Programming Assignment 5
Comparison of TAS, TTAS and BackOff Locks
Submission Date: 9th April 2019, 9:00 pm

Goal: The goal of this assignment is to implement the three locking operations discussed in the class: TAS, TTAS and Backoff Locks. Then compare the performance of these locks by measuring two parameters: average and worst-case waiting time for threads to obtain the locks. You have to implement these algorithms **in C++**.

Details. As explained above, you have to implement three lockins algorithms: TAS, TTAS and Backoff Locks. You have to implement them **in C++**. Each locking algorithm implements a class consisting of two methods: lock and unlock. Your program will read the input from the file and write the output to the file as shown in the example below.

To test the performance of locking algorithms, develop an application, lock-test is as follows (which is same as Assignment-2). Once, the program starts, it creates n threads. Each of these threads, will enter critical section (CS) k times. The pseudocode of the test function is as follows:

Listing 1: main thread

```
1  void main()
2  {
3      ...
4      ...
5      // Declare a lock object which is accessed from all the threads
6      Lock Test = new Lock();
7      ...
8      ...
9      create n testCS threads;
10 }
```

Listing 2: testCS thread

```
1
2  void testCS()
3  {
4      id = thread.getID();
5      for (i=0; i < k; i++)
6      {
7          reqEnterTime = getSysTime();
8          cout << i << "th CS Request at " << reqEnterTime << " by thread " << id;
9          Test.lock();
10         actEnterTime = getSysTime();
11         cout << i << "th CS Entery at " << actEnterTime << " by thread " << id;
12         sleep(t1);
13         Test.unlock();
14         exitTime = getSysTime();
```

```
15          cout << i << "th CS Exit at " << exitTime << " by thread " << id;
16          sleep(t2);
17      }
18  }
```

Here $t1$ and $t2$ are delay values that are exponentially distributed with an average of $\lambda 1, \lambda 2$ milliseconds. The objective of having these time delays is to simulate that these threads are performing some complicated time consuming tasks. The $Test$ variable declared in line 6 declared in main is an instance of Lock class and is accessible by all threads.

As described above, you will measure average and worst-case time taken by threads to enters the CS (eat in this case). The worst-case time taken by a thread to enter the CS gives an indication of starvation in threads.

Input: The input to the program will be a file, named inp-params.txt, consisting of all the parameters described above:$n, k, \lambda 1, \lambda 2$. A sample input file is: 100 100 5 20.

Output: Your program should output to a file in the format given in the pseudocode for each algorithm. A sample output is for TAS lock is as follows:

TAS lock Output:
1st CS Requested at 10:00 by thread 1
1st CS Entered at 10:05 by thread 1
1st CS Exited at 10:06 by thread 1
1st CS Requested at 10:01 by thread 2
.
.
.

Similar to TAS lock, you have to show the output for TTAS and Backoff Locks output. The output must demonstrate the mutually exclusive execution of the threads.

Report: You have to submit a report for this assignment. This report should contain a comparison of the performance of all the three locking algorithms. You must run both these algorithms multiple times to compare the performances and display the result in form of a graph. You must average each point in the graph plot by 5 times.

You run both these algorithms varying the number of threads from 10 to 50 while keeping other parameters same. Please have k, the number of CS requests by each thread, fixed to 10 in all these experiments while having $\lambda 1, \lambda 2$ as 1 and 2.

You measure the average and worst-case time taken to enter the CS by each thread and plot the results as two graphs. In both these graphs, the x-axis will vary the number of threads from 10 to 50. The details of y-axis is shown below:

- Graph1 - Average times: The y-axis will consist of the average times.
- Graph2 - Worst-case time: The y-axis will consist of the average times.

Finally, you must also give an analysis of the results while explaining any anomalies observed.

Deliverables: You have to submit the following:

- The source files containing the actual program to execute. Please name them as: tas-<rollno>.cpp, ttas-<rollno>.cpp, backoff-<rollno>.cpp.
- A readme.txt that explains how to execute the program

- The report as explained above

Zip all the three files and name it as ProgAssn5-<rollno>.zip. Then upload it on the google classroom page of this course. Submit it by the above mentioned deadline.