4.3 Deadlock - System Models, Necessary Conditions leading to Deadlocks, Deadlock Handling - Preventions, avoidance.

4.3 Deadlock

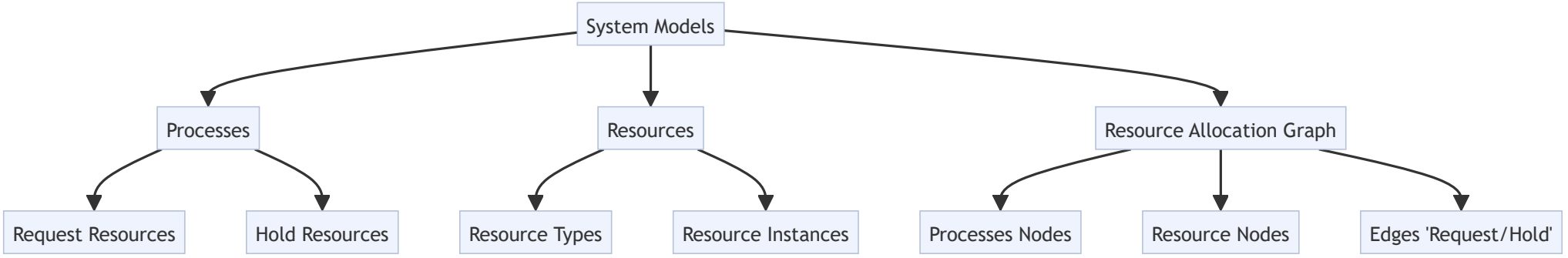
Introduction to Deadlock

Working, Simple Usage, Explanation:
Deadlock occurs in a multiprogramming environment when two or more processes are unable to proceed because each is waiting for the other to release resources. It effectively halts the execution of processes and can cause system freeze if not managed properly.

1. System Models

Explanation:
System models describe how resources and processes interact in a system. In the context of deadlocks, the system model includes the allocation and request of resources among processes.

Diagram



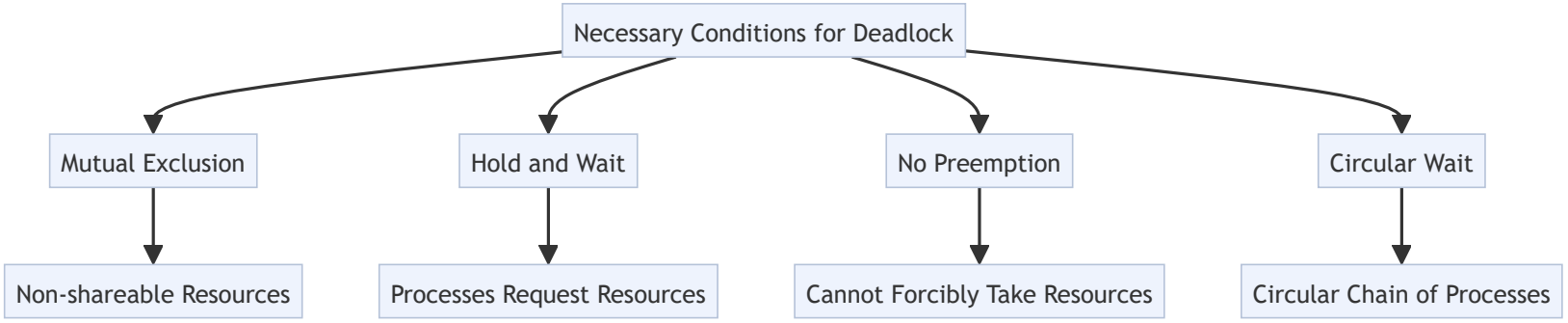
2. Necessary Conditions Leading to Deadlocks

Explanation:
Deadlock can only occur if all of the following conditions are true:

1. **Mutual Exclusion:** At least one resource is held in a non-shareable mode.

2. **Hold and Wait:** A process holding resources can request additional resources without releasing its current resources.
3. **No Preemption:** Resources cannot be forcibly taken from processes.
4. **Circular Wait:** A circular chain of processes exists where each process holds a resource needed by the next process in the chain.

Diagram



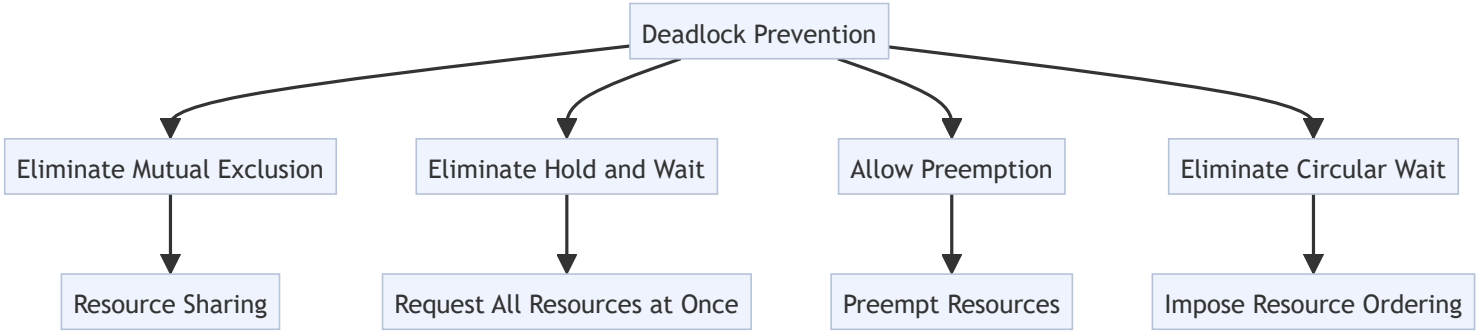
3. Deadlock Handling

3.1 Prevention

Explanation:
Deadlock prevention involves modifying the system's operating rules to ensure that at least one of the necessary conditions cannot hold. Methods include:

- **Eliminating Mutual Exclusion:** Use resource sharing if possible.
- **Eliminating Hold and Wait:** Require processes to request all needed resources at once.
- **Allowing Preemption:** Preempt resources from processes if needed.
- **Eliminating Circular Wait:** Impose a total ordering on resources and require processes to request resources in a specific order.

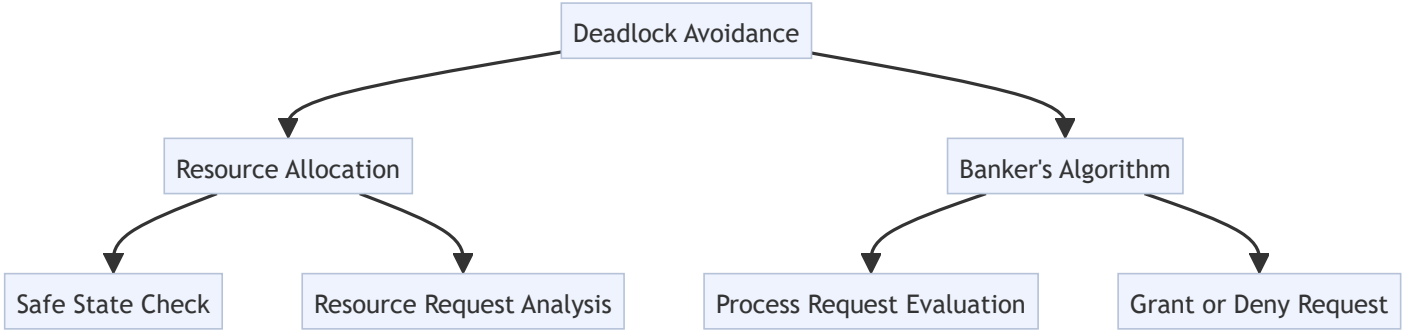
Diagram



3.2 Avoidance

Explanation:
Deadlock avoidance involves ensuring that a system never enters a deadlock state. This is typically done using resource allocation strategies like the Banker's Algorithm, which checks if resource requests can be safely granted without leading to a deadlock.

Diagram



Summary Table

Aspect	Details
System Models	Models interaction between processes and resources. Includes Resource Allocation Graph.
Necessary Conditions	1. Mutual Exclusion 2. Hold and Wait 3. No Preemption 4. Circular Wait

Aspect	Details
Deadlock Prevention	1. Eliminate Mutual Exclusion 2. Eliminate Hold and Wait 3. Allow Preemption 4. Eliminate Circular Wait
Deadlock Avoidance	Use techniques like the Banker's Algorithm to avoid unsafe states and ensure resources are allocated safely.