CS418— Operating Systems

Lecture 10

Process Management—Deadlock and Starvation

Textbook: Operating Systems
by William Stallings

1. Deadlock Concepts

- Deadlock — a simple staircase example.

- Deadlock — a classical case of deadlock.

2. Seven Examples of Deadlock in CS

- **Deadlocks on file requests**

- **Deadlocks in databases**

- **Deadlocks in device allocation**

- **Deadlocks in multiple device allocation**

- **Deadlocks in spooling**

- **Deadlocks in disk sharing**

- **Deadlocks in a network**

3. Conditions for Deadlock

- Mutual exclusion

- Resource holding

- No preemption

- Circular waiting

4. Modeling Deadlocks

- Directed Graph method (Holt,1972)

- A system is deadlocked iff there is a directed cycle.

- Detecting cycles in a dynamic graph is not easy.

5. Handling Deadlocks

- Prevention

- Avoidance (Banker's Algorithm)
  - 1. No customer will be granted a loan exceeding the bank's total capital.
  - 2. A customer will be given a maximum credit limit.
  - 3. No customer will be allowed to borrow over the limit.
  - 4. Sum of all loans $\leq$ bank's total capital.

- Detection (CES Algorithm, by Coffman, et al, 1971)
  - 1. Mark each process that has a row of 0's in the allocation matrix.
  - 2. $W \leftarrow$ available vector.
  - 3. Find i such that process i is currently unmarked and row$-i$ of Q is $\leq W$. If no such row exists, exit.
  - 4. If such a row is found, mark process i and $W \leftarrow W + A_i$, where A is the allocation matrix. Repeat Step 3.
  - **A deadlock exists if and only if there are unmarked processes at the end of the algorithm).**

- Recovery

6. Starvation

- The dining philosophers problem (Dijsktra, 1968)

7. Process Synchronization

- OS must make a resource unavailable to other processes while it is being used by one of them. Only when the resource is released is a waiting process allowed to use the resource. Process synchronization is critical here.
- The common element in all synchronization schemes is to allow a process to finish work on a **critical region** of the program before other processes have access to it.
- Synchronization is usually implemented as a *lock-and-key* arrangement: (1) the process must first see if the key is available and (2) if it is available, it must pick it up and put it in the lock to make it unavailable to other processes.
  - TEST-AND-SET (IBM 360/370): in a single CPU cycle it tests to see if the key is available and if it is, sets it to "unavailable".
  - WAIT-AND-SIGNAL: based on TEST-AND-SET, designed to remove busy waiting.
  - Semaphore: a nonnegative integer variable that's used as a flag.

8. Process Cooperation

- In real life, we have occasions when several processes work directly together to complete a common task. This is still the research topic of people in distributed computing.
  Example: *Several people try to edit a file over the Internet.*

- Producers and Consumers

- Readers and Writers