



**MD
RUF SANJANI
SHANTO**
221-35-1064



**MD MAHFUZ
RAHMAN
NIROB**
221-35-981



**PIJUS
CHANDRO
ROY**
221-35-1050



**SASONJOY
CHANDRA
COACH**
221-35-841



**AHIDUL
ISLAM**
221-35-931

Team Everyone



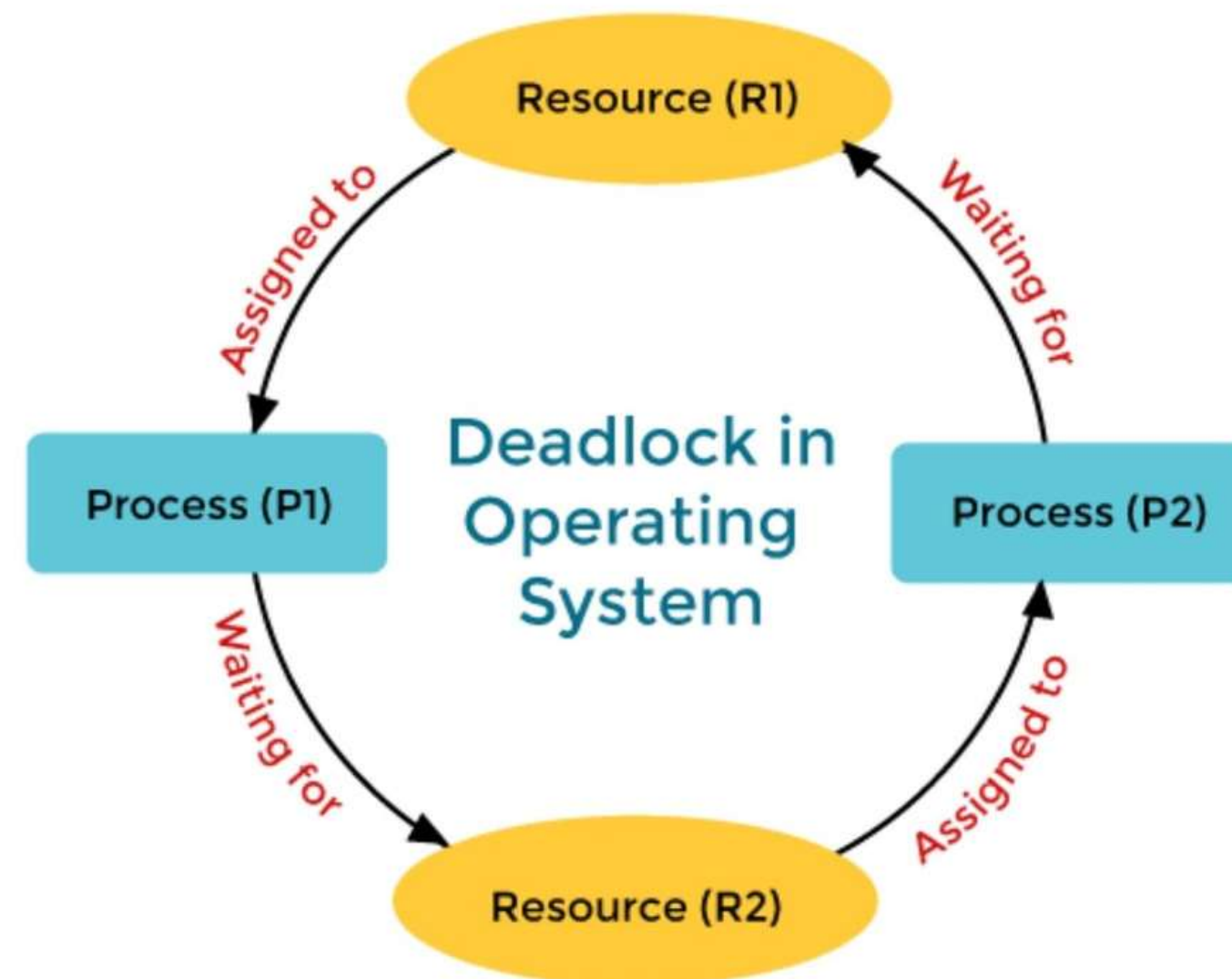


What is the Banker's Algorithm?

- A resource allocation and deadlock avoidance algorithm for operating systems.
- Ensures safe allocation of resources to processes to prevent deadlocks.

DEADLOCK?


- A situation where two or more processes are blocked forever, waiting for each other to release resources needed to proceed.





BANKER'S ALGORITHM COMPONENTS



Key Components:

- **Resource Types**
 - **Processes**
 - **Data Structures :**
 - Max Need Matrix
 - Allocated Matrix
 - Available Matrix
- 



SAFETY ALGORITHM

How the Algorithm Works:

- **Simulate resource allocation:** Allocate resources to processes based on their requests and the available resources.
 - **Finish processes:** Check if any process can finish execution with its remaining resources.
 - **Release resources:** If a process finishes, release its allocated resources and update the Available Matrix.
 - **Repeat until all processes finish:** Continue simulating resource allocation, finishing processes, and releasing resources until all processes finish or a deadlock is detected.
- 
- 



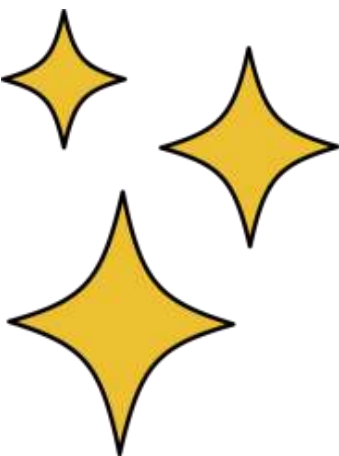
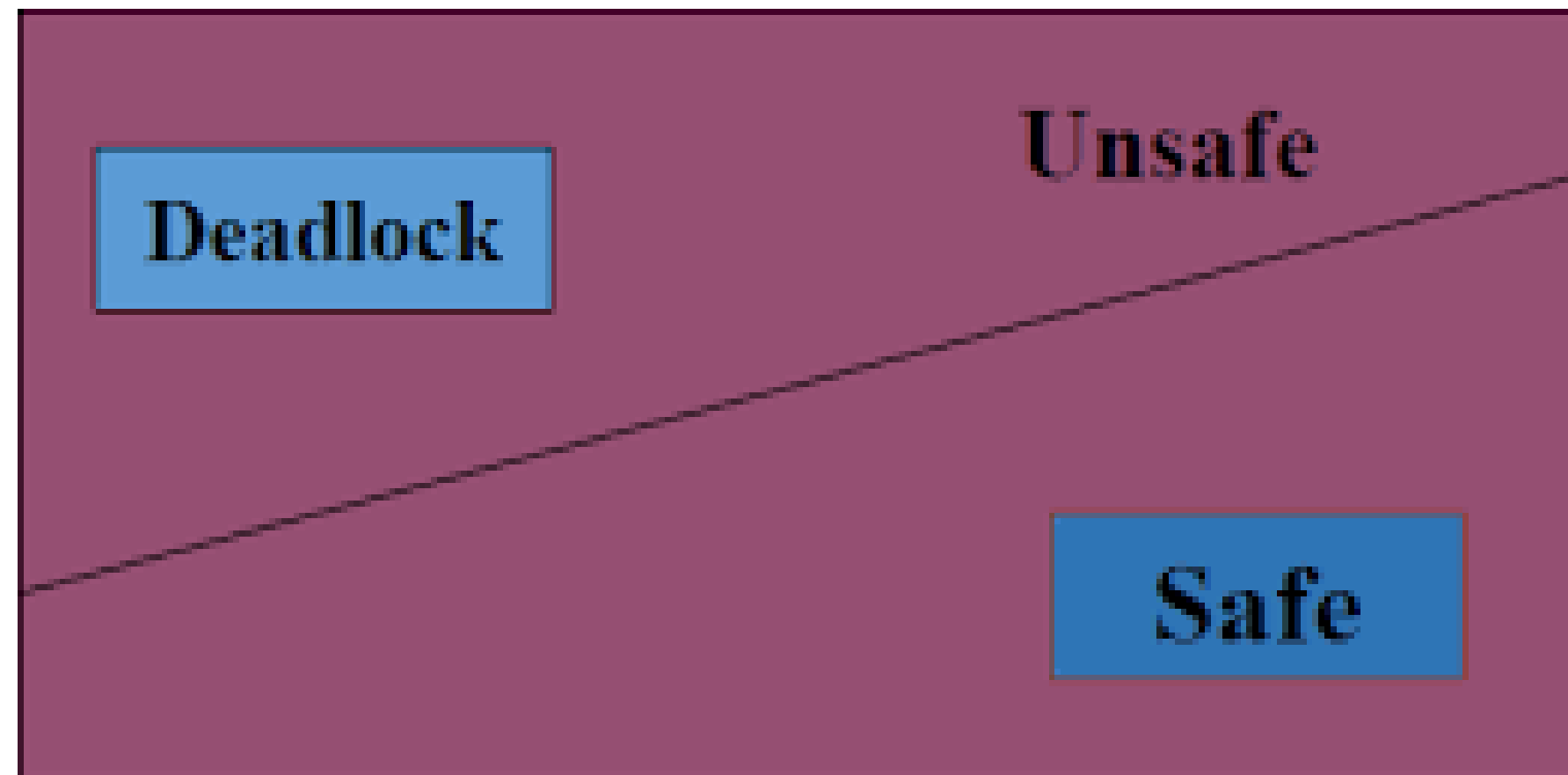
SAFE STATE AND DEADLOCK DETECTION

Safe State:

- A system state where the Safety Algorithm can finish all processes eventually.
- Indicates that granting resource requests will not lead to a deadlock.

Deadlock Detection:

- If the Safety Algorithm fails to finish any process in its simulation, it indicates a deadlock.
- The system must take corrective actions to avoid deadlock, such as resource preemption or process termination.



EXAMPLE

Total A= 10, B=5, C=7;

Processes	Allocation			Max Need			Available			Remaining Need/Requested (max need – allocation)		
	A	B	C	A	B	C	A	B	C	A	B	C
P1	0	1	0	7	5	3	3	3	2	7	4	3
P2	2	0	0	3	2	2	5	3	2	1	2	2
P3	3	0	2	9	0	2	7	4	3	6	0	0
P4	2	1	1	4	2	2	7	4	5	2	1	1
P5	0	0	2	5	3	3	7	5	5	5	3	1
	7	2	5				10	5	7			

Remaining need \leq Work/Available

For P1 : 7 4 3 \leq 3 3 2 ; False

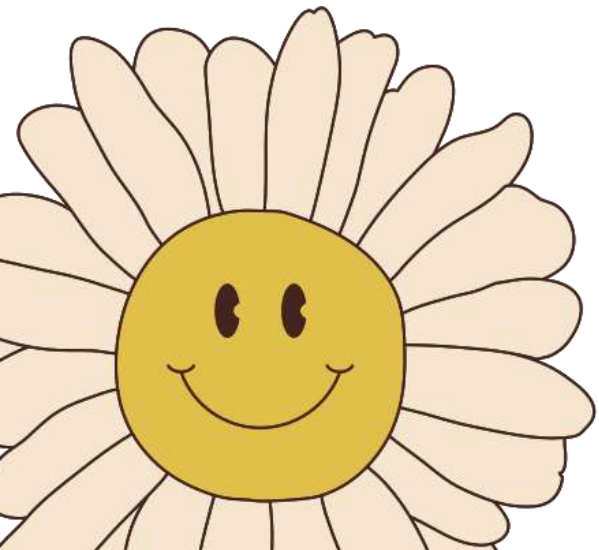
For P2: 1 2 2 \leq 3 3 2; True

So P2 enters the Safe state and the new Available Value will be

$$\begin{aligned}\text{Work/ Available} &= \text{Work (3 3 2)} + \text{Allocation (2 0 0)} \\ &= 5 \ 3 \ 2\end{aligned}$$

So similarly if we update the value and can fulfill the requests of all processes then the process sequence will be:

P2 P4 P5 P1 P3 Safe Sequence



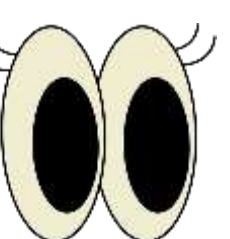



ADVANTAGES AND DISADVANTAGES

Advantages:

- Effective deadlock avoidance: Can prevent deadlocks from occurring in the first place.
- Improved resource utilization: Allocates resources efficiently by considering maximum needs.
- Fairness: Ensures fair access to resources for all processes.

Disadvantages:

- Overhead: Requires maintaining and updating data structures, increasing system overhead.
 - Limited applicability: Works best for systems with static resource requests and predictable execution times.
 - Complexity: Implementation can be complex, especially for large systems with many resources and processes.
- 
- 



THANK YOU

Resources:

<https://www.codetextpro.com>

<https://www.geeksforgeeks.org>

<https://chat.openai.com>