Parallel Lab

Due Date: Friday June 7 2019, 11:59pm

1 Getting Started

The goal of this lab is to accelerate some poorly written code we have provided. You should consider using the optimization techniques (both single threaded and multithreaded) we have covered in class to accelerate this code.

1. To start, grab the lab folder - this command will copy the lab directory to your current working directory
   cp -r /w/class.1/cs/cs33/csbin/ParallelLab .
2. To compile the code:
   make all: This will create the Test executable.
   make clean: Will delete all object files.
3. Once you have successfully compiled, you can run the executable
   ./Test
4. Each time you modify parallel.c, you need to recompile the program using the steps detailed in 2 above.
5. Remember to add your name, UCLA ID and email details in the comment section in the parallel.c file.
6. You are allowed to modify parallel.c for this lab. You **should not edit any other files except parallel.c** as we are going to measure your parallel version against our sequential version (we may change the initial state of the input matrix so do not optimize for the specific data or functions in main.c).
7. We will use **lnxsrv06.seas.ucla.edu/cs33.seas.ucla.edu** to evaluate your code. This machine is an 8-core system with hyperthreading (e.g. 16 virtual cores). You may develop your code elsewhere, but we will only grade based on the performance you obtain on lnxsrv06.seas.ucla.edu. The performance of your optimized code will depend on the load (what else is running) on the system. You can use programs like **top** to see the load of the system. After top starts press 1 to see the load on each core of the machine. We will run on an uncontested machine when we evaluate your code to ensure it is not impeded by other programs.
8. SEASnet servers are shared by all students in the engineering departments, including your fellow CS 33 students. When multiple students are running their programs the performance of your program may be drastically lower. To avoid testing problems due to high machine contention, START EARLY!

2 Submission

1. Submit **only parallel.c** via CCLE.
2. Remember to add you name, UCLA ID and email in the comment section in parallel.c.
3. The last date for submission is Friday, June 7, 2019 @11:59pm.

3 Grading

Grading will be done on lnxsrv06.seas.ucla.edu to prevent any other processes from interfering with the timing results. You will receive points proportional to the speed up:

- 8X and above 8X speed up – 100/100
- 7X speed up – 90/100
- 6X speed up – 80/100
- 5X speed up – 70/100
- 4X speed up – 60/100
- 3X speed up – 50/100
- 2X speed up – 40/100
- less than 2X speed up – 0/100

NOTE:

- **If your solution does not provide the same result as the sequential version, you will receive a 0 regardless of your speed up.** You need to ensure that the code you provide is functionally equivalent, even if we change main.c to supply a different matrix or helper functions. We will not make changes to sequential.c
- **There are NO BONUS POINTS for this lab. Any legal speed up beyond 8X will receive 100/100**

4 Hints

You can use the optimization techniques we have covered in class, including loop tiling, OpenMP, and more.