Programming Assignment #2
Introduction to Parallel Processing MTH/CSE 4082
Due Thursday, Feb 6

Modify the code cpi.c to replace the collective communication operations (Broadcast and Reduction) with point-to-point communication. Process 0 should send the global problem size to each of the processors; this replaces the broadcast operation. After computing their local result, each processor should send process 0 their results, and process 0 should compute the sum; this replaces the reduction operation.

After making the modifications (and verifying that you get the correct result*****) you are to perform to set of tests. Each set of tests should be run with the original code and your modified code. Since we are interested in timings, you should run each problem several times and report the median or average run time. In your write up that you turn in, document the method you used to get timing numbers.

Speedup test

Setting the number of intervals **n=600,000,000** run the codes on 1,2,4,8, and 16 processes, just change **ntasks** in the testmpi.sh file. For each code report the run time and compute the speedup relative to the same code run on one processor. In the write up, plot the speedups.

Scaled efficiency test

Keep the number of intervals *per process* equal to 100,000,000, and run the codes on 1,2,4,8, and 16 processes. For each code report the run time and compute the scaled efficiency relative to the same code run on one processor. In the write up, plot the efficiencies.

Turn in

- A listing of your modified code
- Write up including
    - 4 plots: speedups and efficiencies for original and modified code
    - Method for determining run times
    - Discussion of results

***** One way to check is to run both the original and modified code on a small problem (set n=6 instead of 600,000,000) and check that the computed values of π match in serial and parallel. For small n, the computed value should depend only on n, not which code is run or how many processes are used. For very large n, you may see slight differences in the computed π after the first 12 or so decimals using different codes or numbers of processors since the addition may be done in a different order and thus round-off error may accumulate differently.