**CS 4373/6373 High Performance Computing**

**Chapter 3 Programming Assignment**

**300 Points**

**Using Hammer**, complete the following Programming Assignments from the Book (note: these appear in 3.10 of the "Programming Assignments" section immediately AFTER the Exercises section):

NOTE: PLEASE USE THE SCHEDULER FOR RUNNING JOBS. DO NOT RUN PROGRAMS ON THE LOGIN NODE. DEVELOP AND TEST AT HOME IF POSSIBLE.

Submit a professional report with the responses, output, analysis, and answers to questions as a single PDF or Microsoft Word file. Also, submit all of the source code files, Makefiles, and job scripts (run.sh files) for the project; and describe in a table in the report all of the files uploaded: include filename, purpose (1-2 sentences), and short description (1-2 sentences).

**3.2 Estimating Pi (100 points)**

For 3.2 (and 3.2 only) define the serial algorithm first (No MPI) and time it for the parameters below. Then develop the parallel algorithm and time it for runs for 1, 2, 4, 8, 16, 32, 64 and 128 cores. Graph your results. **Get pi to ±0.05.**

PARAMETERS for 3.2: Sample size = 1E10, Random Seed (see below).

The random seed should be different (but predictable) for each rank. So use

Random seeds: rank 0 = 256, rank 1 = 257, rank 2 = 258, .... rank 31 = 287, …and so on.

**Note:** Please see discussion of the **long long int** type in the problem.

**Submit:**

* Source code
* Output to substantiate results (include in report)
* Graph of results (include in report)
* Description and discussion of the parallel algorithm, including what each task does and the communication between tasks, and identify where in your source code (using small code snippets or line numbers) the parts of the algorithm are located (include in report).
* Pseudo-code for your implementation with explanation. This can be combined with the above item.
* Is your solution weakly-scalable or strongly-scalable? What evidence supports your conclusion? Justify your response by referencing your results and analysis of those results.

**3.8 Parallel Sort (100 points)**

Time your parallel sort algorithm on 1, 2, 4, 8, 16, 32, 64 and 128 cores. Graph your timing results.

PARAMETERS for 3.8: n (number of keys to generate and then distribute) = 1.28\*1E8 Random Seed (use the same method as described above).

**Submit:**

* Source code
* Output to substantiate results (include in report)
  + For your output on 3.8 just include the first and last 20 sorted keys.
* Graph of results (include in report)
* Description and discussion of the parallel algorithm, including what each task does and the communication between tasks, and identify where in your source code (using small code snippets or line numbers) the parts of the algorithm are located (include in report).
* Pseudo-code for your implementation with explanation. This can be combined with the above item.
* Is your solution weakly-scalable or strongly-scalable? What evidence supports your conclusion? Justify your response by referencing your results and analysis of those results.

**Questions (100 points)**

Respond to the following for **each problem** above in your report. For both questions you must include references to the results you obtained executing the programs.

1. Explain your timing methodology for both problems and any limitations it may have. (50 points)

2. Can you make any analytical statements or educated speculations about the graphs, explaining performance in terms of the algorithm, software implementation or hardware? (50 points)