Understanding effect of Cache

System Values

All the system values are find using **lscpu**

Cache size (L1) = 32KB

int size = 4B

So, matrix size that can fit in cache (s) = sqrt (32*2^10/4) = 90.5. S I will be taking s = 64

Avg. Clock Speed = 2411.690 MHz

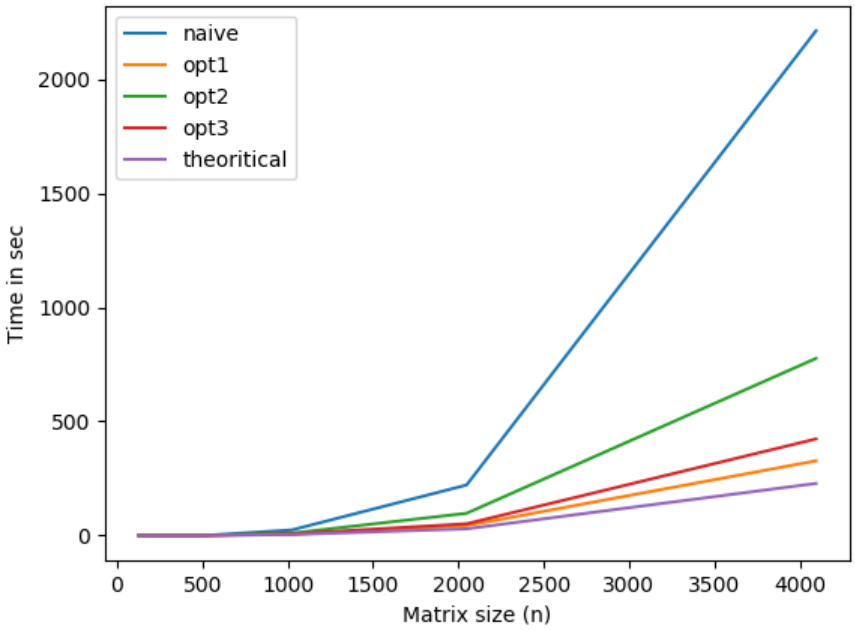
Clock time = 0.415ns

Assuming single instruction takes 4 clocks.

Considering 2*n^3 instructions for a program.

Therefore theoritical execution time = (no. of instruction) *(no of clocks in a instruction)*(clock time)

= 3.32*10^(-9)*(n^3) sec

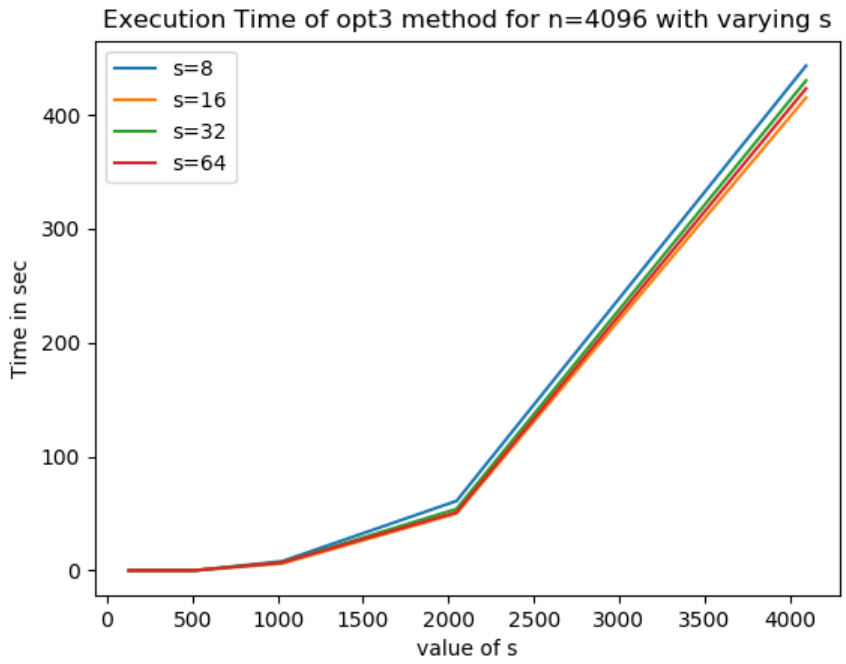


All the time are in seconds

	Theoritical	Naïve	Opt1	Opt2	Opt3
1024	3.56	24	5	10	7
2048	28.52	221	43	97	51
4096	228.15	2212	327	776	423

Optimisation1 of simply switching the for loop is best serial optimized program which is observed. Also, there is huge discrepancy between theoretical values and observed values as we don't know the exact count of instruction executed as well as the no. of clocks taken to execute an instruction. There might be other process as well executing in the background which are not included in the theoretical analysis and affecting the system performance.

Varying ‘s’



Although there is no significant difference in the execution time while varying s. s=16 and s=64 were the lowest. Theoretical calculated value of s is 90. No pattern is observed by varying s.

The data is observed for opt3 method (i.e. recursive one)

	Theoritical	s=8	s=16	S=32	s=64
1024	3.56	8	6	7	7
2048	28.52	61	50	54	51
4096	228.15	443	415	430	423

Compiler Options

Result is for opt1 for n=4096

-O1 79sec

-O2 61sec

-O3 37sec

-Ofast 32sec

This shows significant improvement in execution time when using compiler options. This result were observed for all the cases.