# ITCS 6144/8144 Assignment - Pthread Lock

Prepared by: Abdullah Al Raqibul Islam (UNCC ID # 801151189)

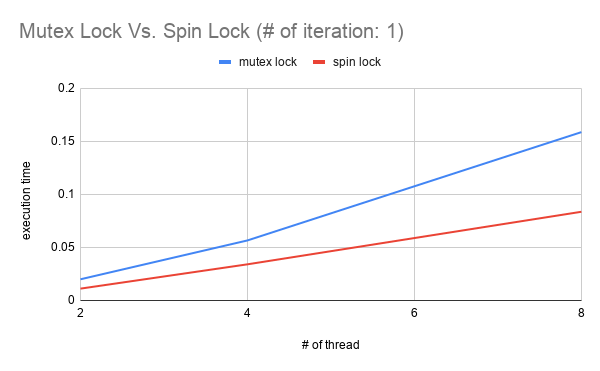
**Test platform:**

* Processor: Intel(R) Xeon(R) CPU E5-2620 2.00GHz (12 Core)
* Linux version 5.0.0-27-generic
* gcc version 7.4.0 (Ubuntu 7.4.0-1ubuntu1~18.04.1)

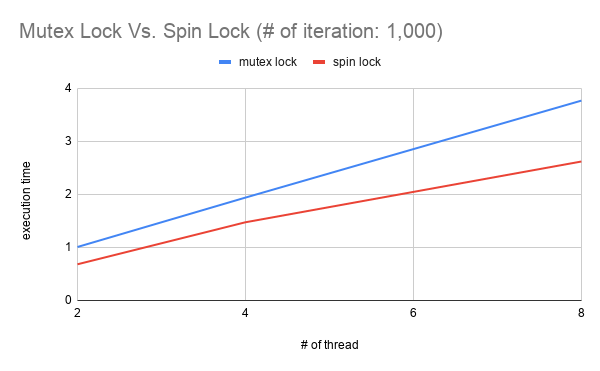
**Program 1:** Mutex Lock and Spin Lock

For this problem, I prepared two programs using both pthread Mutex Lock and Spin Lock. The execution time is showing bellow:

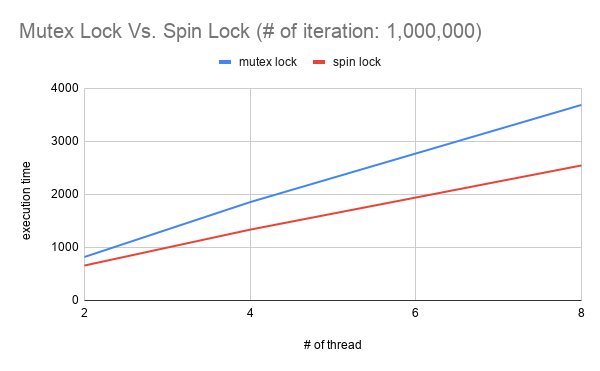
|  |
| --- |
| $ ./build/p1\_mutex\_lock 2  [Mutex Lock] 2 threads with iteration 1 took: 0.020013  [Mutex Lock] 2 threads with iteration 1000 took: 1.009162  [Mutex Lock] 2 threads with iteration 1000000 took: 820.893171  $ ./build/p1\_mutex\_lock 4  [Mutex Lock] 4 threads with iteration 1 took: 0.056631  [Mutex Lock] 4 threads with iteration 1000 took: 1.940355  [Mutex Lock] 4 threads with iteration 1000000 took: 1855.090460  $ ./build/p1\_mutex\_lock 8  [Mutex Lock] 8 threads with iteration 1 took: 0.158721  [Mutex Lock] 8 threads with iteration 1000 took: 3.768429  [Mutex Lock] 8 threads with iteration 1000000 took: 3687.262856 |



|  |
| --- |
| $ ./build/p1\_spin\_lock 2  [Spin Lock] 2 threads with iteration 1 took: 0.011201  [Spin Lock] 2 threads with iteration 1000 took: 0.683132  [Spin Lock] 2 threads with iteration 1000000 took: 659.433738  $ ./build/p1\_spin\_lock 4  [Spin Lock] 4 threads with iteration 1 took: 0.034115  [Spin Lock] 4 threads with iteration 1000 took: 1.475001  [Spin Lock] 4 threads with iteration 1000000 took: 1336.079932  $ ./build/p1\_spin\_lock 8  [Spin Lock] 8 threads with iteration 1 took: 0.083648  [Spin Lock] 8 threads with iteration 1000 took: 2.620485 [Spin Lock] 8 threads with iteration 1000000 took: 2546.134466 |



Reason behind the result:



**Program 2:** Condition Variable

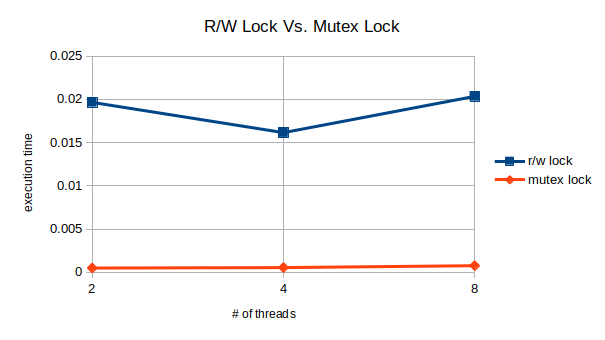
For this problem, I develop a program using pthread Condition Variable. The program takes n as the input parameter to denote how many threads will be created. n will be in range of [2, 4, 8]. n-1 threads randomly add 1 to a shared variable k in an interval of 100 ms. One thread wait until that shared variable reaches 100 and print “Reached to 100!”.

**Program 3:** Reader/Writer Lock

For this problem, I prepared two programs using both pthread Reader Writer Lock and Mutex Lock. The execution time is showing bellow:

|  |
| --- |
| $ ./build/p3\_rw\_lock 2  [Reader/Writer Lock] 2 reader threads took: 0.019652  $ ./build/p3\_rw\_lock 4  [Reader/Writer Lock] 4 reader threads took: 0.016154  $ ./build/p3\_rw\_lock 8  [Reader/Writer Lock] 8 reader threads took: 0.020346 |

|  |
| --- |
| $ ./build/p3\_rw\_lock\_mutex 2  [R/W Mutex Lock] 2 reader thread took: 0.000484  $ ./build/p3\_rw\_lock\_mutex 4  [R/W Mutex Lock] 4 reader thread took: 0.000534  $ ./build/p3\_rw\_lock\_mutex 8  [R/W Mutex Lock] 8 reader thread took: 0.000756 |



Note: As I have noticed a huge fluctuation in the execution time, I run each of the programs 100 times and then take the average.

Reason behind the result: