CS451 "Introduction to Parallel and Distributed Computing"
Homework 2

Submission:

- ***Due by 11:59pm of October 6, 2019 (Sunday Midnight)***
- ***Late penalty: 20% penalty for each day late***
- ***Please upload your assignment on the Blackboard with the following name: CS451_SectionNumber_LastName_FirstName_HW2***
- ***Please also upload your code on Blackboard***
- ***Please do NOT email your assignment to the instructor and TA!***

---

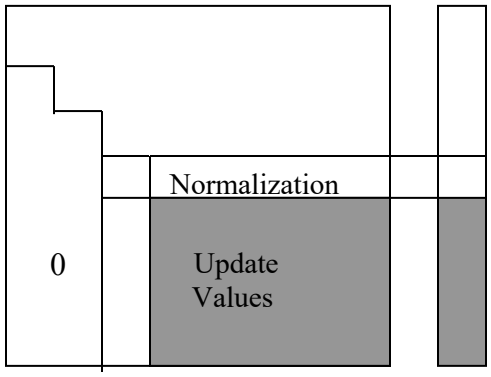
1. Shared memory programming (Pthreads & openMP)

In this problem you are asked to write two parallel algorithms (**one in Pthreads and the other in openMP**) for solving a dense linear equations of the form A*x=b, where A is an n*n matrix and b is a vector. You will use Gaussian elimination without pivoting.

The algorithm has two parts:

(a) Gaussian Elimination: the original system of equation is reduced to an upper triangular form Ux=y, where U is a matrix of size n*n in which all elements below the diagonal are zeros which are the modified values of the A matrix, and the diagonal elements have the value 1. The vector y is the modified version of the b vector when you do the updates on the matrix and the vector in the Gaussian elimination stage.

(b) Back Substitution: the new system of equations is solved to obtain the values of x.



The Gaussian elimination stage of the algorithm comprises (n-1) steps. In the algorithm, the ith step eliminates nonzero subdiagonal elements in column i by subtracting the ith row from row j in the range [i+1,n], in each case scaling the ith row by the factor A_{ji}/ A_{ii} so as to make the element A_{ji} zero. See the figure.

Hence the algorithm sweeps down the matrix from the top left corner to the bottom right corner, leaving subdiagonal elements behind it.

The whole point of parallel programming is performance, so you will be graded partially on the efficiency of your algorithm. Therefore, once you achieve a correct solution, you should perform some experimentation with your programs to try to optimize its performance. You should hand in two working and documented programs. You should provide a basic correctness argument for your solution (arguing that the relevant dependencies and other issues are taken care of by proper synchronization). In addition, you should describe some of the different versions of your program that you tried, what performance results you got out of them, and why you think your current version is reasonably efficient (or what more you could do to make it more efficient).

Suggestions:

- A sequential C code is provided on Blackboard.
- Gaussian elimination involves $O(n^3)$ operations. The back substitution stage requires only $O(n^2)$ operations, so you are not expected to parallelize back substitution.
- The algorithm should scale, assuming n is much larger than the number of processors.
- There should be clear comments at the beginning of the code explaining your algorithm.

Note on cheating: This is an individual assignment. Copying problem solutions or code is cheating. Both the person copying and the person giving the answers will be equally penalized. Make sure you do your own work. Also, make sure your code can't be copied by other students. To do that, you can change every file permission with *chmod 700 [file]*, or you can protect your whole home directory with *chmod 700 /home/username*.