Course
CS-E4110
Course materials
Your points
MyCourses
Zulip Chat

« 3 Countdown latch -- A versatile synchronization tool implemented usin... Course materials 5 Feedback »
CS-E4110 / Round 2 - Scala concurrency - Part 2 / 4 Lock-free Stack -- Implementing lock-free concurrent algorithms.

Assignment description My submissions (2/10)

Lock-free Stack -- Implementing lock-free concurrent algorithms.

Lock-free Stack

In the past exercises, we have seen how locks are important low-level synchronization mechanisms that enforce mutual exclusion and concurrency control policy. However, there are inherent problems that come with locking, its improper implementation, and use. These include deadlock, priority inversion and arguably performance to name a few.

Lock-free programming provides an alternative thread-safe access to shared memory (mutual exclusion) without the use of a synchronization mechanism involving locks. Lock-free programming in practice relies on hardware support. It is often implemented in terms of simpler primitives such as Compare And Set(CAS). Even though designing generalized lock-free algorithms is hard it is still worthwhile to design and implement lock-free data structures. A lock-free stack is one such thread-safe structure that is easy to design and implement.

Code

Download the assignment template here

Task

In this exercise, we implement a lock-free stack using an atomic reference. Assume that you have a class implementing an atomic reference called `SimpleAtomicReference` with an interface as follows:

```
class SimpleAtomicReference[V](initValue: V) {
    protected var value:V
    def compareAndSet(expect: V, update: V): Boolean
    def get: V
    def set(newValue: V): Unit
}
```

Hint

Base your implementation on `compareAndSet(expect: Int, update: Int): Boolean` method.

LockFreeStack.scala

Choose File No file chosen

Submit

Earned points

25 / 25

Exercise info

Assignment category
Programming exercises

Your submissions
2 / 10

Deadline
Tuesday, 16 November 2021, 14:00

Late submission deadline
Tuesday, 23 November 2021, 14:00 (-30%)

Total number of submitters
52

« 3 Countdown latch -- A versatile synchronization tool implemented usin... Course materials 5 Feedback »