coursera

1.4 Lecture Summary

1.4 Liveness and Progress Guarantees

Lecture Summary: In this lecture, we studied three ways in which a parallel program may enter a state in which it stops making forward progress. For sequential programs, an "infinite loop" is a common way for a program to stop making forward progress, but there are other ways to obtain an absence of progress in a parallel program. The first is *deadlock*, in which all threads are blocked indefinitely, thereby preventing any forward progress. The second is *livelock*, in which all threads repeatedly perform an interaction that prevents forward progress, e.g., an infinite "loop" of repeating lock acquire/release patterns. The third is *starvation*, in which at least one thread is prevented from making any forward progress.

The term "liveness" refers to a progress guarantee. The three progress guarantees that correspond to the absence of the conditions listed above are *deadlock freedom*, *livelock freedom*, and *starvation freedom*.

Optional Reading:

1. Deadlock example with synchronized methods in Java

2. Starvation and Livelock examples in Java

3. Wikipedia article on Deadlock and Livelock

Mark as completed