Threads can be used to reduce “wall clock” run time on a multi-CPU system. For this assignment you are to get a copy of my program “jacobi.c” from the /home/phil/public/csci347 directory on the lab machines and modify it to use Pthreads, specifically so it can run faster.

Jacobi Iterations is a method of iterative numerical solution to Laplace’s Equation. It is used in many circumstances. Heat distribution, or electrical fields are common uses. It provides a good platform to investigate threading and speed-ups using Pthreads. The area under consideration is represented by a 2-D array of values. Some values are fixed and the other values are given an initial value. The computation recomputes all non-fixed values in the 2-D array at every iteration. A point’s value in the next iteration is the average of the four points nearest the point in the current array. The difference from the current value to the new value at point \(i, j\) is known as \(\Delta_{i,j}\). The computation quits when the maximum of the \(\Delta_{i,j}\) values for single iteration is less than some tolerance, typically a small number like 0.000001.

The program provided for you does this computation. The edges of the 2-D array are the fixed values. For this solution, row 0 is some non-zero value and all other edges are zero. Each iteration is computed by the “next_gen()” function and it returns the maximum \(\Delta_{i,j}\) for that iteration. The main program allocates two arrays and computes the value alternative from one to the other.

Notice that jacobi.c does command line “option” processing using the getopt(3) library function. For this assignment you will be adding a few options to this program and you are to add them to the getopt(3) processing.

The following are the tasks you need to do for this assignment:

- (5 points) Create a jacobi directory under your csci347_w18 gitlab project and copy jacobi.c unmodified into this directory. Commit it and push it.

- (100 points) Copy jacobi.c to a new file named pt-jacobi.c and modify that version do the computation using Pthreads. First, add a new flag to the getopt(3) processing for the flag “-n num_threads”. The default should be 8. Don’t forget to update the “usage()” function. This will require the rewrite of the “next_gen()” function and it might be best to move the entire compute loop from main() into its own function.
that creates the threads and manages synchronization of the threads until the computation is complete. Here are some issues you will need to think about:

- How to equally divide up the work between the threads.
- How to make sure threads are all working on the same iteration at the same time.
- How to determine if an iteration met the tolerance and the computation is done.

Now, change both jacobi.c and pt-jacobi.c to take a new flag, "-T", that asks for the program to time the computation loop. It should report both clock time and CPU time. (hint: man clock, man gettimeofday)

You should be able to verify that your threaded solution calculates essentially the same thing by computing an 10 by 10 size to a tolerance of 0.001 and verify that the values between the sequential and threaded versions differ by at most 0.0005.

- Let $T_s$ be the the “wall clock” time of the serial solution. Let $T_t$ be the “wall clock” time of the threaded solution. The “speed up” is defined as $T_s/T_t$. First, compare the threaded solution’s time compared to the original solution. What should you expect? Next, find the speedups, 2 threads through 16 threads. You can use the machines in CF416 and CF414 to do your timings. Make sure nobody else is using the machine before you do your timings. htop(1) can show you how much each CPU is being utilized. You can ssh into the lab using the name cf416-01 to cf416-20 and cf414-01 to cf414-20.

- (90 points) Write a report of between 750 and 1000 words, where figures do not count toward word count, describing your experiments in speed-ups. It should have a title line, an author line, and cover the following points:

1. Your PThreads implementation. Give a high level description, don’t quote code.
2. Your experiments and how you conducted them.
3. Your results. Graphs are great things to include. (man gnuplot)
4. An explanation of your results and if things were not as you expected, explain why.
5. A final conclusion paragraph.

Your paper needs to have both source (.tex, .ods, or whatever format you use) as well as a final .pdf document and they need to be in your git repo in the jacobi directory.

- Turn-in will be only by gitlab. When you are done, branch a5. The time you branch a5 will be used as the you turned in a5. Any commits on branch a5 will be used as the final turn in time. This assignment must be “turned in” by 9:00AM on the day due or it will be considered late. A late assignment may be turned in up until the beginning of the final.