EECS 739: Homework 3

Due: Tuesday, March 10, 2015 (At the beginning of lecture)

Answer all of the questions listed below. I will choose one problem to grade; it will be worth 20 points. The submission instructions are identical to those for Homework #1 in regards to how to submit code questions vs. non-code questions.

You may work with at most one other student in the course to formulate ideas only; please write the name of the student with whom you worked (if any) at the top of your homework solutions.

Questions:

1. A matrix A is said to be a checkerboard matrix (my terminology) if its nonzero entries form a checkerboard pattern. That is, A should be decomposable into equally-sized blocks of size $m \times m$ which form A when put together. In addition, the nonzero entries should occur in the odd-numbered blocks in the odd-numbered rows of blocks and in the even-numbered blocks in the even-numbered rows of blocks.

   Write a pseudocode for an efficient parallel linear solver for $Ax = b$, where A is a checkerboard matrix. (No MPI commands are needed for this question. You can explain the steps without using MPI terminology.)

2. Define $h = 1/N, \mu = h^2$, and $q[i][j] = (8\pi^2 + 1)h^2 \sin(2\pi hi)\sin(2\pi hj)$, where $i, j = 0, \ldots, N-1$. Let A be of the form given in Figure 9.18 on p. 535 of the Karniadakais and Kirby book.

   Solve the matrix system $Au = q$ for u using the following methods:

   (a) Serial conjugate gradient

   (b) Serial preconditioned conjugate gradient, preconditioned using incomplete Cholesky.

   Implement the two methods in C++ and solve the systems for both $N = 4$ and $N = 20$. Here N denotes the number of points used in both the $x-$ and $y-$directions (hence the total number of grid points is N^2, which corresponds to the rank of the matrix for which we are solving). Plot the solution $u[i][j]$ at the points (hi, hj) (using either a contour or surface plot). In addition, plot $\|b - Ax\|_2$ as a function of the number of iterations. (Note that $\|v\|_2 = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}$.) Include timing results for the experiment.

   Summarize your findings.

3. Implement the parallel cyclic reduction algorithm for solving tridiagonal linear systems using C++ and MPI. (Note: I suggest that you first create pseudocode for this parallel algorithm by creating a slightly more detailed version than the version we came up with in lecture.)

   Demonstrate that your algorithm yields correct results in both serial and parallel by solving a small linear system using 1, 2, 4, and 8 nodes.

   Run scaling tests on larger tridiagonal systems; I recommend that you solve linear systems with matrices of size 1000×1000 to $10,000 \times 10,000$. Use 40-50 nodes as the maximum number of nodes for your experiments.

   Generate a plot of the speedup for each linear system solved in parallel.

   Summarize your findings.

4. Solve Problem 6.6 on p. 277 in the book by Grama, Gupta, Karypis, and Kumar. Note that the MPI_Allgather function is described on the bottom of p. 263. In addition, the function prototype for this command can be found at the top of p. 264.