4.3 Lecture Summary

4.3 Linearizability

Lecture Summary: In this lecture, we studied an important correctness property of concurrent objects that is called *Linearizability*. A concurrent object is a data structure that is designed to support operations in parallel by multiple threads. The key question answered by linearizability is what return values are permissible when multiple threads perform these operations in parallel, taking into account what we know about the expected return values from those operations when they are performed sequentially. As an example, we considered two threads, T_1 and T_2, performing *enq(x)* and *enq(y)* operations in parallel on a shared concurrent queue data structure, and considered what values can be returned by a *deq()* operation performed by T_2 after the call to *enq(y)*. From the viewpoint of linearizability, it is possible for the *deq()* operation to return item *x* or item *y*.

One way to look at the definition of linearizability is as though you are a lawyer attempting to "defend" a friend who implemented a concurrent data structure, and that all you need to do to prove that your friend is "not guilty" (did not write a buggy implementation) is to show one scenario in which all the operations return values that would be consistent with a sequential execution by identifying logical moments of time at which the operations can be claimed to have taken effect. Thus, if *deq()* returned item *x* or item *y* you can claim that either scenario is plausible because we can reasonably assume that *enq(x)* took effect before *enq(y)*, or vice versa. However, there is absolutely no plausible scenario in which the call to *deq()* can correctly return a code/exception to indicate that the queue is empty since at least *enq(y)* must have taken effect before the call to *deq()*. Thus, a goal for any implementation of a concurrent data structure is to ensure that all its executions are linearizable by using whatever combination of constructs (e.g., locks, isolated, actors, optimistic concurrency) is deemed appropriate to ensure correctness while giving the maximum performance.

Optional Reading:

1. Wikipedia article on the Linearizability

Mark as completed