

Assignment 1

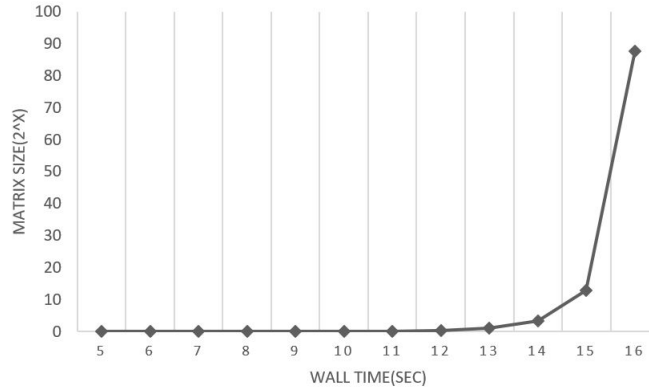
HPC

By Amirhossein Jafarzadeh Ghazi

February 20, 2020

Matrix size

First of all, I wrote a code for matrix-vector multiplication, and then I found that for the matrix size of 2^{15} with the GNU compiler (gcc), the wall time is about 13 seconds. I did not use any parallelization in this part. You can see the relationship between “Wall Time” and “Matrix Size” on the plot below:



GNU versus Intel compiler

At this step I fixed the matrix size to be 2^{15} , and then besides the gcc compiler, I used Intel compiler(icc) for compiling the program. After that I used the no compiler optimization(-O0) and aggressive optimization(-O3). In the table below, you can see wall time for each combination of compilers and optimization modes.

Results Using GNU and Intel Compiler	
Compiler	Double-Loop Time(s)
gcc	12.915142
icc	9.521260

Table 1: Results Using GNU C compiler (gcc) and Intel Compiler(icc)

Results Using different compilers and optimization criteria	
Compiler/Optimization	Double-Loop Time(s)
gcc/O0	12.708227
gcc/O3	9.853336
icc/O0	9.789014
icc/O3	10.044012

Table 2: Results Using different compilers (gcc/icc) and different Optimization criteria (O0/O3)

OMP

Finally, I used parallelization with different number of threads(1, 2, and 4) to compile and run the matrix-vector multiplication code. In addition, for calculating the efficiency, I used equation below:

$$Efficiency = S/n * P \quad (1)$$

Such that:

S : The wall time for the code compile in series.

P : The wall time for the code compiled in parallel.

n : The number of threads.

Results Using different compilers and number of threads		
Compiler/Thread	OMP Time(s)	Efficiency
gcc/1	13.613420	Negative
gcc/2	13.939576	Negative
gcc/4	14.599211	Negative
icc/1	13.842985	Negative
icc/2	13.692490	Negative
icc/4	13.593116	Negative

Table 3: Results Using different compilers(gcc/icc) and different number of threads(1,2,4)

I have no idea about these results! all of them make no sense to me, although I have checked my codes and results one thousand times!