Deadlocks

Readings: Sections 6.3, 6.4

Deadlock approaches

Deadlock prevention
- Disallow one of the three necessary conditions for deadlock occurrence
- Or, prevent circular wait condition from happening

Deadlock avoidance
- Do not grant a resource request if this allocation might lead to deadlock
- And, periodically check for the presence of deadlock and take action to recover

Deadlock detection
- Grant resource requests when possible

Deadlock Prevention

- Design a system in such a way that the possibility of deadlock is excluded
- Two main methods:
  - Indirect
    - Prevent the occurrence of one of the three necessary conditions
  - Direct
    - Prevent the occurrence of a circular wait

Deadlock Avoidance

- Allows the three necessary conditions but makes judicious choices to assure that the deadlock point is never reached
- A decision is made dynamically whether the current resource allocation request will, if granted, potentially lead to a deadlock
- Requires knowledge of future process requests
**Approaches for deadlock avoidance**

**Deadlock Avoidance**

**Process Initiation Denial**
Do not start a process if its demands might lead to deadlock.

**Resource Allocation Denial**
Do not grant an incremental resource request to a process if this allocation might lead to deadlock.

---

**Resource Allocation Denial**

- Referred to as the banker's algorithm
- State of the system reflects the current allocation of resources to processes
- Safe state is one in which there is at least one sequence of resource allocations to processes that does not result in a deadlock
- Unsafe state is a state that is not safe

---

**Figure 6.7 Determination of a Safe State**

(a) Initial state

(b) P2 runs to completion
Figure 6.7 Determination of a Safe State

(c) P1 runs to completion

Figure 6.8 Determination of an Unsafe State

(d) P3 runs to completion
Deadlock Avoidance Restrictions

- Maximum resource requirement for each process must be stated in advance
- Processes under consideration must be independent and with no synchronization requirements
- There must be a fixed number of resources to allocate
- No process may exit while holding resources

Advantages:

- It leads to early detection
- The algorithm is relatively simple

Disadvantage

- Frequent checks consume considerable processor time

Deadline Detection Algorithm

- A check for deadlock can be made as frequently as each resource request or, less frequently, depending on how likely it is for a deadlock to occur

Recovery Strategies

- Abort all deadlocked processes
- Back up each deadlocked process to some previously defined checkpoint and restart all processes
- Successively abort deadlocked processes until deadlock no longer exists
- Successively preempt resources until deadlock no longer exists