CSCI596 Assignment 3—Parallel Computation of π and Scalability Analysis

Due: September 23 (Wed), 2020

The purpose of this assignment is to acquire hands-on experience on the ***scalability analysis*** of a parallel program — one of the key skills you learn in this class. We use a simple application that utilizes the function you have written for assignment 2, where the purpose was to:

(i) Convince ourselves that `MPI_Send()` and `MPI_Recv()` are sufficient to build any parallel programs, using global reduction as a concrete example.

(ii) Perform a unit software test of the `global_sum()` function used in this assignment.

Part I: Programming

Write a message passing interface (MPI) program, `global_pi.c`, to compute the value of π based on the lecture note on "Parallel Computation of Pi" and using the `global_sum()` function you have implemented and unit-tested in assignment 2. Please also utilize the serial program `pi.c` (which computes the value of π) in the assignment 3 package.

(Assignment)

1. ***Submit the source code*** of `global_pi.c`.

(Note)

- Insert `MPI_Wtime()` function (which takes no argument and returns the wall-clock time in seconds as `double`) to measure the running time of the program.

Part II: Scalability

In this assignment, we measure the scalability of `global_pi.c`.

(Assignment)

2. (***Fixed problem-size scaling***) Run your `global_pi.c` with a fixed number of quadrature points, $N_{\mathrm{BIN}} = 10^9$, while varying the number of compute nodes = 1, 2 and 4 with processor per node to be 1 (*i.e.*, the number of processors P = 1, 2 and 4). Plot the fixed problem-size parallel efficiency as a function of P. ***Submit the plot***.

3. (***Isogranular scaling***) In this scalability test, we consider a constant number of quadrature points, $N_{\mathrm{BIN}}/P = 10^9$, per processor for P = 1, 2 and 4. To do this, we slightly modify `global_pi.c` by defining

   ```
   #define NPERP 1000000000  /* Number of quadrature points per processor */
   long long NBIN;
   ```

   and determining the total number of quadrature points as

   ```
   NBIN = (long long)NPERP*nprocs;
   ```

   Run the resulting program `global_pi_iso.c`, and plot the isogranular parallel efficiency as a function of P. ***Submit the plot***.

(Note)

- Please perform the entire scaling tests in a single batch job to minimize measurement fluctuations, using the Slurm script, `global_pi.sl`, in the assignment 3 package.