Concepts in Concurrent Computing

Sources:
William Stallings, “Operating Systems Internals and Design Principles,” 7th edition, Chp 6
https://en.wikipedia.org/wiki/Critical_section
https://en.wikipedia.org/wiki/Race_condition

Terminology

- Atomic Operation
- Race Condition
- Critical Section and Mutual Exclusion
- Deadlock
- Livelock
- Starvation

Atomic Operation

- A function or action implemented as a sequence of one or more instructions that appears to be indivisible
- No other thread can see an intermediate state or interrupt the operation

Race condition

- A condition where a program's correct behavior depends on a sequence of steps in two or more threads (or processes).
  - The threads access some shared resource
- The order of each thread's steps can lead to non-determinism
- Bugs related to race conditions are
  - difficult to confirm (evidence of issue doesn't always appear)
  - difficult to fix (the act of debugging can change the relative timing of threads)
- Examples
  - Lost Update Problem (see earlier slides)
  - Time-Of-Check Time-of-Use (TOCTOU)
    - https://cwe.mitre.org/data/definitions/367.html

Critical Section and Mutual Exclusion

- A section of program code that can lead to a race condition if executed concurrently by two or more threads
- A critical section cannot be executed by more than one thread at a time
- To prevent a race condition, the thread's access to a critical section must be synchronized to ensure mutual exclusion
- Mutual exclusion
  - Thread 1 cannot execute code in the critical section while Thread 2 is executing the critical section. And Thread 2 cannot execute the critical section while Thread 1 is doing so.

Deadlock

- Definition: the permanent blocking of a set of threads that either compete for system resources or communicate with each other.
- A set of threads is deadlocked when each thread is waiting for an event that can only be initiated by another blocked thread.
  - The ”event” is usually releasing a held resource
- Deadlock is permanent because none of the events ever occurs.
- Each thread in the set is waiting because a required resource is held exclusively by another process or thread.

Deadlock - Required Conditions

- Mutual Exclusion
  - Only one thread may use a resource at a time. No resource sharing.
- Hold and Wait
  - A thread may hold resources and request other resources. Resources do not have to be acquired at the same time.
- No preemption
  - No resource can be taken from a thread holding it. A thread must voluntarily release its resources.
- Circular wait
  - Given a closed chain of threads, each thread holds a resource needed by the next thread in the chain

Livelock

- A situation in which two or more threads continuously change their states in response to changes in the other threads.
- None of the threads perform any useful work.
- Non-computing Example
  - Two people walk towards each other in a narrow hall.
  - Both move to the same side of the hall to allow the other easy passage.
  - Both people step to the other side in recognition of what has happened.
    - Back and forth. Back and forth.
  - Neither person can make progress toward their goal even though their state is changing.

Starvation

- A condition in which a thread is indefinitely blocked because other processes are always given preference
- The blocked thread is able to proceed. It is not given a chance.