

Simulate non-deterministic workloads

For this experiment, matrix/vector values between 0 and 1000 have been used, N=500

The following table is based on 5 test runs of each scenario

Execution time	Scheduling
7088.54 ms - 7313.71 ms	dynamic, 1
7573.68 ms - 7631.67 ms	dynamic, 4

Execution time	Scheduling
8613.95 ms - 8652.27 ms	dynamic, 16
8659.81 ms - 8703.49 ms	dynamic, 8
10521.26 ms - 11045.54 ms	dynamic, 32
11181.81 ms - 16384.07 ms	guided, 1
14229.29 ms - 16424.86 ms	guided, 4
14289.35 ms - 16362.35 ms	guided, 8
16840.94 ms - 16879.96 ms	guided, 64
32448.67 ms - 32714.45 ms	static

Based on the results, the `dynamic, 1` scheduling performes the best in the given workload, as it provides the optimal load balancing across available threads.

Example application output with optimal settings (N=1000)

```
xvzf@MBP14 ~/gh/xvzf/htw-mp-gpu-computing/ex1/build (git)-[main] % ./ex1 1000
[+] estimated memory bytes for input/result storage: 3.82 MB
[+] Aloocate memory and generate random matrix (1000x1000) and vector (1000)
[+] Memory allocation successful
[+] Starting normal execution
[+] Normal execution time: 6.26 ms
[+] Starting openmp accelerated execution
[+] openmp execution time: 1.70 ms
[+] Starting openmp nondeterministic execution
[+] openmp nondeterministic execution time: 29318.12 ms
./ex1 1000  116.63s user 0.31s system 396% cpu 29.508 total
```