CMPSCI 377: Operating Systems

Discussion 7

Administrivia

- Lab 1 will be graded by the end of the week
- Lab 2 will be posted today
  - Due two weeks from Friday (November 2)
  - As always, start early

- Today's agenda
  - Deadlock review (no quiz)
  - Lab 2 overview

Deadlocks

- What is a **deadlock**?

- Multiple threads that are unable to proceed because they're all waiting for something

- Example: thread 1 owns resource A and is waiting for resource B, thread 2 owns B and is waiting for A
  - Neither can proceed; threads 1 and 2 are *deadlocked*

Deadlock Conditions

- Four necessary conditions for deadlock
- **Mutual Exclusion**
  - Thread holds a non-sharable resource
- **Hold and Wait**
  - Thread holds a resource and is waiting for another
- **No Preemption**
  - Can’t force a thread to release its resources
- **Circular Wait**
  - List of waiting threads is circular (e.g., A→B→C→A)

Preventing Deadlocks

- Ensure that one of the four conditions doesn't hold
  - Sharable resources (mutual exclusion)
  - Threads must acquire all resources at once (hold & wait)
  - OS can forcibly release resources (no preemption)
  - Request resources in a predetermined order (circular wait)

- Banker's Algorithm
  - Only allocate resources if doing so doesn't expose the system to possible deadlocks
  - Threads declare what resources they *might* need in advance

- How do most OSes actually prevent deadlocks?
  - They don't!
  - Leave it to the programmer

Lab 2 Overview

- Lab 2: CPU Scheduling
- Implement multilevel feedback queue scheduling (MLFQ) in Nachos
- Start with a round-robin scheduler; replace it with a 3-level MLFQ scheduler