Spring, 2017 ECEN 4313, CSCI 4830-010

Concurrent Programming

Homework 2

Assigned: 02/02/2016; Due: 02/14/2016, before class.

Please submit your homework through Moodle.

Problem 1 (20 points)

Implement the bakery algorithm in Java. You need to implement appropriately the lock and unlock functions of the algorithm. This should be a simple exercise designed to make sure that you can fill in the details to the algorithm provided in the textbook.

Submit two well commented .java files, one with the Bakery class and the other with the testing code which uses the lock and unlock functions of the Bakery class.

Problem 2 (40 points)

Filter lock algorithm is one way to generalize the two-thread Peterson lock to n threads. Another way, suggested in class, is to arrange a number of two-thread Peterson locks in a binary tree. Suppose n is a power of two. Each thread is assigned a leaf lock which it shares with one other thread. In the tree-lock's `lock()` method, a thread acquires every two-thread Peterson lock from that thread's leaf to the root. The `unlock` method of the tree-lock unlocks each of the two-thread Peterson locks that the thread has acquired, from the root back to the thread's leaf.

1. Write an implementation of tree-lock that satisfies mutual exclusion and deadlock-freedom.

2. Benchmark your implementation with $n = 8$ and $n = 64$ threads and compare its performance to that of the filter lock. For each benchmark, use a test where the lock protects a counter. In each test, the counter should be incremented (and thus the lock acquired and released) a large number of times. Run each test 3 times and report the average time it takes, for the two different algorithm (filter-lock and tree-lock), to complete the counter increment tests.

Submit the code as well as a table with the experimental results.

Problem 3 (40 points)

The M-exclusion problem is a variant of the starvation-free mutual exclusion problem. We make two changes: as many as M threads may be in the critical section at the same time, and fewer than M threads might fail (by halting) in the critical section. A solution to this problem must satisfy the following conditions:

1. M-Exclusion: At any time, at most M threads are in the critical section.
2. M-Starvation-Freedom: As long as fewer than M threads are in the critical section,then some thread that wants to enter the critical section will eventually succeed (even if some threads in the critical section have halted).

Modify the n-thread Filter Lock algorithm to turn it into an M-exclusion algorithm. Write the algorithm similar to the Filter Lock code in the text book.