Parallel Computer Architecture

Project 2

SIMD and MPI Implementations

Apostolos-Nikolaos Vailakis
2014030174

Tzanis Fotakis
2014030054

Technical University of Crete

Introduction

Within the context of this assignment, two different technologies were implemented to optimize a simplified ω statistic algorithm used to detect natural selection on a set of N DNA sequences. The algorithm calculates the ω statistic using single precision float arrays of N size, where each calculation i uses data only from the i-th cell of each array, making the algorithm extremely parallelizable. The only dependency between each calculation is the deduction of the maximum ω, which is easily solved with small compromises in the algorithm.

SIMD - SSE

The maximum ω statistic applied on a set of N DNA positions is originally calculated using scalar instructions. By using single precision float arrays it is possible to compact four of each scalar instruction into one SSE. To do this the data are originally stored in the memory, aligned to 16 bytes so they can be correctly accessed as _m128 vectors. Assuming N iterations for arrays of N size, the Iterations are reduced to the integer part of N/4 and the algorithm is modified, replacing each scalar instruction with its SSE equivalent. This raises two significant problems, a vector of max ω values, and a possible remaining (N mod 4) of uncalculated ones. The former is solved by three conventional scalar instructions extracting the maximum ω value from the vector. The latter is also solved using scalar instructions, completing the calculations for the last DNA positions. As seen in the benchmarks, a speedup between 2 and 2.5 is achieved using SIMD instructions and even bigger speedup values can be expected from using multi-core capabilities of the processor.

MPI

To further accelerate the execution, a Message Passing Interface (MPI) can be used along with the SSE instruction set. Multi-core execution can be used as, like stated above, dependency between the calculations can only be found on the deduction of the max ω. Proceeding with the multi-core execution code development one can use the code developed using the SSE instruction set and create p equally sized groups of calculations, where p is the number of processes wanted. Each group's size can be calculated as (N/4)/p, where N is the given number of the arrays of N size and p the number of the given processes. As the group size can only be an integer, there might be a remaining of ((N/4)mod p) SSE capable calculations and then another N mod 4 calculations. The former can be calculated as a group of SSE calculations and the latter as a group of normal scalar calculations.

Conclusion

The theoretical acceleration can become clear by e.g. selecting N = 100000 array size to be calculated on P = 8 processes.

- $Serial\ Execution\ Iterations\ =\ N\ =\ 100000$
- $SSE\ Execution\ Iterations \approx \frac{Serial\ Execution\ Iterations}{4} = 25000$
- $SSE\ with\ MPI\ Execution\ Iterations \approx \frac{SSE\ Execution\ Iterations}{8} = 3125$

Using the above example the maximum theoretical speedup can be calculated as:

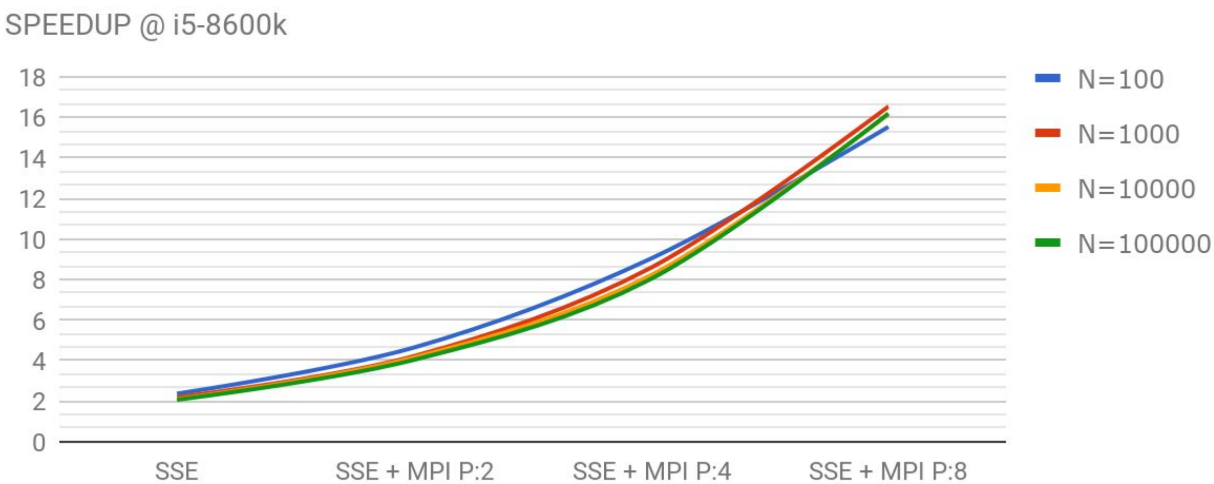
$$Speedup\ = \frac{Serial\ Execution\ Iterations}{SSE\ with\ MPI\ Execution\ Iterations} = 32$$

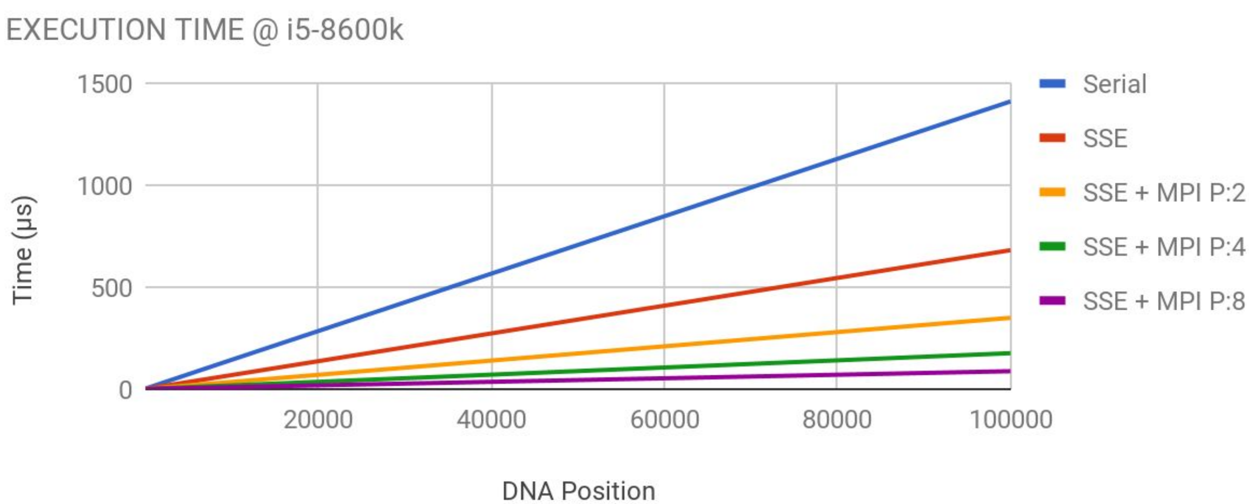
However, as shown below, the experimental results are about 2 times lower than the theoretical due to the SSE Execution speedup which is about 2 rather than the theoretical 4.

Benchmarks on two different processors with different architectures and capability of Hyper-Threading can be found below.

Benchmarks @ i5-8600k

Execution Time					
N	Serial	SSE	SSE + MPI P:2	SSE + MPI P:4	SSE + MPI P:8
100	1.723051	0.726223	0.370502	0.190496	0.110865
1000	14.738798	6.718159	3.487587	1.715183	0.89097
10000	142.585993	67.924738	34.265518	17.33923	8.82411
100000	1411.876202	681.458473	349.206924	175.620556	87.160349
Speedup					
N		SSE	SSE + MPI P:2	SSE + MPI P:4	SSE + MPI P:8
100		2.372619705	4.650584882	9.045077062	15.54188427
1000		2.193874542	4.226073213	8.593134377	16.54241781
10000		2.099176194	4.161209324	8.223317471	16.15868263
100000		2.071844812	4.043093378	8.039356179	16.19860657





Benchmarks @ i7-4710HQ

Execution Time					
N	Serial	SSE	SSE + MPI P:2	SSE + MPI P:4	SSE + MPI P:8
100	3.268957	1.301289	0.753164	0.37384	0.290632
1000	30.978918	13.025522	8.065939	3.159285	2.539158
10000	263.323784	127.993584	63.665628	31.982422	24.777412
100000	2644.415379	1282.791853	637.23731	322.507381	254.922628
Speedup					
N		SSE	SSE + MPI P:2	SSE + MPI P:4	SSE + MPI P:8
100		2.512091472	4.340299058	8.744267601	11.24775317
1000		2.378324492	3.840708193	9.805673752	12.20046882
10000		2.057320186	4.136043141	8.233390955	10.62757418
100000		2.061453207	4.149812538	8.199549948	10.37340388