Assignment 8
Mutual Exclusion with Bounded and Unbounded Waiting Times

Submission Date: 18th October 2015, 9:00 pm

Goal: To implement bounded mutual exclusion algorithm for n threads using compare_and_swap (cas) operation. Then compare its performance with the normal unbounded mutual exclusion algorithm which is again implemented using cas operation.

Details. You have to implement the bounded and normal locking algorithms using cas operation. Each locking algorithm implements a class consisting of two methods: lock and unlock. Denote both the classes as Bounded and Unbounded. Follow the bounded and unbounded implementations using CAS given in the book.

To test the performance of locking algorithms, develop an application, lock-test is as follows. 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

```
void main()
{
    ...
    ...
    // Declare a lock object which is accessed from all the threads
    Lock Test = new Lock();
    ...
    ...
    create n testCS threads;
}
```

Listing 2: testCS thread

```

void testCS()
{
    id = thread.getID();
    for (i=0; i < k; i++)
    {
        reqEnterTime = getSysTime();
        cout << i << "th CS Request at " << reqEnterTime << " by thread " << id;
        Test.lock();
        actEnterTime = getSysTime();
        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 $\lambda1, \lambda2$ seconds. 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.

Input: The input to the program will be a file, named inp-params.txt, consisting of all the parameters described above:$n, k, \lambda1, \lambda2$. 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 as follows:

```
Unbounded Waiting Time 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
.
.
.

Bounded Waiting Time lock output:
1st CS Requested at 11:00 by thread 1
1st CS Entered at 11:05 by thread 1
1st CS Exited at 11:06 by thread 1
1st CS Requested at 11:01 by thread 2
.
.
.
```

The output should demonstrate that mutual exclusion is satisfied.

Report: You have to submit a report for this assignment. This report should contain a comparison of the performance of bounded and unbounded waiting time lock algorithms. You must run both these algorithms multiple times to compare the performances and display the result in form of a graph.

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. You measure the average time taken to enter the CS by each thread.

The graph in the report will be as follows: the x-axis will vary the number of threads while the y-axis will show the average time taken to enter the CS by each thread. Finally, you must also give an analysis of the results while explaining any anomalies observed.

Deliverables: You have to submit the following:

- The source file containing the actual program to execute named as Bounded-<rollno>.c and Unbounded<rollno>.c

- A readme.txt that explains how to execute the program
- The report as explained above

Zip all the three files and name it as Assn8-<rollno>.zip. Then upload it on the google classroom page of this course. Submit it by **18th October 2015, 9:00 pm**.