7. Interbloqueo, Bloqueo Mutuo.

Un sistema está en interbloqueo cuando uno o mas procesos están esperando un evento que no va ha suceder.

Es una situación en la cual uno o mas procesos están bloqueados como consecuencia de que cada proceso tiene retenido un subconjunto de los recursos necesarios para su finalización y tengan que esperar la liberación de los recursos restantes retenidos por otros procesos del mismo grupo.

EN DONDE

Se presenta en sistemas multiprogramados.

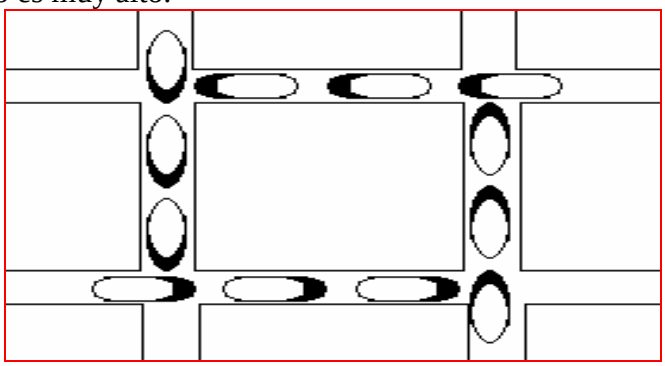
Se presenta cuando los recursos son compartidos y por consiguiente no son suficientes.

Ejemplos.

Trafico de automóviles.

Cuando se da el bloqueo, el trafico se detiene completamente, para resolver este problema, algún(os) carros deben regresar sobre lo avanzado para permitir que los otros puedan avanzar.

El costo del bloqueo es muy alto.



Asignación de recursos insuficientes.

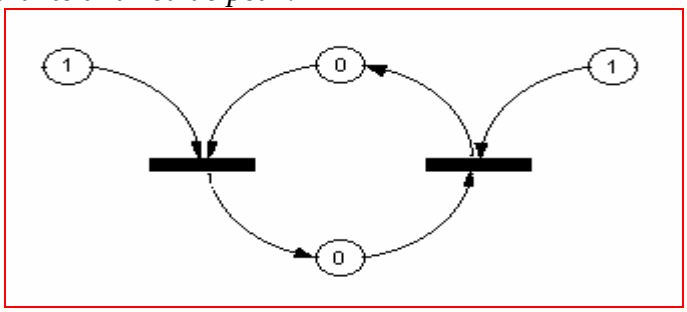
Antes de iniciar un proceso, el planificador empieza a asignarle los recursos que requiere.

Si dos procesos empiezan a solicitar recursos y estos no son suficientes, entonces se puede llegar al bloqueo de ambos procesos.

Ambos estarán esperando que el otro proceso suelte los recursos asignados para poder iniciar, pero como ninguno puede terminar, pues no se ha iniciado, entonces se bloquean.

proceso_uno	proceso_dos	R1	R2	R3
		3	4	2
2R1	1R2	1	3	2
2R2	1R3	1	1	1
2R1	2R2	*	*	1

representación mediante una red de petri.



Los rectángulos son procesos.

Los círculos indican la cantidad de recursos disponibles.

Un proceso se iniciará cuando, exista al menos una unidad de cada insumo de entrada.

Aplazamiento indefinido. (inanición)
Esto sucede cuando un proceso aunque no bloqueado espera un evento que nunca va ha ocurrir.
Si un sistema mantiene los procesos en espera mientras se le asigna recursos o se toma decisiones de planificación, el inicio del proceso puede postergarse indefinidamente.
Esto puede suceder cuando se planifica por prioridad. un proceso puede estar esperando mientras se atiende a los de mayor prioridad.
La solución esta en el *envejecimiento.*

Recursos apropiables.
Son aquellos que pueden ser apropiados por los procesos, ejem. MEMORIA, CPU.
Esto es, que un proceso que hace uso de un espacio de memoria puede ser expulsado por un nuevo proceso quien se apropiara del espacio de memoria.
La CPU se puede alternar rápidamente entre varios procesos. Cuando un proceso no puede apropiarse de la CPU, entonces perderá el control sobre ella y por lo tanto se bloqueara.

Recursos no apropiables. [DES]
Son aquellos que no pueden ser apropiados por otro proceso, hasta que el proceso inicial termine de usar el recurso.

Recursos Compartibles [Disco Duro, Memoria y CPU]
Cuando el recurso puede ser usado por varios procesos, pero solo atiende a uno de ellos por vez.
Se debe administrar esta concurrencia para evitar la espera circular (abrazo mortal)

Recursos no compartible o dedicada [Impresoras, lectora de tarjetas, lectora de disco flexible]
Se pueden asignar a un sólo proceso o a un sólo trabajo a la vez.
Su asignación a diferentes procesos produciría una mezcla caótica.

Condiciones para el bloqueo.
Estas son condiciones necesarias para que se presente el interbloqueo.

Condición	Descripción
Exclusión Mutua.	Los recursos son de uso exclusivo. sólo un proceso puede hacer uso de un recurso
Retencción y Espera	El proceso mantiene la posesión del recurso mientras espera recursos adicionales.
No apropiación	El proceso no suelta el recurso hasta que termine su uso.
Espera Circular.	Los proceso tiene uno o mas recursos que son requeridos por el siguiente proceso.

Si existe bloqueo mutuo entonces se han dado estas condiciones.

Análisis del interbloqueo.
Se plantea cuatro áreas de interés para solucionar el problema del bloqueo mutuo.

prevención	Se trata de eliminar toda posibilidad de que se presente el interbloqueo Pueden resultar muy costosos.
evitación	Se ponen condiciones menos restrictivas que en la prevención. Permite que se presente la posibilidad de un interbloqueo, pero la esquiva cuando esta a punto de suceder.
detección	Se ha permitido que ocurra el interbloqueo. Se debe determinar si ha ocurrido o no un interbloqueo y saber que recursos y procesos están implicados.
recuperación	Permite eliminar el interbloqueo para que el sistema pueda seguir trabajando La recuperación es un proceso complejo, a veces hay que eliminar completamente uno o varios los procesos implicados, perdiéndose el trabajo avanzado.

a. Prevención del Inter Bloqueo.

Para prevenir que se presente el interbloqueo debemos asegurar que por lo menos una de la condiciones necesarias no se cumpla.
Entonces se debe plantear las siguientes políticas, independientes una de otra.

Exclusión mutua.

Se debe de conservar la exclusión mutua para los recursos no compartibles. (ejem. impresora)

No	Problema
1	*No es posible evitar el inter bloqueo negando la condición de exclusión mutua.* Pues por su propia naturaleza algunos recursos no pueden compartirse durante algunos instantes de tiempo.

Retención y Espera

Cada proceso deberá pedir todos los recursos que necesita al mismo tiempo y no podrá seguir hasta obtenerlos todos.

No	Problema
1	*Desperdicio de los recursos.* Un proceso debe pedir todos los recursos, algunos de los cuales pueden no usarse. Alternativa: Dividir el programa en procesos que hagan uso de recursos de manera independiente.
2	*Aplazamiento Indefinido* Pues los recursos pueden no estar disponibles todos al mismo tiempo.

No Apropiación.

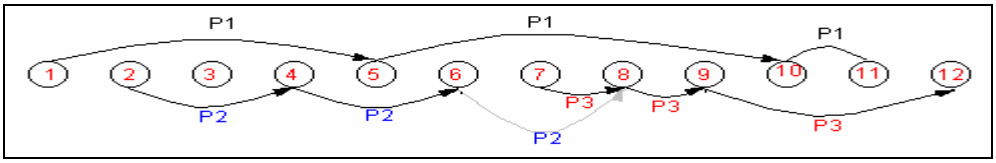
Si a un proceso le falta algún recurso, deberá liberar los recursos ya asignados.

No	Problema
1	*Perdida de trabajo* Un proceso puede perder todo el trabajo realizado cuando no pueda obtener un recurso. Alternativa : Evaluar la frecuencia con que ocurre el problema, si es poca se puede implementar.
2	*Aplazamiento Indefinido* El proceso se aplaza mientras pide y libera recursos

Espera Circular.

A los recursos se les asigna un *ordenamiento lineal* (numeración única ascendente).
Los números asignados a los recursos deben reflejar el orden natural en que son solicitados.
Si a un proceso se le ha asignado un recurso en lo sucesivo solo podrá pedir los recursos que siguen en el ordenamiento.

No	Problema
1	*Perdida de trabajo* Un proceso que requiera los recursos en orden diferente puede conservar los recursos adquiridos mucho tiempo, hasta que se liberen los recursos requeridos.
2	*Limitación a los programadores* Se debe programar de tal manera que los recursos se requieran en el orden indicado.



¿Como demuestro que aplicando este algoritmo no se presentará el interbloqueo?

b. Evitación del Inter Bloqueo.

Los métodos de prevención adolecen de que son muy costosos. (bajo uso de los recursos, reducción de la productividad del sistema).

Algortimo del banquero de Dijkastra

Requiere información de como se solicitaran los recursos.

Declarando el numero máximo de recursos de cada tipo que pueda necesitar.

Indicando el orden en que solicitara sus recursos y cuando los liberara.

El algoritmo debe examinar dinámicamente el estado de la asignación de los recursos para evitar que se presente la espera circular.

Sean D(r) las unidades del recurso r con que cuenta el sistema.
R(r,p) el requerimiento máximo de recursos del tipo r, para el proceso p.
Entonces el sistema podrá asignar un requerimiento R(r,p) si: D(r) >= P(r) + R(r,p) tal que: P(r) = P(r) + R(r,p) donde P(r), es el numero de recursos asignados del recurso r
Cuando el proceso p libere los recursos asignados P(r) = P(r) - R(r,p)

Estados del sistema:

Se definido por el numero de recursos disponibles y asignados y por la demanda máxima de los procesos

Estado seguro	Si el sistema puede asignar recursos a cada proceso(hasta el máximo), siguiendo algún orden y así evitar el inter bloqueo.
Estado inseguro	El sistema no puede impedir que los procesos soliciten recursos de tal manera que ocurra el bloqueo mutuo.

Ejemplos de transición de un estado seguro a uno inseguro.

Procesos		Estado Seguro	Estado Inseguro
Necesidad	Proc	Asignado	Asignado
4	P1	1	1
6	P2	4	4
8	P3	5	6
		2	1
Se dispone de 12 recursos			

Estado seguro.

Existe al menos una forma de que terminen todos los procesos.

Se podría asignar los 2 recursos que sobran a P2 y esperar a que este luego libere luego 6 recursos.

Estado inseguro.

No se puede garantizar que los procesos terminen.

Si P3 solicita 1 recurso, sobraría 1 recurso.

En este estado no se puede garantizar que los procesos terminen pues algunos de ellos podría pedir un recurso más que no puede ser satisfecho.

Pero este no es un estado de bloqueado, sino que no puede garantizarse que termine.

Análisis del Algortimo del banquero de Dijkastra
Con este algoritmo se satisfacen las condiciones de espera, no apropiación y espera circular, pero los procesos necesitan el uso exclusivo de los recursos.
Los usuarios puede solicitar los recursos cuando lo necesiten.
El sistema solo satisface las peticiones que lleven a un estado seguro.

No	Problema
1	numero fijo de recursos asignables No siempre se puede contar con los recursos declarados
2	Numero de usuarios constante La población de usuarios en los sistemas grandes puede cambiar constantemente
3	Espera limitada El algoritmo requiere que el sistema satisfaga todas las peticiones en un tiempo finito
4	Devolución de los recursos asignados El algoritmo requiere que los usuarios salden sus prestamos (recursos asignados) en un tiempo finito
5	Declaración de necesidades Muchos usuarios no saben que recursos ven ha necesitar.

c. Detección del Inter Bloqueo.
Permite determinar si existe interbloqueo, identificando a los recursos y procesos implicados.
La operación de este algoritmo implica, uso de CPU.

Algoritmo para detectar el interbloqueo.
USO DE LAS REDES DE PETRI

Modelo formal de una Red de Petri.
Una RP se puede representar por una quíntuple definido por:

$$N = (L,P,A,M,D)$$

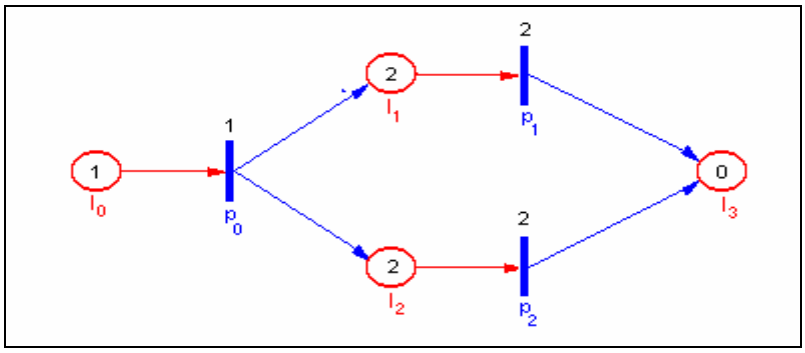
donde:

L = (l1,l2,l3,.....,lm)	Conjunto finito de Lugares
P = (p1,p2,p3,.....,pn)	Conjunto finito de Transiciones.
$A \subset I \cup O$	Conjunto de arcos direccionados entre lugares y transiciones y entre transiciones y lugares.

además:

I = {L X P} = {(l1,p1),...(l1,pn),...(lm,p1),...(lm,pn)}	Conjunto de arcos que representa los lugares lh de entrada a las transiciones pi.
O = {P X L} = {(p1,l1),...(p1,lm),...(pn,l1),...(pn,lm)}	Conjunto de arcos que representa los lugares lj las transiciones pk.
M: L --> F, M(l1,l2,l3,.....,lm) = (f1,f2,f3,.....,fm)	conjunto de marcas o fichas asociadas a cada luga
D: P --> H, D(p1,p2,p3,.....,pn) = (h1,h2,h3,.....,hn)	conjunto de tiempos asociados a cada transición.

Ejemplo:



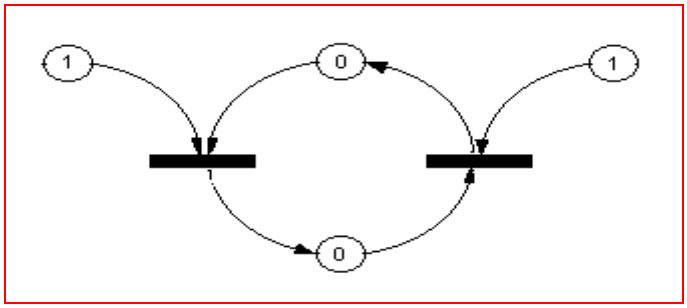
L = (l0, l1, l2, l3)
P = (p0, p1, p2)

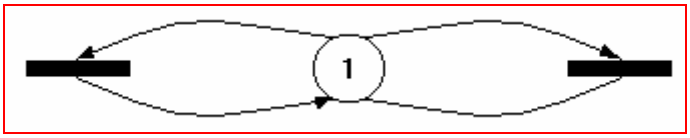
A ⊂ I U O
donde:
I = {LxP} = {(l0,p0),(l1,p1), (l2,p2)}
O = {PxL} = {(p0,l1),(p0,l2), (p1,l3), (p2,l3)}

M : L --> F, M(l0,l1,l2,l3) = (1,2,2,0)
D : P --> H, D(p0,p1,p2) = (1,2,2)

Ejemplos:
Para determinar si existe inter bloqueo se debe de analizar el gráfico.

Reducción del gráfico.
Para reducir el gráfico, se debe de borrar el arco que conecta el recurso con el proceso, cuando un proceso tenga todos los recursos necesarios para que se inicie.
Si la gráfica no puede ser reducida, para todos los proceso, entonces los procesos irreductibles forma el conjunto de procesos que están en interbloqueo.
No interesa el orden en que se reduzca el grafo, el resultado siempre será el mismo.





Uso del algoritmo de detección

¿Cuándo se debe usar este algoritmo?.

Se debe de determinar dos aspectos en el uso de este algoritmo:

Con que frecuentemente ocurre un bloqueo mutuo. Si son frecuentes, entonces se debe invocar frecuentemente al algoritmo. Pero su uso requiere tiempo de CPU.
Cuantos procesos pueden estar afectados por el inter bloqueo. Los procesos afectados estarán paralizados, degradando la productividad del sistema

d. Recuperación después del Inter Bloqueo.

Cuando el algoritmo de detección determina que existe el interbloqueo, existen las siguientes alternativas:

1. Abortar uno o mas procesos para romper la espera circular
2. Quitar uno o mas recursos a los procesos bloqueados.

Terminación de procesos.

Con estos métodos el sistema recupera todos los recursos asignados a los procesos terminados.

Abortar todos los procesos que están en interbloqueo.

Rompe el interbloqueo definitivamente pero es muy costoso.

Pero no es fácil terminar un proceso, este podría estar realizando una operación que no se puede recuperar.

Abortar un proceso gradualmente hasta lograr eliminar el interbloqueo.

Incurre en tiempo de procesamiento, pues luego de abortar cada proceso se debe invocar al algoritmo de detección, hasta estar seguros que no existe interbloqueo.

El problema es determinar que proceso se debe eliminar a continuación. Se tratara de abortar el proceso 'menos costoso'.

Para determinar el proceso 'menos costoso' se deberá considerar los siguiente factores:

Prioridad del proceso, tiempo de ejecución y tiempo ya ejecutado, cantidad de recursos asignados, facilidad de expropiar el recurso, recursos adicionales que necesita para terminar, proceso que deberán de abortarse adicionalmente.

Expropiación de los recursos.

Los recursos se expropian y se asignan a otros proceso.

Se debe considerar;

Selección del proceso.

Se debe seleccionar el proceso y el orden de expropiación de los recursos para 'minimizar el costo'.

Se debe considerar: número de recursos asignados, cantidad de tiempo ya consumido, etc.

Retroceso.

Si se expropia un recursos a un proceso se debe de retroceder el proceso hasta llegar a un estado seguro, como esto no es sencillo se prefiere, retroceder totalmente.

Bloqueo indefinido.

Como garantizar que los recursos a expropiar no son del mismo proceso.

Si la decisión de seleccionar a un proceso se basa en el costo, puede suceder que siempre se elija a la misma víctima.

Si se llega a esto entonces habrá un situación de bloqueo indefinido.

Una solución es incluir el numero de retrocesos en el factor costo.