Sistemas Operativos II

TEMA 4.2
Interbloqueos

Contenido

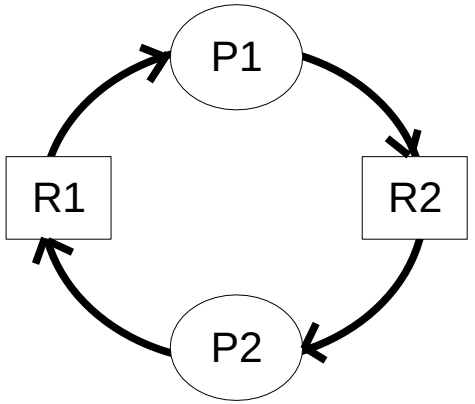
Interbloqueos

1. Concepto de interbloqueo y aplazamiento indefinido
2. Recursos y categorías de recursos
3. Condiciones de interbloqueo
4. Prevención de interbloqueos
5. Evitación de interbloqueos
6. Detección y recuperación de interbloqueos

Concepto de Interbloqueo

Ejemplo. Es posible que dos procesos necesiten simultáneamente los dos mismos recursos y los soliciten en el siguiente orden:

- P1 solicita el recurso R1, y el SO se lo concede.
- P2 solicita el recurso R2, y el SO se lo concede.
- P1 solicita el recurso R2. Como R2 está asignado a P2, P1 se bloquea esperando que R2 sea liberado.
- P2 solicita el recurso R1. Como R1 está asignado a P1, P2 se bloquea esperando que R1 sea liberado.
- **INTERBLOQUEO.**
- **BLOQUEO MUTUO.**
- ***DEADLOCK.***



Concepto de Interbloqueo

Definición:

- Un conjunto de procesos está en bloqueo mutuo (ó interbloqueo ó *deadlock*) si **todos los procesos del conjunto están esperando un evento que sólo otro proceso del conjunto puede causar**.
- En la mayor parte de los casos, el evento que cada proceso está esperando es la liberación de algún recurso que actualmente está en poder de otro miembro del conjunto.
  - Ni el número de procesos ni el número y tipo de los recursos son datos relevantes para esta definición.

En este curso estudiaremos las condiciones de interbloqueo, técnicas de prevención, evitación, detección y recuperación de I/B.

4.2.1 RECURSOS Y CATEGORÍAS DE RECURSOS

Recursos y Categorías

- Un recurso puede ser:
  - Un **dispositivo físico** de E/S: impresora, lector de CD, ...
  - Un **elemento SW** como un conjunto de registros de una base de datos, bloqueados para realizar una transacción segura.
  - **Informalmente**: “cualquier elemento que sólo pueda ser usado por un proceso en un instante dado”.

Categorías de recursos:

- **Expropiable**: aquel que se puede arrebatar al proceso que lo tiene sin que haya efectos adversos.
- **No expropiables**: aquel recurso que no puede quitársele a su poseedor actual sin hacer que el cómputo falle.

En general, en los interbloqueos intervienen recursos no expropiables.

Recursos y Categorías

Ejemplo: dos procesos A y B, van a necesitar la impresora.

- A se coloca en memoria, solicita la impresora y se le concede, pero no llega a imprimir. Libera la memoria y entra B.
- B tiene la memoria y solicita la impresora, pero como la tiene asignada A, no puede continuar.
- Interbloqueo en potencia, la memoria es expropiable.

- La **secuencia de sucesos que se requiere para usar un recurso** es:
  - **Solicitar el recurso**. Si el recurso no está disponible, el proceso que realiza la solicitud tiene que esperar.
  - **Utilizar el recurso**.
  - **Liberar el recurso**:
    - O bien se despierta a todos los procesos que estaban esperando
    - O bien los procesos que esperan vuelven a intentar el acceso al recurso pasado un tiempo.

4.2.2 CONDICIONES DE INTERBLOQUEO

Condiciones de Interbloqueo

En 1971, Coffman y sus colaboradores demostraron que deben cumplirse las siguientes cuatro condiciones para que haya un bloqueo mutuo:

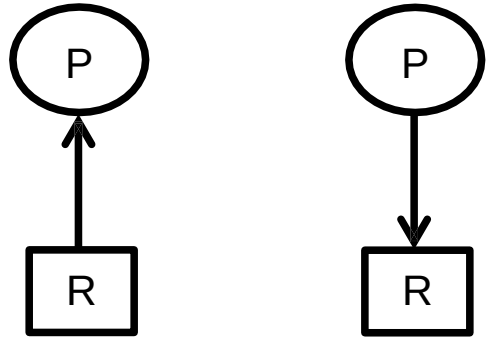
1. **Exclusión mutua**: cada recurso, o bien está asignado a un solo proceso o bien está disponible.
2. **Retener y esperar**: los procesos que actualmente tienen recursos que les fueron otorgados previamente pueden solicitar nuevos recursos.
3. **No expropiación**: no es posible quitarle a un proceso los recursos que le fueron otorgados previamente. El proceso que los tiene debe liberarlos explícitamente.
4. **Espera circular**: debe haber una cadena circular de dos o más procesos, cada uno de los cuales está esperando un recurso retenido por el siguiente miembro de la cadena.

Si alguna de las condiciones no se cumple, no puede haber bloqueo mutuo.

Condiciones de Interbloqueo

En 1972, Holt mostró un método gráfico para modelar las condiciones de Coffman y averiguar si existe interbloqueo.

- **Grafos dirigidos con dos tipos de nodos:**
  - **Círculos, que representan procesos**
  - **Cuadrados, que representan recursos**
- **Si la arista va de un recurso a un proceso ⇒ recurso asignado.**
- **Si la arista va de un proceso a un recurso ⇒ proceso espera recurso porque éste ya ha sido asignado.**



Asignado Espera

Si existe un ciclo en un grafo construido siguiendo estas reglas, los procesos implicados en el ciclo sufren INTERBLOQUEO

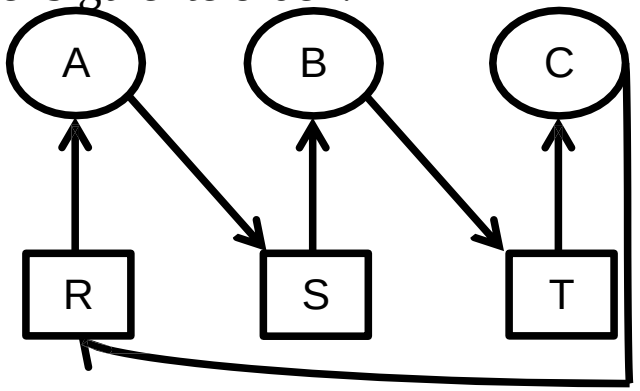
Condiciones de Interbloqueo

Ejemplo: supongamos los procesos A, B, C, y los recursos R, S y T. Supongamos que suceden las siguientes peticiones de recursos:

- A: solicita R, solicita S, libera R, libera S.
- B: solicita S, solicita T, libera S, libera T.
- C: solicita T, solicita R, libera T, libera R.

Si el SO decide que se ejecuten las peticiones en el siguiente orden:

1. A solicita R
2. B solicita S
3. C solicita T
4. A solicita S
5. B solicita T
6. C solicita R



INTERBLOQUEO (Ciclo en el grafo)

Condiciones de Interbloqueo

Estrategias empleadas para manejar bloqueos mutuos:

- **Algoritmo del avestruz.**
  - Pasos básicos:
    - **Ver el peligro** (interbloqueo)
    - Meter la cabeza debajo de la arena (**no hacer nada**).
  - Es la estrategia más **sencilla y barata**. Los profesionales opinan de diversa manera acerca de esta estrategia:
    - Matemáticos: lo consideran inaceptable.
    - Ingenieros: lo aceptan porque es mucho más probable un fallo de HW que una caída del sistema por interbloqueo.
- **Prevención de interbloqueos**, negando estructuralmente alguna de las condiciones de Coffman.
- **Evitación de interbloqueos de manera dinámica**, controlando la asignación de recursos.
- **Detección de interbloqueos y recuperación**.

4.2.3 PREVENCIÓN DE INTERBLOQUEOS

Prevención de Interbloqueos

- Si para que se produzca I/B deben cumplirse las cuatro condiciones de Coffman, la manera de prevenir I/B es **forzar a que alguna de esas condiciones no se cumpla**.

Eliminación de condición de Exclusión Mutua:

- Dos procesos podrían utilizar simultáneamente el mismo recurso.
  - **Problema**: la impresora podría ser usada a la vez por dos procesos !!
  - **Solución**: SPOOL (Simultaneos Peripherial Operation On Line), que encola los trabajos que van a la impresora de modo que sólo el proceso SPOOL utiliza el recurso impresora.
    - Problema: no todos los recursos pueden ser gestionados con *spool*, además hay competencia por usar el *spool*.

Prevención de Interbloqueos

Eliminación de condición de Retener y Esperar:

- Forzar a los procesos a solicitar todos los recursos antes de entrar en ejecución. Problemas:
  - No todos los procesos conocen *a priori* los recursos que necesitarán.
  - Los recursos no se aprovechan de manera óptima.
- Exigir que los procesos que solicitan un recurso liberen temporalmente todos los recursos que retienen. Sólo si la petición tiene éxito el proceso podrá volver a recibir los recursos que poseía. Problema:
  - Se pierde eficiencia en el uso de los recursos.

Eliminación de condición de No Expropiación:

- Eliminar esta condición significaría que, por ejemplo, el SO podría arrebatar la impresora a cualquier proceso que la estuviera usando, interrumpiendo la impresión.

Prevención de Interbloqueos

Eliminación de condición de Espera Circular:

- Forzar a que un proceso sólo pueda disponer de un recurso en un instante dado. Cuando necesite de otro recurso, deberá liberar el primero. Problema:
  - Si un proceso necesita simultáneamente dos recursos (ej. E/S muy larga) esta solución no es válida.
- Crear una **numeración global** para todos los recursos, por *ejemplo*:
  1. DVD
  2. Impresora
  3. Grabadora de CDs
  4. Unidad de cinta (ZIP)

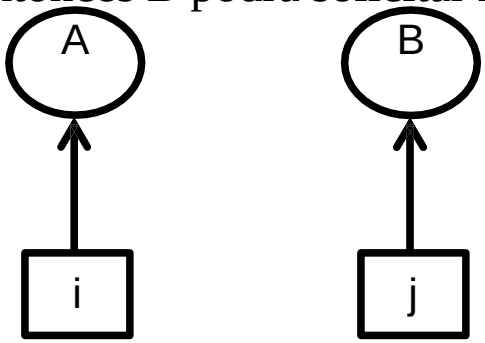
  de modo que los procesos siempre **soliciten los recursos por orden creciente de número**. Para nuestro ejemplo, un proceso podría solicitar la impresora y luego la unidad de cinta, pero no al revés. **Con esta regla, el grafo de asignación de recursos nunca tendrá ciclos**.

Prevención de Interbloqueos

Ejemplo:

Dos procesos A y B tienen asignados los recursos con numeración i y j. Estos recursos serán distintos, i ≠ j.

- Si i < j, entonces A podrá solicitar j, pero B no podrá solicitar i.
- Si i > j, entonces B podrá solicitar i, aunque A no podrá solicitar j.



No habrá ciclo en el grafo,
por tanto no habrá I/B
(extensible a *n procesos)*

- Problema: puede que ninguna ordenación satisfaga al sistema.

- **Variante**: obligar a los procesos a solicitar recursos con un número mayor que los que ya poseen. Al liberar todos se "reinicia" su cuenta.

4.2.4 EVITACIÓN DE INTERBLOQUEOS

Evitación de Interbloqueos

Algoritmo del Banquero (un sólo tipo de recursos):

- Algoritmo de planificación para la asignación de recursos que evita el bloqueo mutuo (Dijkstra, 1965).
- Toma como modelo la manera en que un banquero de una pequeña ciudad administraría las líneas de crédito que concede a sus clientes:

Nombre	Utilizados ↓	Máximo ↓
Andrés	0	6
Bárbara	0	5
Pedro	0	4
Marta	0	7

Disponibles: 10

Se tienen cuatro clientes (Andrés, Bárbara, Pedro, Marta), a cada uno de los cuales se ha otorgado un límite máximo en su línea de crédito (6, 5, 4, 7).

El banquero sólo dispone de 10 ud, porque sabe que es improbable que todos los clientes soliciten el máximo a la vez

- Analogía: los clientes son procesos, las unidades máximas de crédito son el número máximo de recursos de un tipo (por ejemplo, impresoras) que un proceso solicitará, y el banquero es el SO.

Evitación de Interbloqueos

Algoritmo del Banquero (un sólo tipo de recursos): (sigue)

- Un **estado** de este tipo es **seguro** si existe una secuencia de otros estados que conduzca a una situación en la que todos los procesos obtienen todos los recursos y finalizan.

Nombre	Utilizados ↓	Máximo ↓
Andrés	1	6
Bárbara	1	5
Pedro	2	4
Marta	4	7

Disponibles: 2

(a)

Nombre	Utilizados ↓	Máximo ↓
Andrés	1	6
Bárbara	2	5
Pedro	2	4
Marta	4	7

Disponibles: 1

(b)

- La figura (a) muestra un estado seguro, mientras que la (b) muestra un estado inseguro.

Evitación de Interbloqueos

Algoritmo del Banquero (un sólo tipo de recursos): (sigue)

- El algoritmo del banquero consiste en analizar el sistema a cada petición de un recurso:
  - Si **la asignación** de ese recurso **conduce a un estado seguro**, la asignación **se realiza**.
  - Si **la asignación conduce a un estado inseguro**, se hace **esperar** a la petición (se suspende el proceso que la hace).
- Para **determinar si un estado es seguro**, el sistema debe comprobar si con los recursos disponibles puede satisfacer todas las peticiones del proceso que está más cerca de su máximo.
  - Si los tiene, se consideran disponibles el número máximo de recursos de ese proceso y se verifica si se satisface al siguiente más cercano a su máximo, y así hasta el final. **Si todos se pueden satisfacer, el estado es seguro**.

Evitación de Interbloqueos

Algoritmo del Banquero para múltiples tipos de recursos:

El estado de un conjunto de procesos se forma construyendo dos matrices:

- ***MA***, recursos asignados.
- ***MN***, recursos necesitados (aún se necesitan).

Y tres vectores que representan lo siguiente:

- ***E***, recursos existentes.
- ***P***, recursos asignados.
- ***A***, recursos disponibles.

El algoritmo para verificar si un estado es seguro sigue estos pasos:

1. Buscar una fila ***r*** de la matriz ***MN*** cuyos elementos sean menores o iguales que todos los del vector ***A***. Si no existe esta fila podría haber I/B.
2. Si esa fila existe, suponemos que el proceso solicita todos los recursos que le faltan y termina, liberando los recursos de los que ya disponía. Se marca el proceso como terminado, y se suma a ***A*** la fila de ***MA*** correspondiente a él.
3. Repetir 1 y 2 hasta que todos los procesos se marquen como terminados. En ese caso el estado será seguro. Si no se llega, **podría haber I/B**.

Evitación de Interbloqueos

Algoritmo del Banquero para múltiples tipos de recursos: (sigue)

Ejemplo:

Cinco procesos, A, B, C, D y E; y cuatro tipos de recursos U (cinta), S (escáner), I (impresora) y CD. Se construyen las matrices y vectores:

Pr.	U	S	I	CD
A	3	0	1	1
B	0	1	0	0
C	1	1	1	0
D	1	1	0	1
E	0	0	0	0

MA

(Recursos asignados)

Pr.	U	S	I	CD
A	1	1	0	0
B	0	1	1	2
C	3	1	0	0
D	0	0	1	0
E	2	1	1	0

MN

(Recursos necesitados)

$E = (6\ 3\ 4\ 2)$

$P = (5\ 3\ 2\ 2)$

$A = (1\ 0\ 2\ 0) = E - P$

¿Es seguro este estado?

Evitación de Interbloqueos

Algoritmo del Banquero para múltiples tipos de recursos: (sigue)

Ejemplo: en el mismo escenario, el estado es el que representan las siguientes matrices y vectores, ¿podría satisfacerse la petición de una impresora por parte del proceso ***E*** ?

Pr.	U	S	I	CD
A	3	0	1	1
B	0	1	1	0
C	1	1	1	0
D	1	1	0	1
E	0	0	0	0

MA
(Recursos asignados)

Pr.	U	S	I	CD
A	1	1	0	0
B	0	1	0	2
C	3	1	0	0
D	0	0	1	0
E	2	1	1	0

MN
(Recursos necesitados)

E = (6 3 4 2)

P = (5 3 3 2)

A = (1 0 1 0)

Inconvenientes de los algoritmos del banquero:

- Necesita saber los recursos que solicitará cada proceso al iniciar la ejecución.
- No considera variaciones en el número de procesos y recursos.

4.2.5 DETECCIÓN Y RECUPERACIÓN DE INTERBLOQUEOS

Detección y Recuperación de I/B

- El SO realizará **comprobaciones periódicas** para asegurarse de que no hay interbloqueo.
  - Cada vez que un recurso se asigna o libera, el SO actualiza el grafo de recursos y procesos.
    - Si encuentra algún ciclo, el SO "mata" a uno de los procesos que participen en el ciclo.
    - Si aún así, sigue habiendo ciclos, continúa eliminando procesos hasta que los ciclos desaparecen.
- Variante: únicamente **comprobar** "de vez en cuando" **que no hay ningún proceso que lleve bloqueado demasiado tiempo**. Sin grafo.
  - Si se detecta algún proceso así, es eliminado inmediatamente.
- La detección y recuperación es una estrategia usada con profusión en *macrocomputadores*, porque en ellos terminar y reiniciar un proceso suele ser aceptable.
  - Necesita de la correcta restauración del estado anterior.