CS4234 Homework #1

1. Consider a very simple computational model for studying air pollution over the U.S. The goal is to predict the concentration of a single pollutant at each point in an $n_x \times n_y \times n_z$ mesh. A typical numerical approach would require the solution of a linear system of $n = n_x n_y n_z$ equations. The linear system would be banded, with bandwidth $m = n_y n_z$.

The classical algorithm for solving linear systems of equations is Gaussian elimination with pivoting (GE); its computational complexity is $2mn^2 + 4nm$ floating point operations. Suppose there is a second algorithm, conjugate gradient (CG), whose time complexity is $2mn_2^2 + (2n_z + 8)n$.

(a) Suppose $n_x = 3000$, $n_y = 1500$, and $n_z = 30$. Estimate the amount of time it would take to solve this problem using GE on a computer capable of doing 1 trillion floating point operations per second (flops) (1 Tflop = $10^{12}$ flops).

(b) Suppose you are required to use a mesh whose relative dimensions are the same as given in Part (a). How large can the mesh be if you want to finish the GE calculation in one hour? Hint: let $n_x = 3000\alpha$, $n_y = 1500\alpha$, $n_z = 30\alpha$, and solve for $\alpha$.

(c) Suppose now that you get your hands on a 10 Tflop machine (i.e., $10^{13}$ flops). If everything goes right, you might speed up the calculation in Part (a) by a factor of 10. Suppose instead you want to solve the biggest problem you can in one hour. What problem (mesh) size can you solve using GE in one hour on the 10 Tflop machine?

(d) Repeat Part (a) for CG.

(e) Repeat Part (b) for CG.

(f) Repeat Part (c) for CG.

2. Suppose the best sequential algorithm for solving a certain problem takes time $T_s = n + n^2$, where $n$ is a measure of the problem size. Now suppose that we have three parallel algorithms, whose time complexities as a function of $n$ and $p$, the number of processors, are as follows:

$$T_1 = n + n^2/p,$$

$$T_2 = (n + n^2)/p + 1.5p^2,$$

$$T_3 = (n + n^2)/p + 100\log_2(p) + 200.$$

(a) Prepare a set of graphs showing the parallel run time, parallel speedup, and parallel efficiency achieved by each of the algorithms, for $p = 1, 2, 4, 8, 16, \ldots, 1024$ processors and for $n = 100, 1000, \text{ and } 10000$. For each measure and each $n$, it is most informative to show all three algorithms on the same graph.

(b) What observations can you make about the relative performance of these three algorithms from this data? (For example, “Algorithm 2 is slightly faster for small $p$ because all of the $n + n^2$ work is parallelized and the $p^2$ term is not too expensive when $p$ is small.”)

(c) Determine which algorithm is best asymptotically (show your work) for the case:

i. $n$ is fixed and $p$ grows;

ii. $p$ is fixed and $n$ grows;

iii. both $n$ and $p$ grow, but $n/p$ is fixed.