CS 5740 **Parallel and Distributed Computing**
Assignment # 3 Due Date: October 25, 2011

Cooperative Communication

You will be required to simulate heat transfer across a rod of uniform material and use message broadcasting to communicate between processes.

You have a thin rod of unit length made of uniform material surrounded by a blanket of insulation so that temperature changes along the length of the rod are a result of heat transfer at the ends of the rod and heat conduction along the length of the rod. Both ends of the rod are exposed to an ice bath having temperature 0°C, while the initial tempareture at distance x from the end of the rod is $100\sin(\pi x)$.

Over time, the rod gradually cools. A partial differential equation models the temperature at any point of the rod at any point in time. The finite difference method is one way to solve a partial differential equation on a computer, giving an approximate solution.

The finite difference method solving this problem stores temperatures in a 2D matrix as follows.

$$\begin{array}{ccccccc}
0 & \cdots & u_{i-1,m} & u_{i,m} & u_{i+1,m} & \cdots & 0 \\
\vdots & & \vdots & \vdots & \vdots & & \vdots \\
0 & \cdots & u_{i-1,1} & u_{i,1} & u_{i+1,1} & \cdots & 0 \\
0 & \cdots & u_{i-1,0} & u_{i,0} & u_{i+1,0} & \cdots & 0
\end{array}$$

where $u_{i,j}$ shows the temperature at position i and time j of the rod. Notice that the end points are always 0. Each row contains the temperature distribution of the rod at some point in time. The rod is divided into n sectons of length h, so each row has $n+1$ elements. Increasing n reduces the error in approximation. Time from 0 to T is divided into m discrete entities of length k so the matrix contains $m+1$ rows. The initial temperature distribution along the length of the rod is represented by the points in the bottom row, using the expression given earlier. The temperature at the ends of the rod are represented by the left and right edges of the grid.

In the finite difference method, the algorithm steps forward in time, using values from time j to compute the value for time $j+1$ using the formula

$$u_{i,j+1} = ru_{i-1,j} + (1-2r)u_{i,j} + ru_{i+1,j}$$

where $r = k/h^2$.

Implementation The master process will receive the number of points in the rod and the time over which the simulation is to be performed. These two quantities will be provided as command line arguments.

We will assign the computation of $u_{i,j+1}$ as a primitive task to each worker process. The worker needs to know i (its identity), j (the time), and k and h (constants specified by the master process) based on user input in the command line argument). Make sure that you set up the system so that $r < 1.0$.

Each primitive computation requires communication between tasks. The computation of $u_{i,j+1}$ requires the values of $u_{i-1,j}$, $u_{i,j}$, and $u_{i+1,j}$. The task can be set up such that there is a worker corresponding to each column in the matrix, instead of having one worker for each element in the matrix. Effectively, the worker will start to compute $u_{i,j+1}$ only after it has finished computing $u_{i,j}$. So, the worker i needs to know at time $j+1$ that the workers $i-1$ and $i+1$ have completed their job. The two workers will convey this message by broadcasting this values and worker i will wait until it has received the value from the two neighbors.

What to handin

Create your programs in a directory called *username*.3 where *username* is your user name on admiral. Once you are done with everything, *remove the executables and object files*, copy the directory to admiral, and issue the following commands:

```
% cd
% ~bhatias/bin/handin cs5740 3
```