COP5611 programming assignment NO. 3
A distributed application and a simplified application specific distributed checkpoint and recovery scheme

OBJECTIVES

- Experience with the development of distributed applications
- Experience with checkpoint and recovery schemes

DESCRIPTION

In this project, you first develop a distributed implementation of the Jacobi method, and then add a checkpoint and error recovery mechanism to the application. The Jacobi method is a commonly used iterative method to solve partial differential equations. It begins with an initial estimate for the solution and successively improves it until the solution is as accurate as desired. For example, applying the Jacobi method to solve the *Laplace equation* on the unit square

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$$

with boundary conditions shown in the following figure, we get

$$u_{i,j}^{(k+1)} = \frac{u_{i-1,j}^{(k)} + u_{i,j-1}^{(k)} + u_{i+1,j}^{(k)} + u_{i,j+1}^{(k)}}{4}.$$

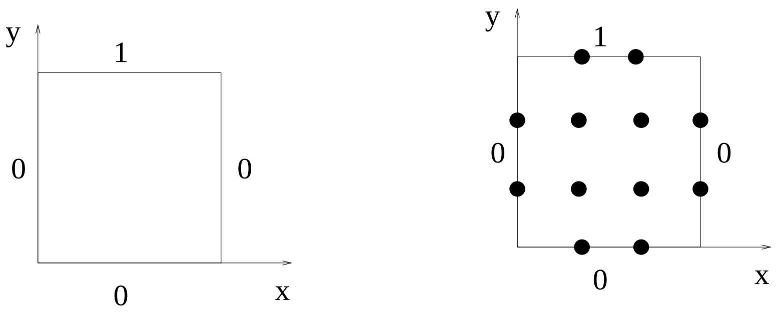

Figure 1: Boundary conditions (left) and mesh(right) for Laplace equation

In this project, you will implement a multi-process solution for the generalized Jacobi method, where

$$u_{i,j}^{(k+1)} = a_1 * u_{i-1,j}^{(k)} + a_2 * u_{i,j-1}^{(k)} + a_3 * u_{i+1,j}^{(k)} + a_4 * u_{i,j+1}^{(k)}.$$

The parameters a_1, a_2, a_3 and a_4, as well as three other parameters, a_5, a_6 and a_7, will be given in a file proj3.input. The parameters a_5 and a_6 specify the boundary condition. Assuming that the problem domain is given by Mesh[0..N+1][0..N+1], where Mesh[1..N][1..N] are the interior grid points of our interest. The boundary condition will be specified as $Mesh[i][0] = a_5 * i$, $Mesh[i][N+1] = a_6 * i$, $Mesh[0][i] = 0$, $Mesh[N+1][i] = 0$. The interior grid points should be initialized to be 0 at the beginning. The parameter a_7 specifies the error that is allowed in the solution. The program should perform the Jacobi computation iteratively until $|u_{i,j}^{(k+1)} - u_{i,j}^{(k)}| < a_7$, for any i and j, in which case, the program should stop and write the following two vectors Mesh[N/2, 0..N+1] and Mesh[0..N+1, N/2] (as binary data) to file proj3.output. Proj3.output will contain Mesh[N/2, 0], Mesh[N/2, 1], ..., Mesh[N/2, N+1], Mesh[0, N/2], Mesh[1, N/2], ..., Mesh[N+1, N/2]. The data type for all the variables we discussed should be *float*.

The program should take a command line parameter which specifies the number of processes to be used to solve the problem. When M processes are used to solve the problem, the size of the two dimensional mesh should be Mesh[0..N+1, 0..N/M+1] and the domain should be evenly distributed among the processes. Examples of domain partitioning are shown in the following figure. The maximum value for M is 4.

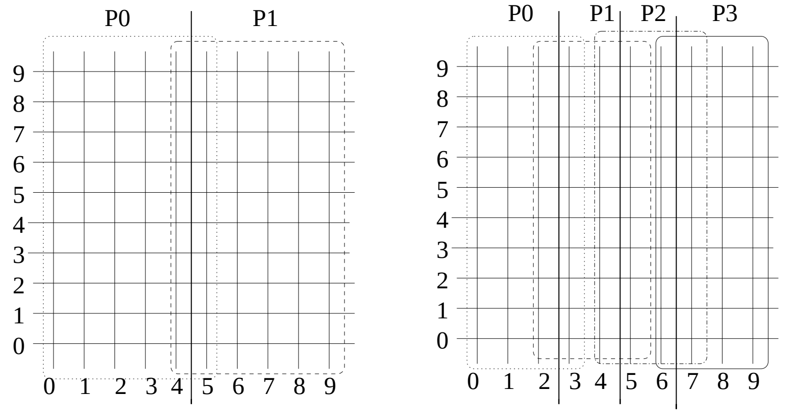

Figure 2: Examples for domain partitioning

A checkpoint and recovery mechanism is to be added to this application. The mechanism should be added in such a way that if one process failed, the user can kill all other processes, and restart the program. However, after the program is restarted, the computation continues not from scratch, but from the last checkpoint. This mechanism can be realized with two routines: recovery(char *filename) and checkpoint(char *filename). The checkpoint routine stores all the critical data in the application (in the Jacobi application, the meshes and other scalars such as the iteration number), into file *filename*. The recovery routine restores the data from the file if the file is there. Usually, the recovery routine is placed before the main loop while the checkpoint routine is placed within the main loop to perform checkpoint every N iterations.

Note: The computation part of this program is simple. However, the inter–process communication for this program is fairly difficult. Even the initialization can be complicated. Start the project at your earliest time!!!!

DEADLINES AND MATERIALS TO BE HANDED IN

This project consists of two phases. Phase one is the development of the distributed Jacobi method. This phase is due on March 28. The second phase is the development of the checkpoint and recovery mechanism. This phase is due on April 11. At the due date, you need to do a demo with Amit Karwande and send your program with all the supporting files to karwande@cs.fsu.edu. You should have a README file describing how to compile and run your program and the known bugs in your code. Students on the Panama city campus will not do the demo, so the README file will carry a very high weight.

GRADING POLICY

- Demo, README file, etc (30)
- Program works for one process (10)
- Program works for two processes (10)
- Program works for any number of processes (10)
- Checkpoint and Recovery (40)
- Extra 25 for a checkpoint/recovery mechanism that does not require all processes to be restarted. (25)

MISCELLANEOUS

Cheating policy will be strictly enforced.