Lab3 TDDC78 – LaplaceSolver with OpenMP

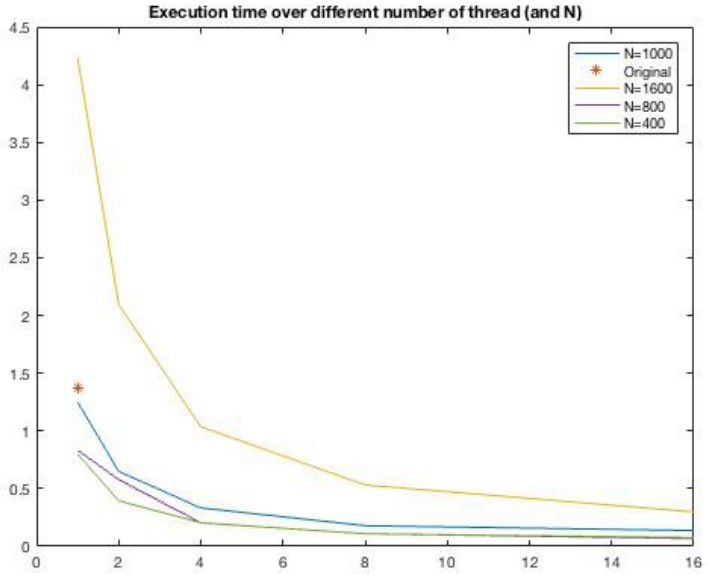
Jonathan Bosson, jonbo665

1 Program Description

The program in question is a heat equation solver using the Jacobi method to through iterations calculate the temperature. Basically what the algorithm does is to calculate the temperature for each point on the matrix by adding its forward-neighbors together.

Fortran stores arrays column wise, which means I want to iterate the matrix column wise to more efficiently use the cache memory. There's a main loop that iterates *maxiter=1000* times. In this loop I set a barrier between the first and last column to make sure all threads has read the columns. Now a new loop is created that will iterate every column of the matrix. The 'omp do' is used to allocate even amount of work to each thread. Onwards the usual Jacobi algorithm is used with the previously read lastColumn in the case where the thread is at the last iteration. The reduction method calculates the error for each thread, once the error is smaller than the maximum tolerance the program will exit the loop.

2 Result



Above is a figure showing how the execution time changes with the number of threads. *maxiter = 1000* was used and a varying problem sizes. The red *-sign is the execution time it took for the original solver to finish with N=1000. The blue line is N=1000, yellow is N=1600 and so on. It can be seen that the execution time is somewhat linear to the problem size, which indicates that the result I get is correct.