Deadlocks

Chapter 7

Using 2 mutexes….

- Recall – if two threads are attempting to enter two different critical sections (A and B)
  - All is well if both ask for A, then B
  - All is well if both ask for B, then A
  - Things can be very bad if one asks for A, then B and the other the opposite order!

- If things break just right (or badly) – we could be deadlocked.

Deadlocks

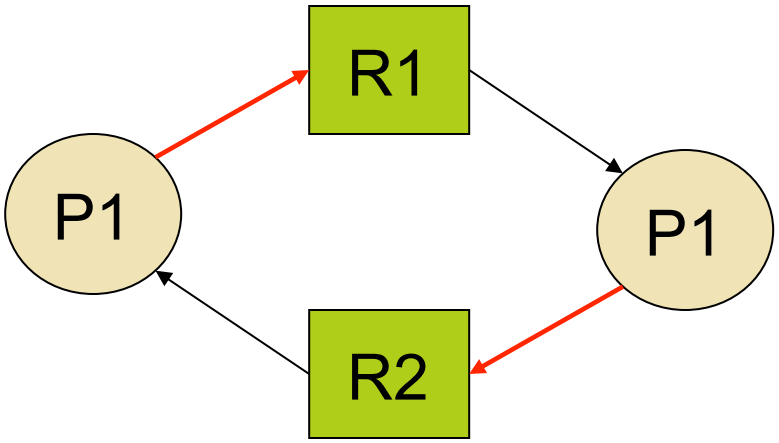
- Deadlocks are not just about “data” synchronization…
  - Deadlocks occur in resource allocation
    - DVD/CD
    - CPU
    - Disks
    - Printers
    - Mutex/Monitor

Deadlock Requirements

- Mutual Exclusion
- Hold & Wait
- No Preemption
- Circular Wait

Modeling Resource Allocation

- Can define deadlocks as cycles in a directed graph:
  - Resources & Process are Nodes/Vertices
  - Assignment Edges
  - Request Edges



Handling Deadlocks

- Eliminate Them
  - Prevention algorithms - *attack requirements*
  - Avoidance algorithms - *careful planning*
- Detection Algorithms - *recovery mechanisms*

Avoidance Algorithms

- System can be in either a *safe* or *unsafe* state with respect to deadlocks
  - safe implies no deadlocks
  - unsafe implies there *may* be a deadlock
- Resource Graph Solution
- Banker's Algorithm

Deadlock Detection & Recovery

- Use graphs to detect deadlocks
  - When do we run the algorithm?
  - Does it matter?

- Recovery:
  - Terminate processes?
    - who?
  - Preempt resource (take it away)?
    - who?
    - what?