Homework 9

As in the OpenMP program, create a binary tree and initialize its nodes with a between 0 and 1. In parallel, determine how many nodes have a value of < 0.5. In your OpenMP program that solved this problem, each thread eventually sequentially traversed a subtree counting values < 0.5. In this program each process should do that. You should use the same number of processes as threads you used in HW 6. Your tasks are as follows:

1. Get a copy of (at least) the subtree that a process needs to search to that process. This can be done in three ways: (1) each process initializes the entire search tree locally. If you do this, make sure each process starts out with the same random number seed so that every process has a copy of the same tree. (2) each process initializes its subtree locally. In this case you can have each process use the default seed for its random number generator that creates values for the node. It is an interesting problem as to how to allow each process to create the same sub-tree it would have done globally (as in 3, explained next), but that problem is not part of this homework. (3) have a process initialize a full tree and send to each process the subtree that it will process.
2. If there are P processes, have each process do the traversal and counting on 1/P of the tree.
3. Gather up the counts from each of the P processes into a single count.

Turn in your program and the results of your run, including timing information compared to your OpenMP version.