

INSTITUTO POLITECNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

UNIDAD 5

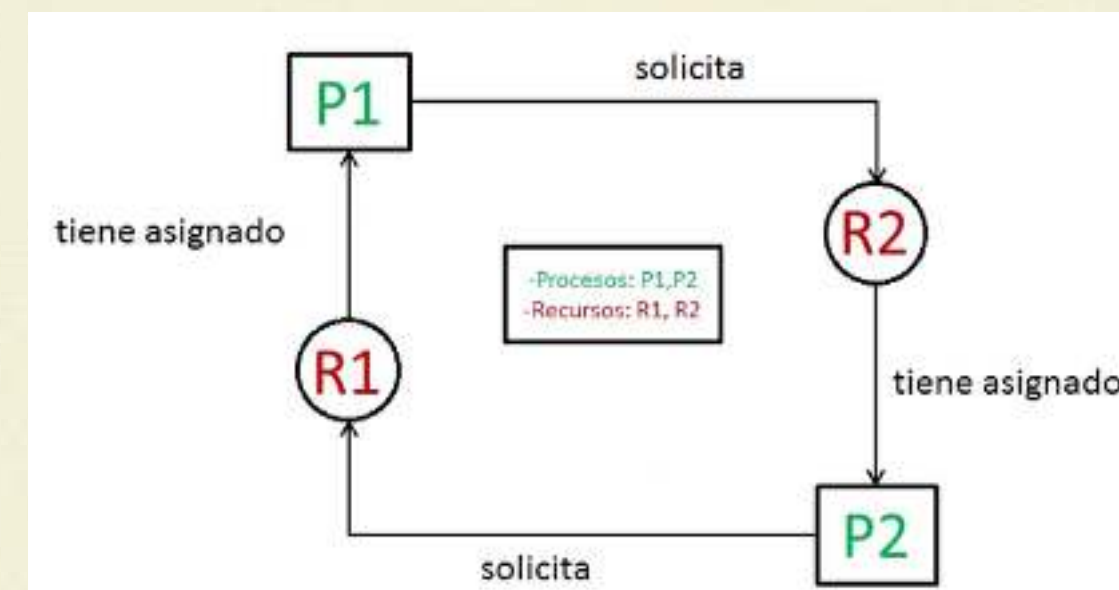
5.3 Capas de software y bloqueo mutuo



Interbloqueos

Definición:

- Situación en la que varios procesos compiten por un número finito de recursos y quedan bloqueados esperando que otros recursos se liberen.





Interbloqueos

Ejemplo :

- Analogía de los trenes que se detienen en un cruce sin poder avanzar hasta que el otro tren haya salido.

Importancia:

- Los interbloqueos impiden que los procesos concurrentes completen sus tareas.

Objetivos del Capítulo:

- Describir los interbloqueos.
- Presentar métodos para prevenir y evitar interbloqueos.



Modelo de Sistema

Recursos del Sistema:

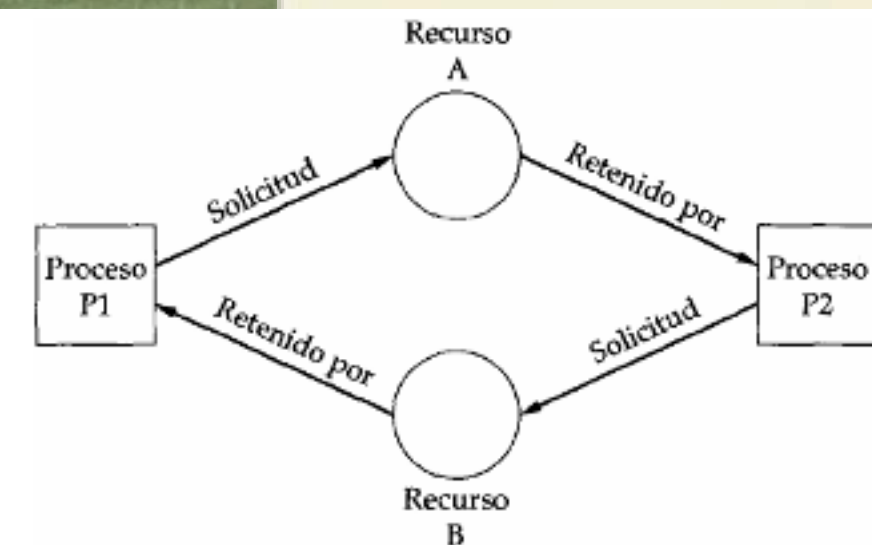
- Los recursos en un sistema informático son variados e incluyen espacio de memoria, ciclos de CPU, dispositivos de entrada/salida (E/S) como impresoras y unidades de DVD. Cada tipo de recurso puede tener múltiples instancias disponibles.
- Por ejemplo, en un sistema con dos impresoras, ambas impresoras pertenecen a la misma clase de recurso, pero pueden considerarse como instancias separadas.



Modelo de Sistema

Asignación de Recursos:

- **Solicitud:** Un proceso solicita una instancia de un recurso. Si el recurso solicitado no está disponible inmediatamente, el proceso debe esperar hasta que el recurso se libere.
- **Uso:** Una vez que el recurso es asignado, el proceso puede operar sobre él (por ejemplo, si el recurso es una impresora, el proceso puede imprimir en ella).
- **Liberación:** Tras completar su tarea, el proceso libera el recurso para que otros procesos puedan utilizarlo.



Modelo de Sistema

Llamadas al Sistema:

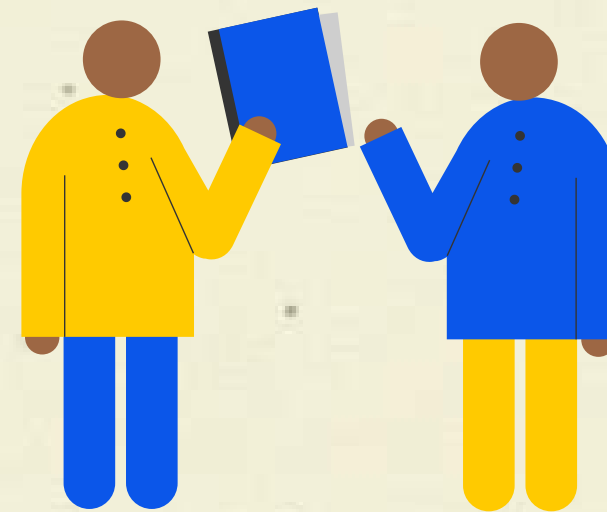
- Las solicitudes y liberaciones de recursos se gestionan mediante llamadas al sistema. Ejemplos incluyen:
 - request(), release(): Para solicitar y liberar dispositivos.
 - open(), close(): Para abrir y cerrar archivos.
 - allocate(), free(): Para asignar y liberar memoria.



Modelo de Sistema

Tablas del Sistema:



- El sistema mantiene tablas para registrar el estado de cada recurso (libre o asignado) y para llevar un seguimiento de los procesos que tienen asignados recursos.





Caracterización de los Interbloqueos


Definición:

- 
- Un interbloqueo es una situación en la que un conjunto de procesos está bloqueado porque cada proceso está esperando un recurso que está retenido por otro proceso en el mismo conjunto. Esto crea un ciclo de dependencia circular donde ningún proceso puede avanzar.
- 

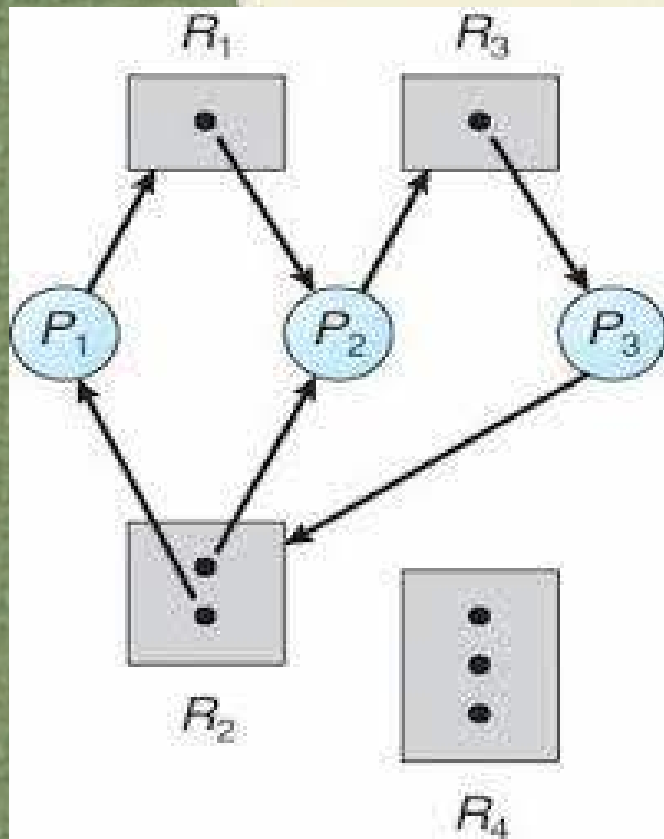


Caracterización de los Interbloqueos

Consecuencias:

- En un interbloqueo, los procesos nunca terminan de ejecutarse, lo que lleva a la paralización de esos procesos y al uso ineficiente de los recursos del sistema.
 - Esto puede causar una disminución en el rendimiento del sistema y, en casos extremos, una paralización total del sistema.
- 

Condiciones Necesarias para un Interbloqueo



Exclusión Mutua:

- Al menos un recurso debe estar en modo no compartido, es decir, solo un proceso puede usarlo a la vez.

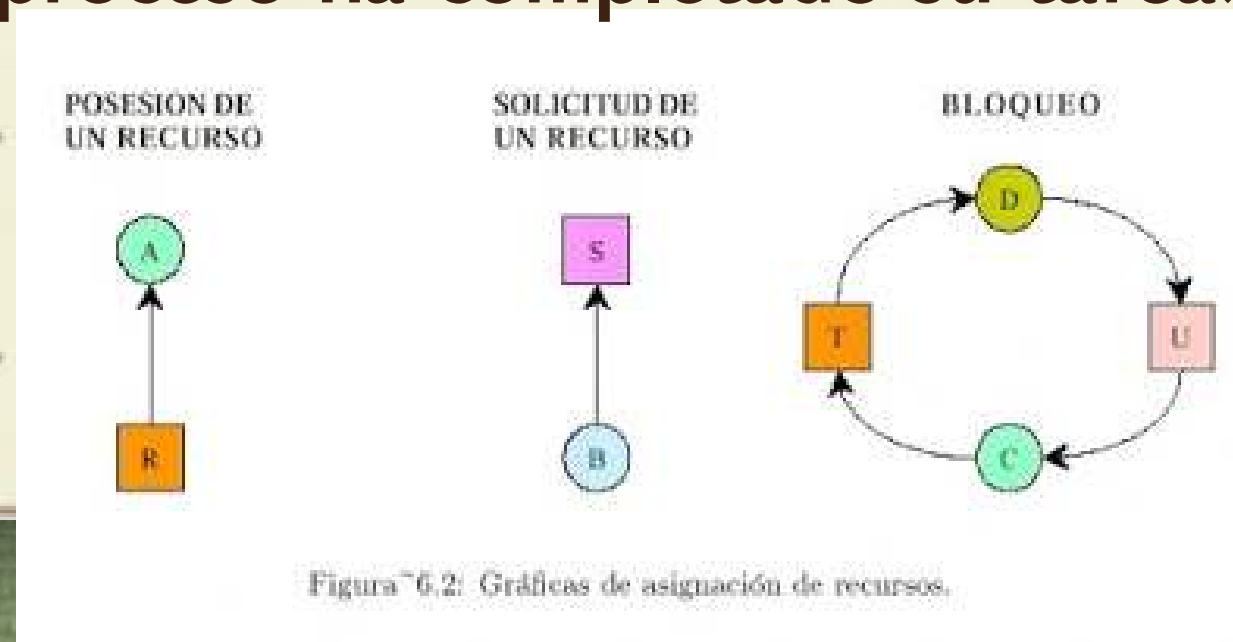
Retención y Espera:

- Un proceso debe retener al menos un recurso mientras espera adquirir recursos adicionales que están siendo retenidos por otros procesos.

Condiciones Necesarias para un Interbloqueo

Sin Desalojo:

- Los recursos no pueden ser desalojados forzosamente de los procesos que los están utilizando. Los recursos solo pueden ser liberados voluntariamente por el proceso que los está reteniendo, una vez que este proceso ha completado su tarea.



Condiciones Necesarias para un Interbloqueo

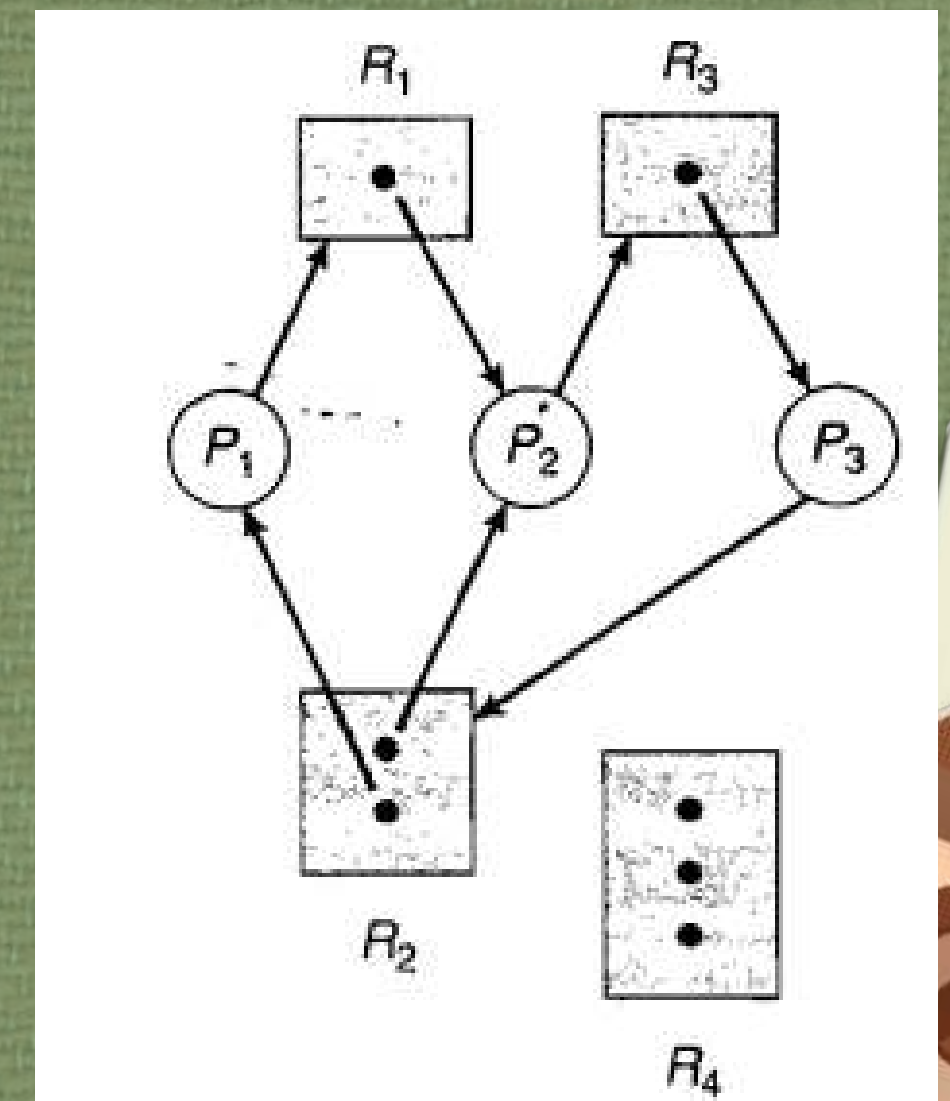
Espera Circular:

- Debe existir un conjunto de procesos $\{P_0, P_1, \dots, P_n\}$ tal que cada proceso P_i esté esperando un recurso que es retenido por el siguiente proceso P_{i+1} , y el último proceso P_n está esperando un recurso retenido por el primer proceso P_0 . Esta condición crea un ciclo de espera circular.

Grafo de Asignación de Recursos

Definición:

- Un grafo de asignación de recursos es una representación gráfica que muestra las relaciones entre los procesos y los recursos del sistema. Este grafo ayuda a visualizar cómo los recursos están asignados y cuáles son solicitados por los procesos.






Grafo de Asignación de Recursos



Elementos del Grafo:

- **Vértices:** Los vértices del grafo representan procesos (P_i) y recursos (R_j).
 - **Aristas de Solicitud:** Una arista dirigida desde un proceso P_i a un recurso R_j indica que el proceso P_i ha solicitado el recurso R_j y está esperando por él.
 - **Aristas de Asignación:** Una arista dirigida desde un recurso R_j a un proceso P_i indica que el recurso R_j ha sido asignado al proceso P_i .
- 




Grafo de Asignación de Recursos



Ejemplos:

- Grafo sin Ciclo: Si el grafo no contiene ningún ciclo, entonces ningún proceso del sistema está en estado de interbloqueo.
- Grafo con Ciclo: Si el grafo contiene un ciclo, entonces puede existir un interbloqueo. Cada proceso en el ciclo está esperando un recurso retenido por otro proceso en el ciclo.

Interpretación:

- La presencia de un ciclo en el grafo de asignación de recursos es una condición necesaria y suficiente para la existencia de un interbloqueo. Si hay un ciclo, hay interbloqueo. Si no hay ciclo, no hay interbloqueo.
- 

Métodos para Tratar los Interbloqueos

Prevenir Interbloqueos:

- Se puede emplear un protocolo para asegurar que el sistema nunca entre en estado de interbloqueo. Esto se logra mediante la prevención de una o más de las condiciones necesarias para que ocurra un interbloqueo.





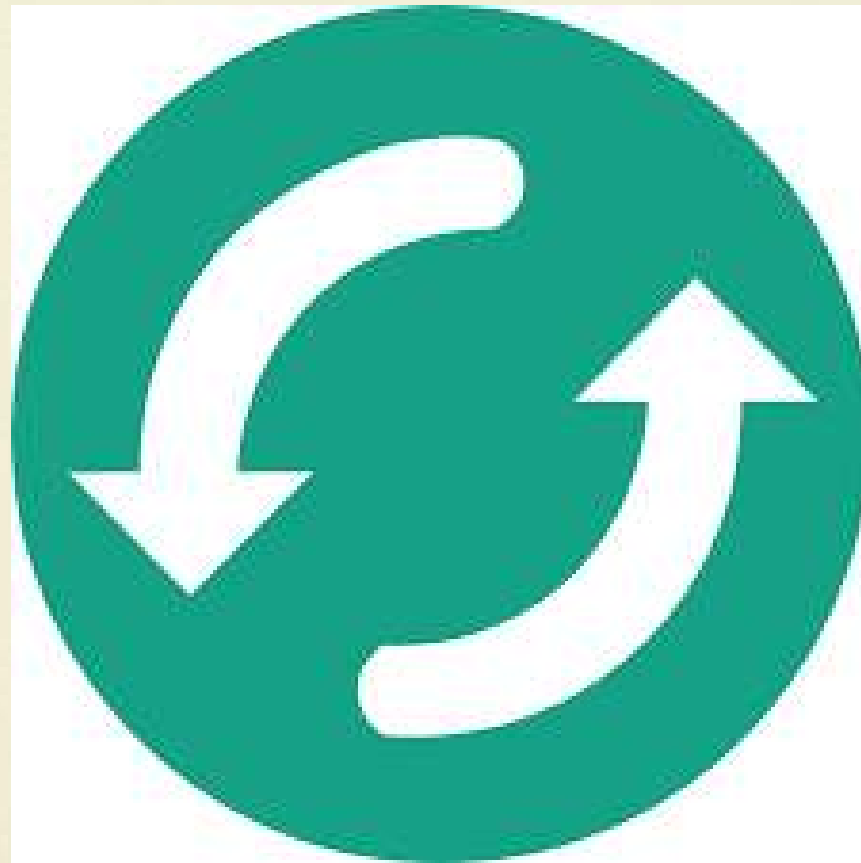
Métodos para Tratar los Interbloqueos

Ejemplos:

- **Política de Asignación Ordenada:** Los procesos deben solicitar recursos en un orden predefinido, evitando la espera circular.
- **Restricciones en Solicitudes:** Limitar las solicitudes de recursos para que los procesos no puedan retener recursos mientras esperan por otros.

Métodos para Tratar los Interbloqueos

Detectar y Recuperar:



- Permitir que el sistema entre en estado de interbloqueo, pero implementar mecanismos para detectarlo y recuperarse de él.
- Detección:
 - Algoritmos que examinan el estado del sistema para identificar ciclos en el grafo de asignación de recursos.
- Recuperación:
 - Abortar Procesos: Finalizar uno o más procesos para romper el ciclo de interbloqueo.
 - Liberar Recursos: Forzar la liberación de recursos retenidos por procesos.

Métodos para Tratar los Interbloqueos

Ignorar el Problema:



- Actuar como si los interbloqueos nunca se produjeran. Este método es común en muchos sistemas operativos, como Unix y Windows, donde la responsabilidad de evitar interbloqueos recae en los desarrolladores de aplicaciones.

Ejemplo:

- Reiniciar Manualmente: En casos extremos, los administradores de sistemas pueden tener que reiniciar manualmente los procesos o el sistema para resolver un interbloqueo.



Métodos para Tratar los Interbloqueos

Prevención de Interbloqueos:

- Métodos y algoritmos específicos diseñados para garantizar que al menos una de las condiciones necesarias para el interbloqueo no se cumpla.
- Ejemplos:
 - Protocolos de Adquisición y Liberación: Diseñar el sistema de manera que los procesos soliciten y liberen recursos de manera que no puedan formar ciclos.



Métodos para Tratar los Interbloqueos

Evasión de Interbloqueos:

- Requiere información adicional sobre los recursos que los procesos necesitarán y el tiempo de vida de los procesos.
- Algoritmo del Banco: Se asegura de que los recursos estén disponibles de manera que se pueda satisfacer una solicitud sin entrar en estado de interbloqueo.



Métodos para Tratar los Interbloqueos

Detección y Recuperación:

- **Algoritmos de Detección:**
 - Revisan periódicamente el estado del sistema para detectar ciclos en el grafo de asignación de recursos.
- **Recuperación:**
 - Estrategias para resolver interbloqueos una vez detectados, como abortar procesos o liberar recursos forzosamente.



Prevención de Interbloqueos, Exclusión Mutua, Retención y Espera

- **Interbloqueo (Deadlock)**

Es una situación en la que dos o más procesos no pueden avanzar porque cada uno está esperando que el otro libere un recurso.

- **Exclusión Mutua**



Se refiere a la condición de que solo un proceso puede utilizar un recurso crítico a la vez. Esto es fundamental para evitar que los procesos interfieran entre sí.

Prevención de Interbloqueos, Exclusión Mutua, Retención y Espera

- **Retención y Espera**

La retención y espera es una condición donde un proceso que ya tiene al menos un recurso, espera adicionalmente por otros recursos que están siendo retenidos por otros procesos.





Para prevenir interbloqueos, se deben evitar al menos una de las siguientes condiciones:

- **Exclusión Mutua:** Aunque es difícil eliminar, se puede minimizar permitiendo el acceso compartido cuando sea posible.
- **Retención y Espera:** Se pueden requerir procesos a solicitar todos los recursos de una vez, en lugar de uno a la vez.
- **No Desalojo:** Permitir que un proceso desaloje recursos si no puede obtener todos los que necesita.
- **Esperar Circular:** Imponer un orden lineal de recursos y requerir que cada proceso solicite recursos en este orden.





Sin Desalojo, Espera Circular

- **Sin Desalojo (No Preemption)**

No desalojo significa que un recurso asignado a un proceso no puede ser forzado a liberarse hasta que el proceso complete su tarea con ese recurso. Para prevenir interbloqueos, se puede permitir que los procesos desalojen recursos cuando no pueden obtener todos los recursos que necesitan.

Sin Desalojo, Espera Circular

- **Espera Circular (Circular Wait)**

La espera circular ocurre cuando existe un conjunto de procesos $\{P_1, P_2, \dots, P_n\}$ donde P_1 está esperando un recurso que posee P_2 , P_2 está esperando un recurso que posee P_3 , y así sucesivamente, hasta que P_n está esperando un recurso que posee P_1 . La prevención de la espera circular implica imponer un orden global en los recursos y requerir que los procesos soliciten recursos en un orden ascendente.



Evasión de Interbloqueos

Implica tomar decisiones a priori sobre la solicitud de recursos para asegurar que el sistema nunca entre en un estado de interbloqueo. Esto se logra mediante el uso de algoritmos como el del banquero, que evaluará si conceder una solicitud de recursos puede llevar a un estado inseguro.



Estado Segura

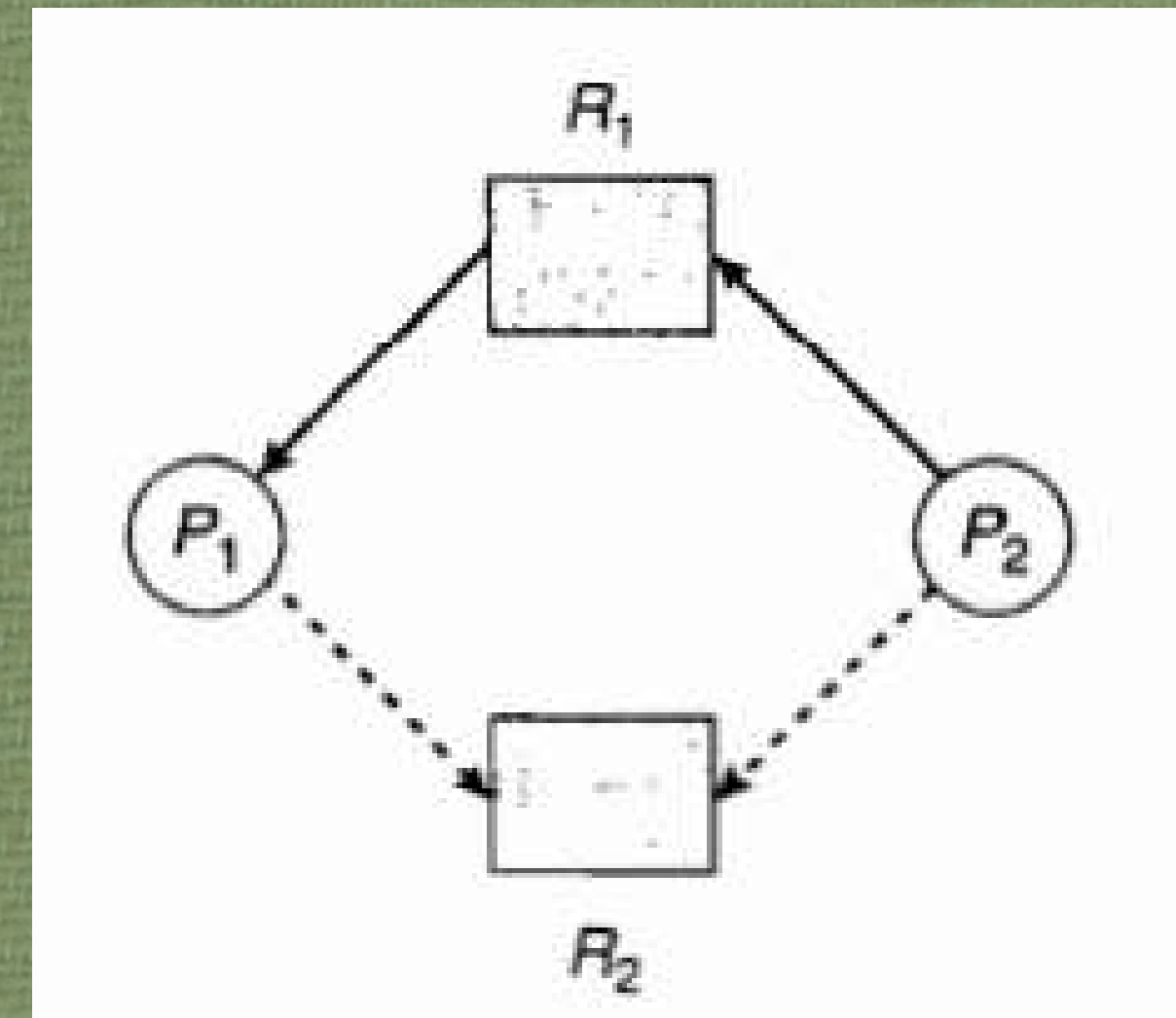


Es aquel en el que el sistema puede asignar recursos a cada proceso de tal manera que todos los procesos puedan completar su ejecución sin causar interbloqueos. En un estado seguro, existe una secuencia segura de procesos que permite que cada proceso finalice su ejecución eventualmente.



Grafo de Asignación de Recursos


Es una herramienta visual utilizada para representar la asignación de recursos en un sistema. Los nodos representan procesos y recursos, y los arcos indican la asignación y solicitud de recursos. Un ciclo en este grafo puede indicar un posible interbloqueo.





Algoritmo del Banquero



Propuesto por Edsger Dijkstra, es un método para evitar interbloqueos. Este algoritmo simula la asignación de recursos y evalúa si la nueva asignación llevará a un estado seguro. Si la asignación resulta en un estado inseguro, se deniega la solicitud de recursos.






Funcionamiento del Algoritmo del Banquero

- **Solicitud de Recursos:** Un proceso solicita recursos.
- **Simulación:** El sistema simula la asignación de estos recursos al proceso.
- **Evaluación del Estado:** Se evalúa si el sistema permanece en un estado seguro después de la asignación.
- **Decisión:** Si el estado es seguro, los recursos se asignan; de lo contrario, la solicitud se deniega.




Algoritmo de Seguridad, Algoritmo de Solicitud de Recursos

- Inicialización: Se crean dos vectores, Work (copia del vector Available) y Finish (inicializados a false).
 - Búsqueda de Proceso: Se busca un proceso i que aún no ha finalizado ($\text{Finish}[i] == \text{false}$) y cuyas necesidades ($\text{Need}[i]$) pueden ser satisfechas con los recursos disponibles (Work).
 - Asignación de Recursos: Si se encuentra tal proceso, se asignan los recursos a ese proceso actualizando Work y se marca el proceso como completado ($\text{Finish}[i] = \text{true}$). Se repite el paso 2.
 - Verificación de Seguridad: Si todos los procesos han sido marcados como completados ($\text{Finish}[i] == \text{true}$ para todos los i), entonces el sistema está en un estado seguro.
- 



Algoritmo de Solicitud de Recursos

- Verificación de Necesidades: Se verifica si la solicitud de recursos del proceso P_i no excede sus necesidades máximas ($Request[i] \leq Need[i]$). Si no, se genera un error.
 - Disponibilidad de Recursos: Se verifica si los recursos solicitados están disponibles ($Request[i] \leq Available$). Si no, el proceso debe esperar.
 - Asignación Simulada: Si los recursos están disponibles, se asignan provisionalmente los recursos al proceso y se actualizan las matrices $Available$, $Allocation$ y $Need$.
 - Verificación de Seguridad: Se ejecuta el algoritmo de seguridad para comprobar si el nuevo estado es seguro. Si es seguro, la asignación se completa; de lo contrario, el proceso debe esperar.
- 

Un Ejemplo Ilustrativo

Consideremos un sistema con cinco procesos (P_0 a P_4) y tres tipos de recursos (A, B y C). Usando matrices Allocation, Max, y Available, se verifica si una solicitud de recursos puede ser concedida sin poner el sistema en riesgo de interbloqueo. La matriz Need se calcula como $\text{Max} - \text{Allocation}$. Luego, se utiliza el algoritmo de seguridad para verificar si el nuevo estado es seguro tras una solicitud de recursos.

	<u>Allocation</u>	<u>Max</u>	<u>Available</u>
	A B C	A B C	A B C
P_0	0 1 0	7 5 3	3 3 2
P_1	2 0 0	3 2 2	
P_2	3 0 2	9 0 2	
P_3	2 1 1	2 2 2	
P_4	0 0 2	4 3 3	

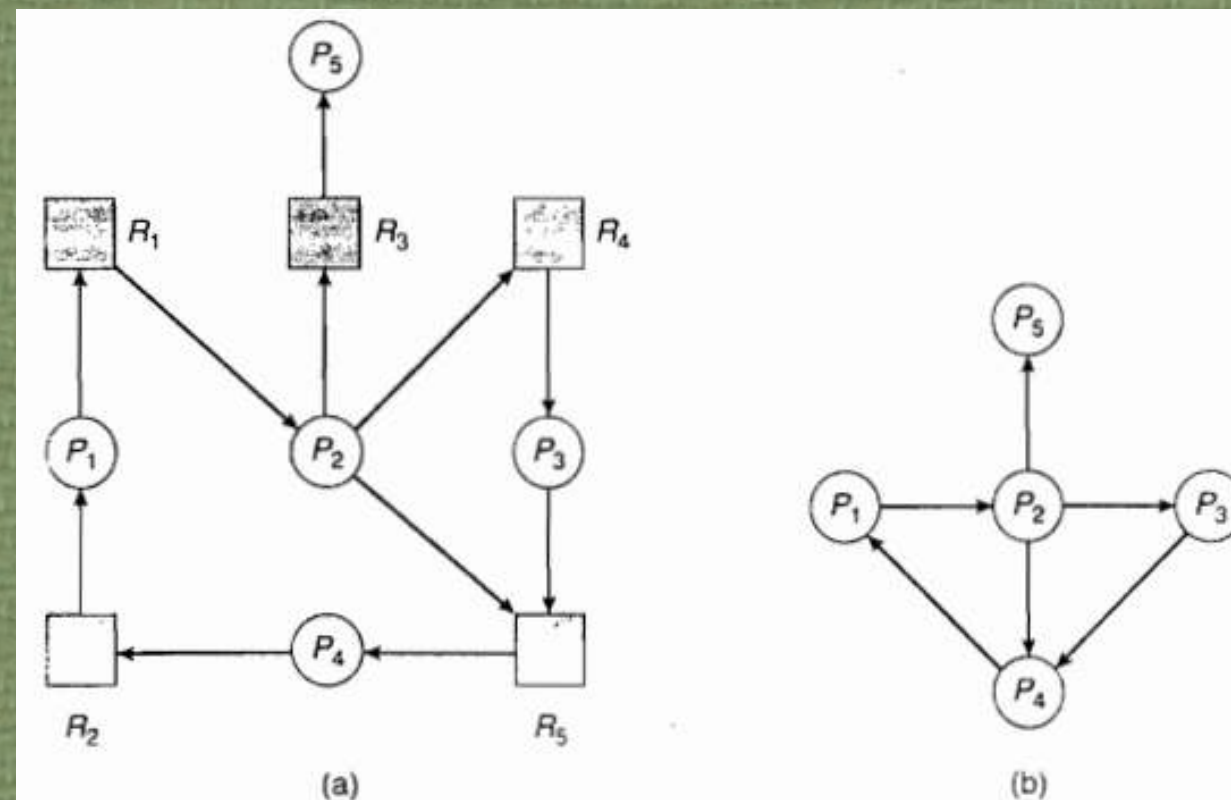
	<u>Need</u>
	A B C
P_0	7 4 3
P_1	1 2 2
P_2	6 0 0
P_3	0 1 1
P_4	4 3 1

	<u>Allocation</u>	<u>Need</u>	<u>Available</u>
	A B C	A B C	A B C
P_0	0 1 0	7 4 3	2 3 0
P_1	3 0 2	0 2 0	
P_2	3 0 2	6 0 0	
P_3	2 1 1	0 1 1	
P_4	0 0 2	4 3 1	

Detección de Interbloqueos, Una Sola Instancia de Cada Tipo de Recurso

Detección de Interbloqueos con una Sola Instancia de Cada Tipo de Recurso:



- Utiliza un grafo de espera donde una arista $P_i \rightarrow P_j$ indica que el proceso P_i está esperando un recurso que posee P_j .
- Un interbloqueo existe si y solo si el grafo de espera contiene un ciclo.



● *Varias Instancias de Cada Tipo de Recurso*

Para sistemas con múltiples instancias de cada tipo de recurso, se utiliza una versión extendida del algoritmo de detección de interbloqueos similar al algoritmo del banquero:

1. Inicialización: Se crean vectores Work (copia de Available) y Finish (todos false).
2. Búsqueda de Proceso: Se busca un proceso i que pueda ser completado ($\text{Request}[i] \leq \text{Work}$).
3. Asignación de Recursos: Si se encuentra tal proceso, se actualiza Work y Finish[i], y se repite el paso 2.
4. Determinación de Interbloqueo: Si Finish[i] == false para algún i , el sistema está en un estado de interbloqueo.




Utilización del Algoritmo de Detección, Recuperación de un Interbloqueo, Terminación de Procesos

Frecuencia de Detección:

- Depende de la frecuencia de interbloqueos y el número de procesos afectados.
- Si los interbloqueos son frecuentes, el algoritmo de detección debe ejecutarse frecuentemente.

Recuperación de un Interbloqueo:

- Terminación de Procesos:
 - a. Interrumpir todos los procesos involucrados en el interbloqueo.
 - b. Interrumpir procesos uno a uno hasta romper el ciclo de interbloqueo.
 - Apropiación de Recursos:
 - c. Selección de una víctima: Determinar qué recursos apropiar y de qué procesos.
 - d. Anulación: Interrumpir y reiniciar el proceso después de liberar sus recursos.
 - e. Inanición: Asegurar que un proceso no sea continuamente seleccionado como víctima.
- 




Apropiación de Recursos, Resumen

Apropiación de Recursos:

- Liberar recursos de procesos seleccionados para resolver interbloqueos.

Resumen:

- Un estado de interbloqueo se produce cuando dos o más procesos están en espera indefinida.
- Métodos para tratar interbloqueos:
 - a. Prevención o evasión.
 - b. Detección y recuperación.
 - c. Ignorar el problema y reiniciar los sistemas si ocurre.





Resumen del Video: Modelación de Bloqueos

Introducción al Interbloqueo:

- El modelo del sistema se basa en la solicitud, utilización y liberación de recursos.
- Los procesos a menudo necesitan recursos de entrada y salida simultáneamente, lo que puede causar interbloqueos.

Condiciones de Coffman para el Interbloqueo:

1. Exclusión mutua: Solo un proceso puede usar un recurso a la vez.
 2. Retención y espera: Procesos mantienen recursos mientras esperan otros.
 3. No apropiación: Los recursos no pueden ser forzadamente retirados.
 4. Espera circular: Existe una cadena de procesos esperando recursos entre sí.
- 




Resumen del Video: Modelación de Bloqueos

Representación Gráfica de Interbloqueos

- En la representación gráfica de interbloqueos, los procesos se representan con círculos y los recursos con cuadrados. Las flechas que van desde los recursos hacia los procesos indican que el recurso está siendo utilizado por el proceso, mientras que las flechas que van desde los procesos hacia los recursos indican una solicitud del recurso por parte del proceso.

Conclusión

- Para que se produzca un interbloqueo, deben cumplirse las cuatro condiciones de Coffman: exclusión mutua, retención y espera, no apropiación, y espera circular. Si alguna de estas condiciones no se cumple, el sistema no entra en un estado de interbloqueo.
- 

Conclusión

Los interbloqueos son fundamentales para la eficiencia y seguridad de los sistemas operativos, evitando la paralización total de procesos. Se caracterizan por la espera circular de recursos y ocurren cuando se presentan simultáneamente cuatro condiciones específicas. La prevención y el manejo adecuado de interbloqueos son esenciales para mantener la estabilidad del sistema.

Los métodos de prevención, evasión y detección son cruciales para tratar los interbloqueos. Aunque representan un desafío significativo, con un manejo correcto, se pueden convertir en una oportunidad para optimizar y asegurar la continuidad operativa de los sistemas.



GRACIAS POR SU ATENCIÓN

www.reallygreatsite.com

