Sistemas Operativos
Interbloqueo (Deadlocks)

Semestre II-2013

Interbloqueo

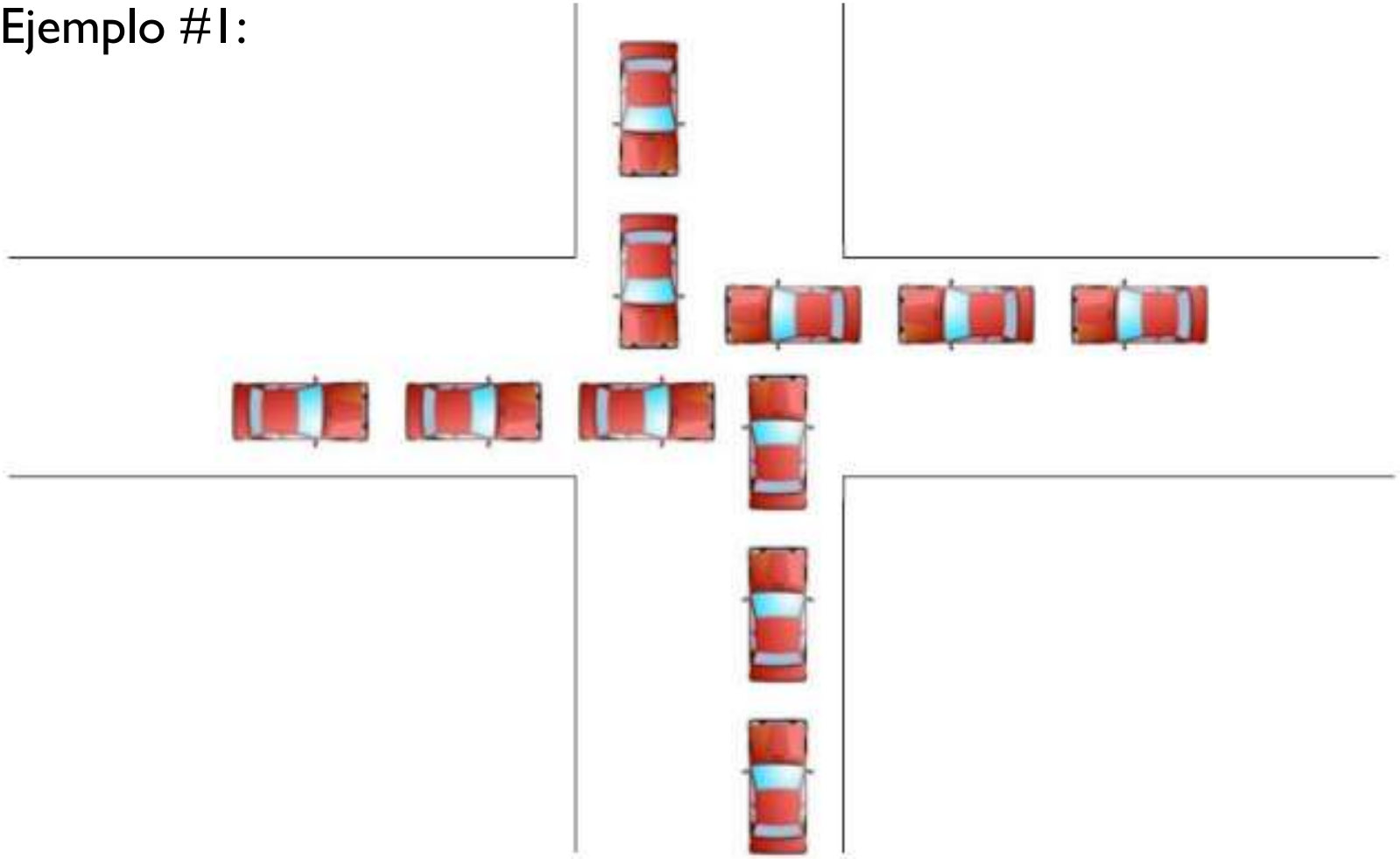
- Modelo de Sistema
- Caracterización de Interbloqueo
- Métodos para Lidiar con el Interbloqueo
- Prevención de Interbloqueo

Objetivos

- Describir el fenómeno conocido como Interbloqueo en sistemas de computo
- Presentar diferentes métodos para prevenir o evitar Interbloqueo en sistemas de computo

Ejemplos Interbloqueo

- Ejemplo #1:



Ejemplos Interbloqueo

- Ejemplo #2:
  - Un sistema que tiene dos unidades de disco
  - P1 y P2 cada uno tiene una unidad de disco y necesita la otra unidad

- Ejemplo #3:
  - Semáforos A y B, inicializados a 1

P0	P1
wait (A);	wait(B);
wait(B);	wait(A);

Ejemplos Interbloqueo

- Ejemplo #4:

```
/* thread one runs in this function */
void *do work one(void *param)
{
   pthread mutex lock(&first mutex);
   pthread mutex lock(&second mutex);
   /** * Do some work */
   pthread mutex unlock(&second mutex);
   pthread mutex unlock(&first mutex);
   pthread exit(0);
}
/* thread two runs in this function */
void *do work two(void *param)
{
   pthread mutex lock(&second mutex);
   pthread mutex lock(&first mutex);
   /** * Do some work */
   pthread mutex unlock(&first mutex);
   pthread mutex unlock(&second mutex);
   pthread exit(0);
}
```

Definición

- "Un conjunto de procesos estará en un estado de interbloqueo, cuando todos los procesos del conjunto estén esperando a que se produzca un suceso que sólo puede producirse como resultado de la actividad de otro proceso del conjunto" – Silberschatz, et al.

Interbloqueo - Tips

- Un interbloqueo puede producirse:
  - Entre hilos dentro de un proceso.
  - Entre hilos de distintos procesos.
- Las solicitudes y liberaciones son llamadas al sistema:
  - Solicitudes y liberaciones de dispositivos (request() y release()).
  - Apertura y cierre de archivos (open() y close ()).
  - Asignación y liberación de memoria (allocate() y free ()).
- La solicitud y liberación de recursos que el SO no gestiona se pueden realizar a través de:
  - Operaciones wait() y signal() de semáforos.
  - E/S mapeada a memoria.
- Una tabla del SO registra el estado de cada recurso.
  - Libre o asignado, y en caso de asignación a cuál proceso se ha sido asignado dicho recurso.

Modelo del sistema

- Un sistema consta de un número finito de recursos, los cuales se distribuyen entre una serie de procesos en competencia.
- P es igual al conjunto de todos los procesos dentro del sistema.
  - P={P1,P2,…..Pn }
- Tipos de Recursos.
  - R={R1, R2, . . ., Rm}
    - Tiempo de CPU.
    - Buses de E/S.
    - Memoria principal.
    - Memoria secundaria.
    - Dispositivos.
      - Unidades de Cinta, Impresoras.
    - Estructuras de datos, archivos, BD y semáforos.

Modelo del sistema

- Cada tipo de recurso Ri tiene Wi instancias.
- Cada proceso utiliza los recursos según el siguiente orden:
  - Solicitud (request).
  - Uso (use).
  - Liberación (release).
- S = {S0, S1, …}
  - Es el conjunto de estados que representan la asignación de Rj a Pi.
    - Se produce un cambio de estado cuando un proceso realiza una acción.
    - Esto permite identificar una situación de interbloqueo.

Caracterización de un interbloqueo

- Condiciones de Interbloqueo (Condiciones de Coffman).
- Exclusión Mutua.
  - Sólo un proceso puede usar un recurso a la vez.
- Retención y Espera.
  - Un proceso puede retener unos recursos en espera que se le asignen otros que actualmente están retenidos por otros procesos o al mismo tiempo que solicita otros.
- No apropiación.
  - Ningún proceso puede ser forzado a abandonar un recurso que esté reteniendo. Un recurso sólo puede ser liberado voluntariamente por el proceso que lo retiene, después de que dicho proceso haya completado su tarea.

Caracterización de un interbloqueo

- Circulo vicioso de espera o Espera circular.
  - Existe una cadena cerrada de procesos, cada uno de los cuales retiene, al menos un recurso que necesita el siguiente proceso de la cadena, es decir:
    - "Debe existir un conjunto {P0, P1, …, Pn} de procesos en espera, tal que P0 está esperando por un recurso retenido por P1, P1 está esperando por un recurso retenido por P2, …, Pn–1 está esperando por un recurso retenido por Pn, y Pn está esperando por un recurso retenido por P0"

Caracterización de un interbloqueo

- Condiciones necesarias y suficiente.
  - Las tres primeras condiciones son necesarias para que ocurra un interbloqueo. Su existencia deriva del diseño del SO y sólo garantizan que puede ocurrir un interbloqueo, no que existe uno actualmente.
  - La condición de Espera Circular, es una condición suficiente para que ocurra el interbloqueo. Se produce durante la ejecución de procesos en SO que cumplen con las condiciones de diseño anteriores.
    - Observaciones.
- Si se dan las condiciones necesarias (diseño) y la condición suficiente (ejecución), podemos decir que estamos en presencia de un interbloqueo.

Tipos de recursos

- Recursos Reutilizables.
  - Aquellos que pueden ser utilizado por un proceso y no se agota con el uso. Se pueden reservar y después devolver al sistema una vez que el proceso ha acabado de utilizarlos.
  - Ejemplos:
    - Procesador.
    - Buses de E/S.
    - Memoria principal.
    - Memoria secundaria.
    - Dispositivos.
      - Unidades de Cinta, Impresoras.
    - Estructuras de datos, archivos, BD y semáforos.
  - El sistema siempre tiene una cantidad fija y limitada de unidades de un recurso reutilizable.

Tipos de recursos

- Recursos Consumibles.
  - Aquellos que pueden ser creados (producidos) y destruidos (consumidos).
  - No hay límite en el # de recursos consumibles de un tipo en particular.
  - Por ejemplo datos de entrada.
    - Se reservan pero nunca son liberados.
  - Cuando un proceso adquiere un recurso, éste deja de existir.
  - Ejemplos:
    - Interrupciones.
    - Señales.
    - Mensajes.
    - Información en buffers de E/S.

Grafo de asignación de recursos

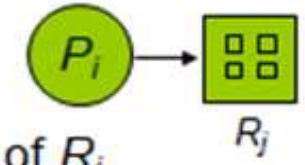
- El grafo consta de un conjunto de vértices V y de un conjunto de aristas E.
  - V está dividido en dos tipos:
    - P = {P1, P2, …, Pn}
      - Conjunto conformado por todos los procesos activos en el sistema.
    - R = {R1, R2, …, Rm}
      - Conjunto conformado por todos los tipos de recursos en el sistema.
    - Cj es el número de unidades en el sistema del tipo de recurso Rj. Los puntos dentro de un nodo recurso representa el número de unidades del recurso.
- Arista de Solicitud.
  - Pi → Rj
- Arista de Asignación.
  - Rj → Pi

Grafo de Asignación de Recursos

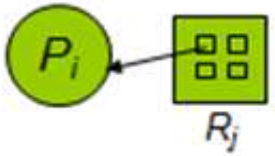
- Process

- Resource Type with 4 instances

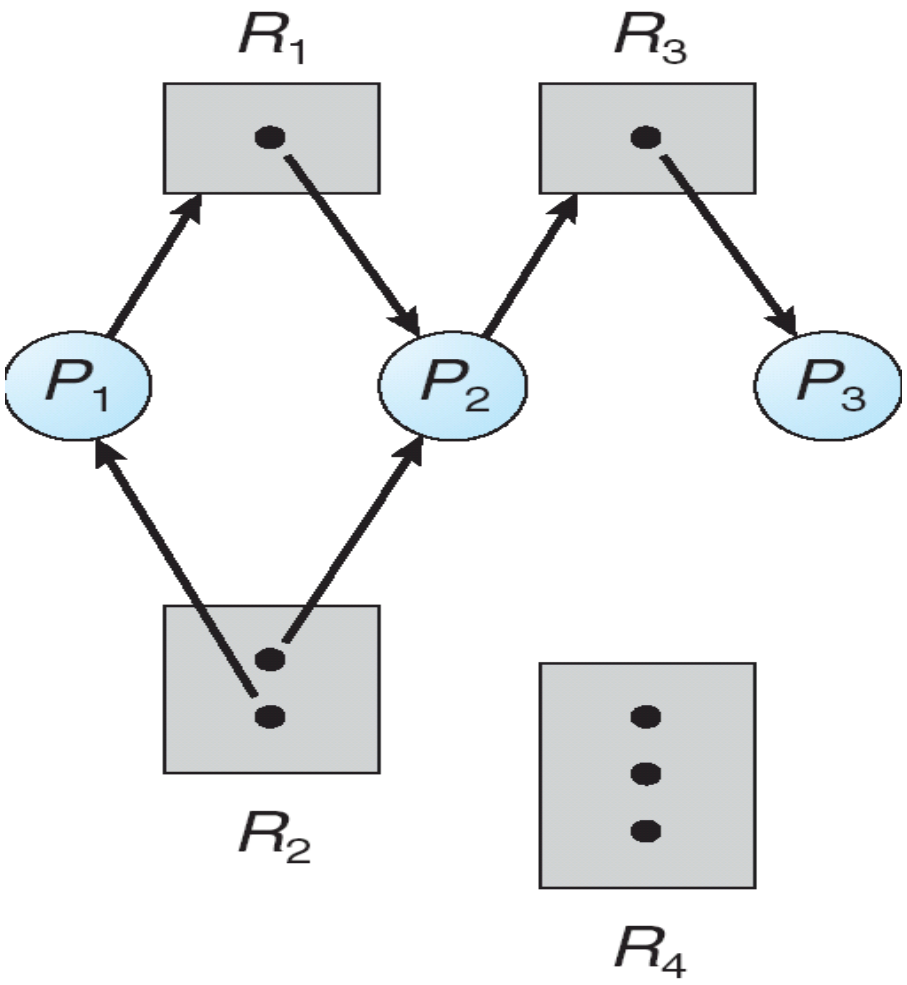
- P_i requests instance of R_j



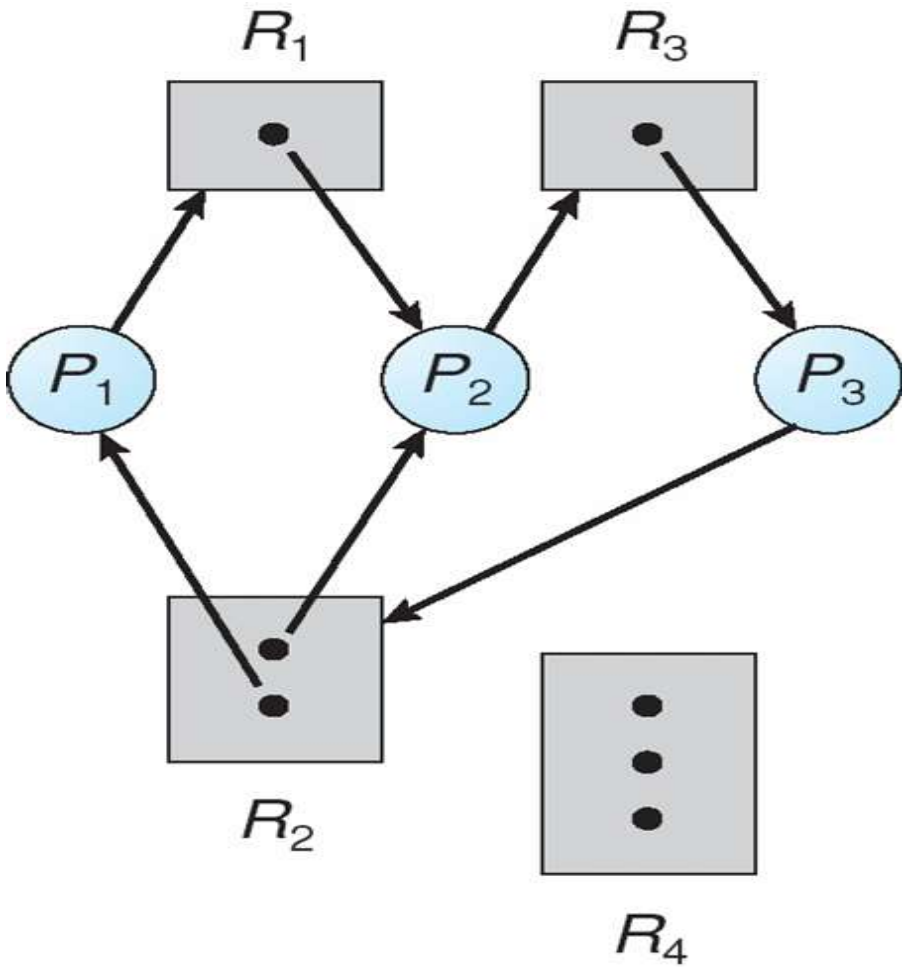
- P_i is holding an instance of R_j



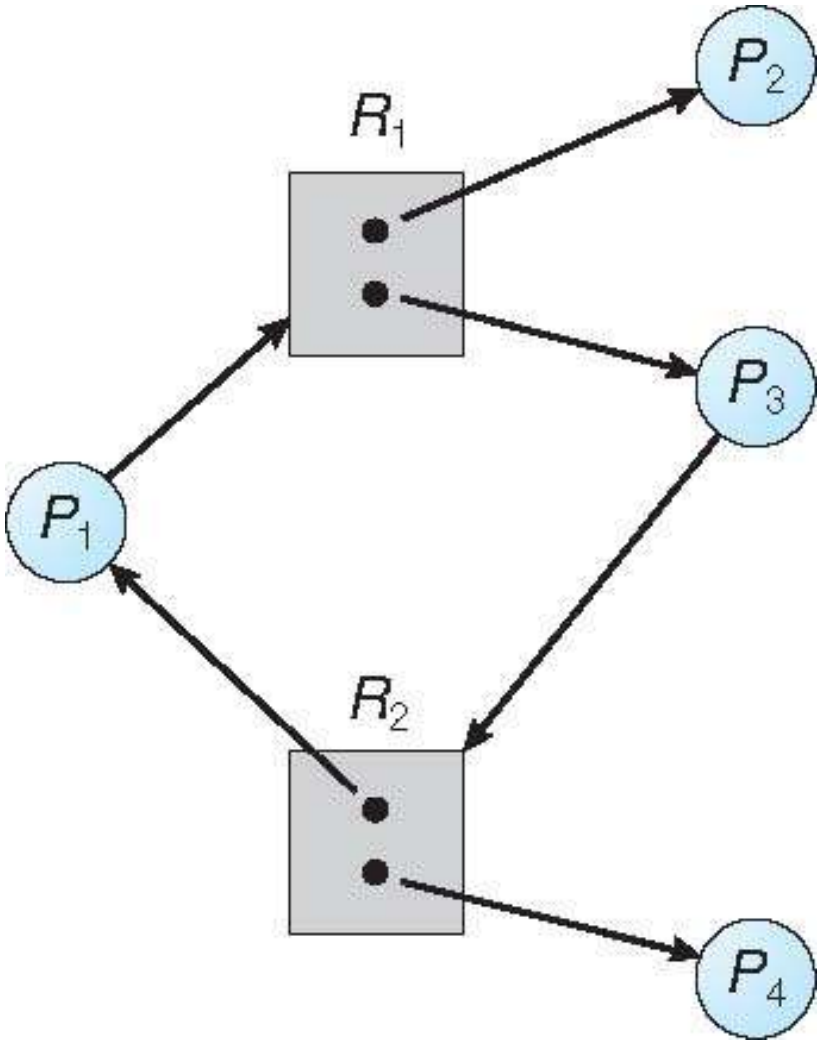
Ejemplo de un Grafo de Asignación de Recursos



Ejemplo de un Grafo de Asignación de Recursos con Interbloqueo



Ejemplo de un Grafo de Asignación con un Ciclo (sin Interbloqueo)



Hechos Básicos

- Si un grafo de asignación de recursos no contiene ciclos → No existe Interbloqueo

- Si un grafo de asignación de recursos contiene ciclos:
  - Si solo existe una instancia por recurso, entonces hay interbloqueo
  - Si existen diversas instancias por recurso, podría existir un interbloqueo

Métodos para manejar los interbloqueos

- Prevención y Evasión.
  - Asegurar que el sistema "nunca" estará en un estado de interbloqueo, se diseñan los gestores de recursos de modo que se garantiza el incumplimiento de al menos una de las condiciones en todo momento.
    - Win NT/2000 previenen la espera circular sobre objetos Mutex.
- Detección y Recuperación:.
  - Permitir que el sistema entre en estado de interbloqueo, detectarlo y realizar una recuperación.
- Ignorar el problema y pretender que los interbloqueos nunca ocurren el sistema.
  - Utilizado por sistemas UNIX.

Prevención de Interbloqueo

- Métodos Indirectos
  - Impedir la aparición de alguna de las primeras tres condiciones
- Métodos Directos
  - Evitar la aparición de la cuarta condición

Prevención de Interbloqueo

- Métodos Indirectos
  - Exclusión Mutua. No puede anularse ya que se deben evitar inconsistencia en datos y recursos, ya que se debe asegurar que un recurso será utilizado sólo por un proceso a la vez.
  - Retención y Espera
    - Se debe tratar de impedir que un proceso solicite algunos recursos mientras posee otros. Hay dos formas de hacer esto:
      - Exigir que todos los procesos soliciten todos los recursos que requieren cuando sean creados
      - Exigir que un proceso previamente libere todos los recursos que tiene asignados antes de solicitar recursos nuevos.

Prevención de Interbloqueo

- Métodos Indirectos
  - No apropiación
    - Si un proceso que retiene ciertos recursos se le deniega una nueva solicitud, dicho proceso deberá liberar sus recursos anteriores y solicitarlos de nuevo cuando lo requiera junto con la nueva solicitud.
    - Si un proceso solicita varios recursos, primero se verifica si están disponibles. Si lo están se asignan. Si no lo están y actualmente están retenidos por otro proceso que espera por recursos adicionales, el SO puede expulsar al segundo proceso y exigirle que libere sus recursos o simplemente despojarlo de los recursos requeridos por el primer proceso
      - Un proceso solo podrá reiniciarse cuando se le asignan los nuevos recursos que ha solicitado y recupera cualquier recurso que se le haya despojado mientras esperaba
    - Sólo se evita el interbloqueo si hay 2 procesos que posean la misma prioridad. Técnica solo aplicable a recursos cuyo estado pueda salvarse y restaurarse luego fácilmente. (Ejemplo: Registros del CPU y espacio en memoria)

Prevención de Interbloqueo

- Métodos Directo
  - Espera Circular. Imponer un orden lineal total a todos los recursos del sistema; y obligar a que todas las solicitudes de recursos en el sistema se realicen en orden creciente o decreciente.