Handling Deadlock

Jonathan Geisler

March 2, 2007

Taylor University
CSS

Conditions necessary for deadlock

1. Mutual exclusion
2. Hold resources & wait
3. No preemption of resources
4. Circular wait

Options for dealing with deadlock

1. Steer clear
   - Avoidance
   - Prevention
2. Detect
3. Ostrich
   - Do nothing
   - Most frequent

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?
2. Hold & Wait

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?
2. Hold & Wait
   - Ask for resources at process startup

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?
2. Hold & Wait
   - Ask for resources at process startup
   - Ask for resources when none are held

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?
2. Hold & Wait
   - Ask for resources at process startup
   - Ask for resources when none are held
3. No Preemption

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?
2. Hold & Wait
   - Ask for resources at process startup
   - Ask for resources when none are held
3. No Preemption
   - Release on wait

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?
2. Hold & Wait
   - Ask for resources at process startup
   - Ask for resources when none are held
3. No Preemption
   - Release on wait
   - Take resources from waiting process

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?
2. Hold & Wait
   - Ask for resources at process startup
   - Ask for resources when none are held
3. No Preemption
   - Release on wait
   - Take resources from waiting process
4. Circular Wait

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?
2. Hold & Wait
   - Ask for resources at process startup
   - Ask for resources when none are held
3. No Preemption
   - Release on wait
   - Take resources from waiting process
4. Circular Wait
   - Acquire resources in specified order

Prevention

This technique works by making one of the necessary conditions impossible to reach.

1. Can we remove Mutual Exclusion?
2. Hold & Wait
   - Ask for resources at process startup
   - Ask for resources when none are held
3. No Preemption
   - Release on wait
   - Take resources from waiting process
4. Circular Wait
   - Acquire resources in specified order
   - FreeBSD provides checker (witness) to help debug this condition

Avoidance

This technique works by ensuring the processes always remain in a "safe state." It may be possible that the four necessary conditions are held, but they will never cause a problem because we have ensured that we will never reach a deadlocked state.

1. Resource allocation graph
   - Add claim edges
   - Cycle detection whenever a process requests a change from claim to use
2. Bankers algorithm
   - Pretend allocation is successful
   - Find if still in a safe state

Detection

- Periodically check for cycles in wait-for graph
- Modified bankers algorithm
- When should we check?

Recovery

1. Process termination
   - Kill one (which one?)
   - Kill all
2. Resource preemption
   - Which resources?
   - Which process?
   - How do we recover?
   - Can we prevent starvation?