MSIM 406/506: Introduction to Distributed Simulation
Homework: Synchronization Barriers

Purpose:

To develop a basic synchronization capability using a butterfly barrier.

Procedure:

1. An algorithm for the butterfly barrier was developed in class, but you should document the algorithm and include it with your assignment. Implement the algorithm so that it works with any power of 2 number of processes, but in particular you need to be able to run 1, 2, 4, and 8 processes. Your code should work by looking at the number of processes from MPI, and adapting to that size, assuming the size is a power of 2. While we have been discussing asynchronous communication in class, the barrier is a synchronous process and you can use synchronous messages to implement it.
    a. For the barrier you can use synchronous communication. You may use MPI's send and recv, but not sendrecv (and certainly not MPI's barrier).
    b. To compute the next process to communicate with, you will need to use bitwise operations in C++. Given and process id (id), and a bit to flip (bitflip – 0 flips the rightmost bit), you can compute a new process id (newid) using:
    ```
    newid = id ^ (1 << bitflip);
    ```
    The ^ performs a bitwise exclusive-or, and the << performs a left shift of the bits. So (1 << bitflip) computes $2^{bitflip}$, placing a 1 in the bitflip position, and this is then exclusive-or'ed with id to flip that bit.
2. Test and demonstrate the barrier by developing a test harness. Each "time step" in the harness should include a random time delay using either Sleep or a loop with a large iteration count to emulate work to be done by the process followed by a barrier.
3. Demonstrate your barrier with the provided program. The program performs the simulation of the n-body problem. It is a 2D gravitational simulation. You are to replace the ring barrier included with the code from your butterfly barrier.
    a. Test your software using 2, 4, and 8 processes. Implement and observe the timings resulting from different numbers of processes.
    b. Try different values for n and different time duration.
4. For extra credit, you could replace your barrier with MPI's barrier and compare the performance results.

Deliverables:

1. Algorithm (just document what we discussed in class)
2. Code (zipped project directory(ies)) – appropriately cleaned up
    a. Barrier with test harness
    b. Barrier with n-body simulation

Suggested Support Documentation:

1. http://social.technet.microsoft.com/wiki/contents/articles/mpi-application-development-for-windows-hpc-server.aspx.
2. http://www.open-mpi.org/doc/v1.6/.
3. http://pluralsight-free.s3.amazonaws.com/joe-hummel/files/tutorials/classichpc.pdf.