

## Programming Assignment 2 – pthreads

Early Due Date (+5%): Fri. 10/25, 11:59pm

Final Due Date: Mon. 10/28, 11:59pm

### Assignment

Using your implementation of a model scientific computing problem from programming assignment 1, investigate threading this application with posix threads.

#### Part One - Disposable Threads

Implement a pthread parallel version of your sequential program using a disposable threads model which meets the following requirements:

- create / destroy your threads within the convergence loop, creating new threads for each loop iteration.
- implement the number of threads as the third program parameter.  
Your program should now execute similar to: `disposable <affect_rate> <epsilon> <num_threads> <infile>`
- Name this program: `<lastname>_<firstname>_disposable.c` (for example: `jones_jeffrey_disposable.c`).
- In other respects, conform to the requirements for assignment 1.

#### Part Two - Persistent Threads

Implement a pthread parallel version of your sequential program using a persistent threads model which meets the following requirements:

- Move thread creation outside of the convergence loop, and add a barrier or barriers as necessary within your program to properly synchronize the threads.
- Destroy all threads only one time.
- Name this program: `<lastname>_<firstname>_persistent.c` (for example: `jones_jeffrey_persistent.c`).
- Your program should now execute similar to: `persistent <affect_rate> <epsilon> <num_threads> <infile>`
- In other respects, conform to the requirements for part one above.

### Testing and Submission Instructions

- Provide a single make file which will build both program versions, naming the executables "*disposable*" and "*persistent*", as appropriate. Your makefile should execute with the command "**make**," with no parameters.
- Compile your programs, as with assignment 1, with optimizer level3 (-O3) and (-lrt) <small LRT> option. (Note: the -pthread option may be used in lieu of or in addition to -lrt on some systems.)

- Instrument your programs as with assignment 1.
- Determine the optimal number of threads for each version of your threaded program (disposable and persistent). Using the values for epsilon and affect-rate you developed in lab 1 to achieve the target serial run-time, collect timing results for both of your optimized (thread number) parallel versions using test data file testgrid\_400\_12206.
- Other than execution times, your parallel program should achieve the same results (final convergence values, number of iterations required) as your serial program.
- Ensure the program can be compiled with "make", before submitting.
- Create a directory "cse5441\_lab2". Within this directory, place:
  - all program files (.c and .h files);
  - makefile
- Submit a .pdf report with your performance results to Carmen.
- You will submit your program on OSC, instructions to be provided.

## Report Requirements

- Present your program output and timing results for the testgrid\_400\_12206 test file from your sequential program as well as both versions of your pthreads parallel program (for all requested numbers of threads).
- Summarize timing results, being sure to answer the following questions (at a minimum).
  - Did this program perform better sequentially or in parallel?
  - Which number of threads was most effective?
  - Which parallel version (disposable or persistent) was most effective?
  - How did your results match or conflict with your expectations?
  - Were there any unexpected anomalies in the timing information collected?
  - Which timing methods seem best for parallel programs? How does this compare with your expectations?
- If your parallel program produces results which differ from your serial version, point this out and explain.
- Submit all reports in .pdf format.