Fall 2020

<u>CSCI 6454: HW1 (12 points)</u>

On Shared-memory parallel computing

<u>Reproducing results from lecture slides</u>

Write programs (preferably, in C/C++) that demonstrate each of the following:

1. *A very simple Hello World program using 10 POSIX threads. You should print something like "Hello World from thread#n", here n is the thread ID. [1]*
2. *Do the same as question 1 using OpenMP threads. [1]*
3. *Regarding the computation of PI: first, implement the serial PI program (approximating integration as summation as specified in lecture slides), then, implement and simple OpenMP version of parallel program. [2]*
4. *Provide another implementation of parallel (OpenMP) PI using padding. Use different numbers for padding to compare the performance. [1.5]*
5. *Write yet another parallel (OpenMP) PI program that uses both implicit and explicit barriers. [1.5]*
6. *Write a parallel OpenMP program to compute the sum and average of all elements in a 2D array. Use a nested "for" loop—show the use of parallel-for, reduction, and different types of scheduling (static vs dynamic). [2]*
7. *For the questions 3, 4, and 5: provide the execution time using different numbers of threads/cores. Draw a set of plots that show strong scaling and parallel efficiency. For question 6, show the difference in execution time of static and dynamic scheduling. [3]*

Instructions:

1. If you want to use a different programming language, make sure that the language supports the constructs you need.
2. For each implementation, follow good programming practices. Have plenty of comments in your source file. Even though I did not quite specify the input and output formats for some problems, you should have descriptive and

intelligible prompts and outputs. You MUST have a README file that specifies your input/output and how to run and any assumptions you made.

3. Your programs MUST compile and run. Otherwise, you will lose significant points. Provide some screenshots of your program running and producing outputs.
4. For question 7, provide an MS Word (or similar) document with the results and plots.
5. To develop you code and performe experimentation, you may want to use LONI or your own computer. Most modern computers have 4 cores or so. Thus, you should be able to get at least 4 data points for performance plots. Although, the logical threads can be higher than the number of physical threads.
6. If you want to use LONI, create an account with LONI. Mention my name as your collaborator. I can then have your request approved and share with you my own allocation. Let me know your intention. You can also get a LONI account through the lab supervisor you are working with.
7. Submit solutions in a zipped archive file through Moodle. Email me your work ONLY IF Moodle is not working.
8. Name each archive file as HW1_YourFirstName_YourLastName.
9. All work must be your own. This is not a team assignment.
10. **Plan according to the submission deadline. Start early and submit on time.** Late submission might be accepted as per the instructor's discretion with significant penalty.
    DEADLINE: 10 September 2020, Thursday 11:59pm.