2020-2021 EE4107 : Introduction to Parallel Programming Techniques

Shared Memory Programming with Pthreads
Programming Assignment – 5

Exercise – 1 [3 points]

Write a Pthreads program that implements the trapezoidal rule. Use a shared variable for the sum of all the threads' computations. Create implementations based on busy-waiting, mutexes, and semaphores to enforce mutual exclusion in the critical section and compare them in terms of performance. What advantages and disadvantages do you see with each approach? Identify suitable performance metrics and running scenarios to support your answer.

Exercise – 2 [3 points]

Recall that in C a function that takes a two-dimensional array argument must specify the number of columns in the argument list. This is quite common for C programmers to only use one-dimensional arrays, and to write explicit code for converting pairs of subscripts into a single dimension. Modify the Pthreads matrix-vector multiplication, which we examined during the lecture sessions, so that it uses a one-dimensional array for the matrix and calls a matrix-vector multiplication function. How does this change affect the run-time? Run your program and take measurable results to support your answer.

Exercise – 3 [4 points][Pthread Implementation]

a) Modify the matrix-vector multiplication program so that it pads the vector `y` when there's a possibility of false sharing. The padding should be done so that if the threads execute in lock-step, there's no possibility that a single cache line containing an element of `y` will be shared by two or more threads. Suppose, for example, that a cache line stores eight `doubles` and we run the program with four threads. If we allocate storage for at least 48 `double`s in `y` , then, on each pass through the for `i` loop, there's no possibility that two threads will simultaneously access the same cache line. [2 points]

b) Modify the matrix-vector multiplication so that each thread uses private storage for its part of `y` during the for `i` loop. When a thread is done computing its part of `y`, it should copy its private storage into the shared variable. [1 point]

c) How does the performance of these two alternatives compare to the original program? How do they compare to each other? [1 point]