Programming Assignment 2

ECE 759, Prof. TW Huang

Sai Tadinada

GitHub link to programming tasks:
https://github.com/phantom3012/repo759/tree/main/HW02

1 Question 1

1.a

scan.cpp can be found at https://github.com/phantom3012/repo759/blob/main/HW02/scan.cpp

1.b

task1.cpp can be found at https://github.com/phantom3012/repo759/blob/main/HW02/task1.cpp

1.c

Scaling analysis reveals an exponential increase in time starting from the point where the size of the array reachs 2^{24} elememts. See Figure 1

2 Question 2

2.a

convolution.cpp can be found at https://github.com/phantom3012/repo759/blob/main/HW02/convolution.cpp

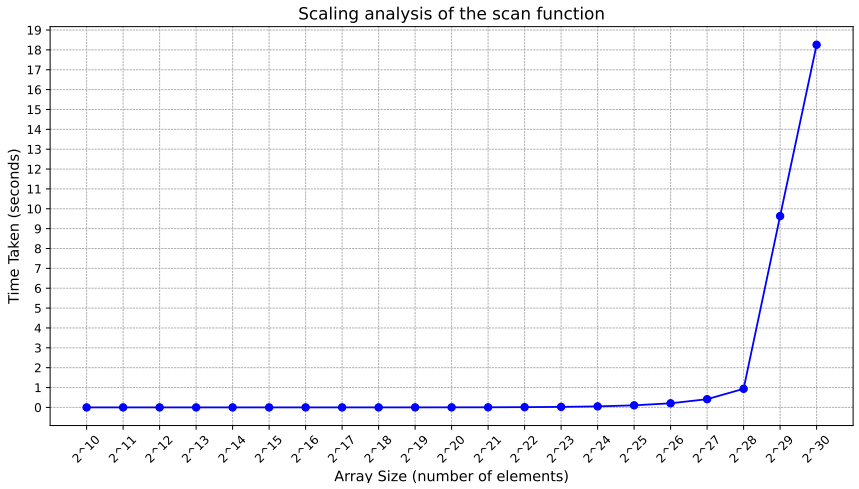

Figure 1: Scaling analysis of scan.cpp

2.b

task2.cpp can be found at https://github.com/phantom3012/repo759/blob/main/HW02/task2.cpp

3 Question 3

3.a

Implementation of mmul1 can be found at https://github.com/phantom3012/repo759/blob/main/HW02/matmul.cpp#L3

3.b

Implementation of mmul2 can be found at https://github.com/phantom3012/repo759/blob/main/HW02/matmul.cpp#L13

3.c

Implementation of mmul3 can be found at https://github.com/phantom3012/repo759/blob/main/HW02/matmul.cpp#L23

3.d

Implementation of mmul4 can be found at https://github.com/phantom3012/repo759/blob/main/HW02/matmul.cpp#L33

3.e

task3.cpp can be found at https://github.com/phantom3012/repo759/blob/main/HW02/task3.cpp

3.f

In ascending order of time taken, the functions can be arranged as follows: mmul2 $<$ mmul1 $\approx$ mmul4 $<$ mmul3

mmul2 takes the least time because all accesses to all arrays are done in row-major form. In mmul1 and mmul4, the access to one operand is done in the column major form and in mmul3, all accesses are column major. This leads to a lot of cache misses and hence, the time taken is more. When accessed in row major, the time taken is significantly less (by almost a factor of 2)

mmul4 and mmul1 take almost the same time. mmul4 on some instances may take slightly longer depending on the overheads posed by vector implementation.