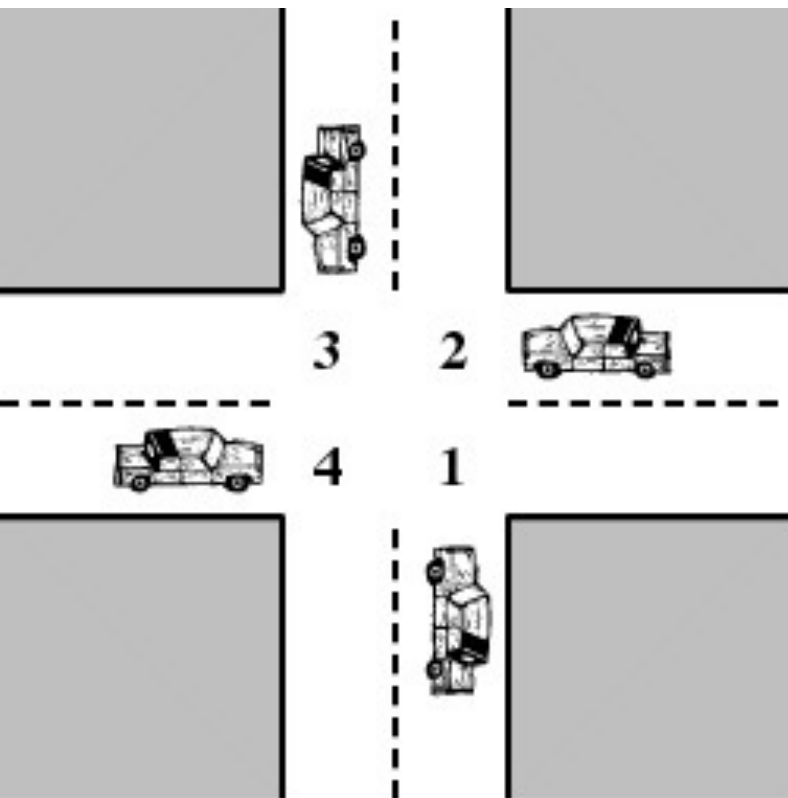


Capítulo 6

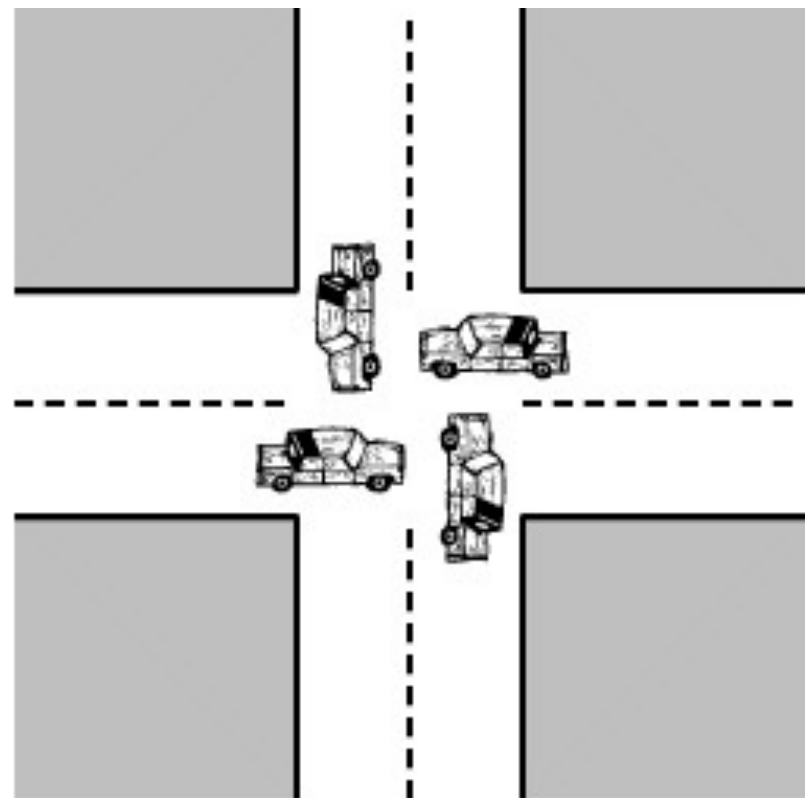
Concurrencia: interbloqueo e inanición

Interbloqueo

- Bloqueo permanente de un conjunto de procesos que compiten por los recursos o bien se comunican unos con otros.
- No existe una solución eficiente.
- Suponen necesidades contradictorias de recursos por parte de dos o más procesos.



(a) Posible interbloqueo



(b) Interbloqueo

Figura 6.1. Representación del interbloqueo.

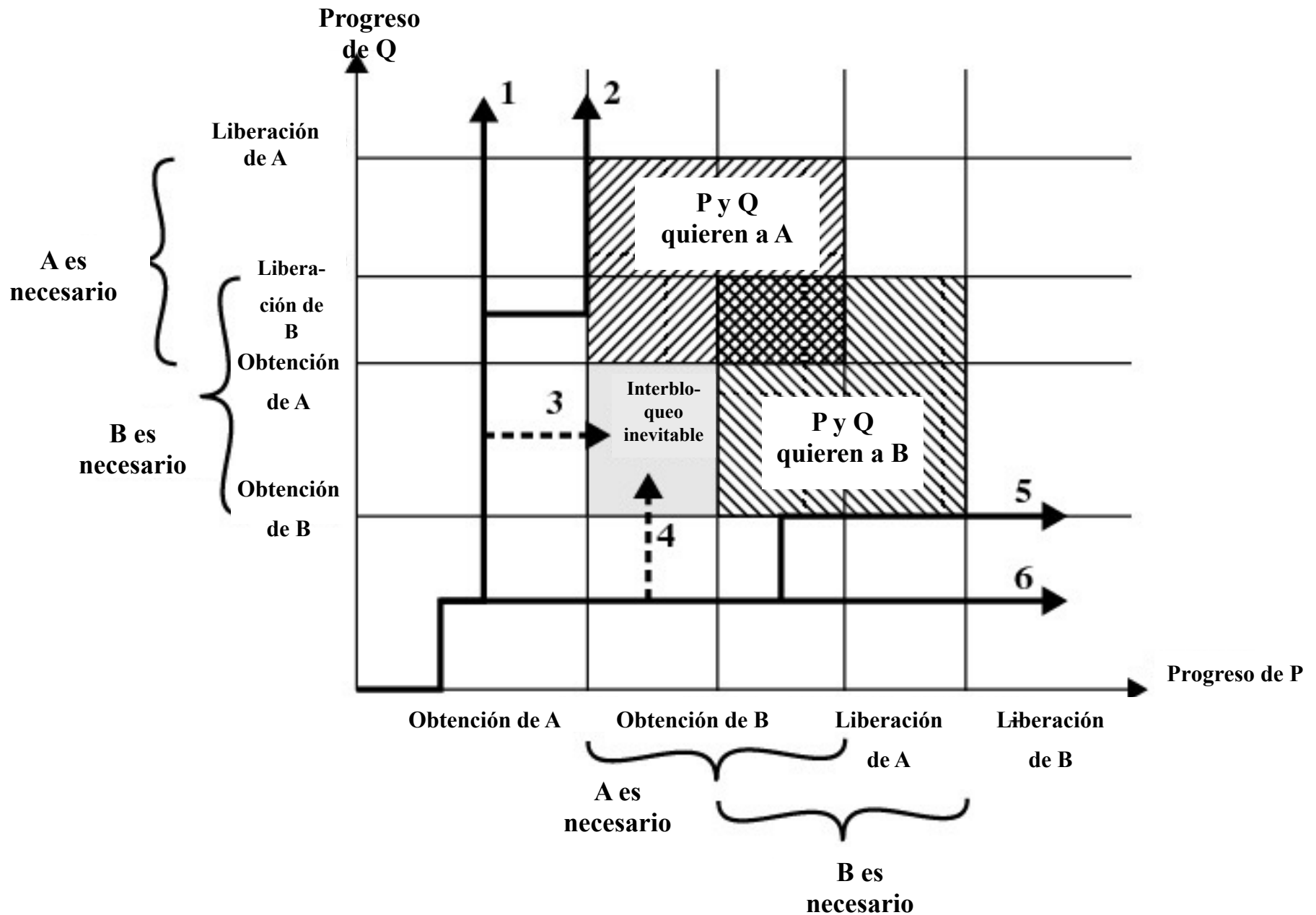


Figura 6.2. Ejemplo de interbloqueo [BACO98].

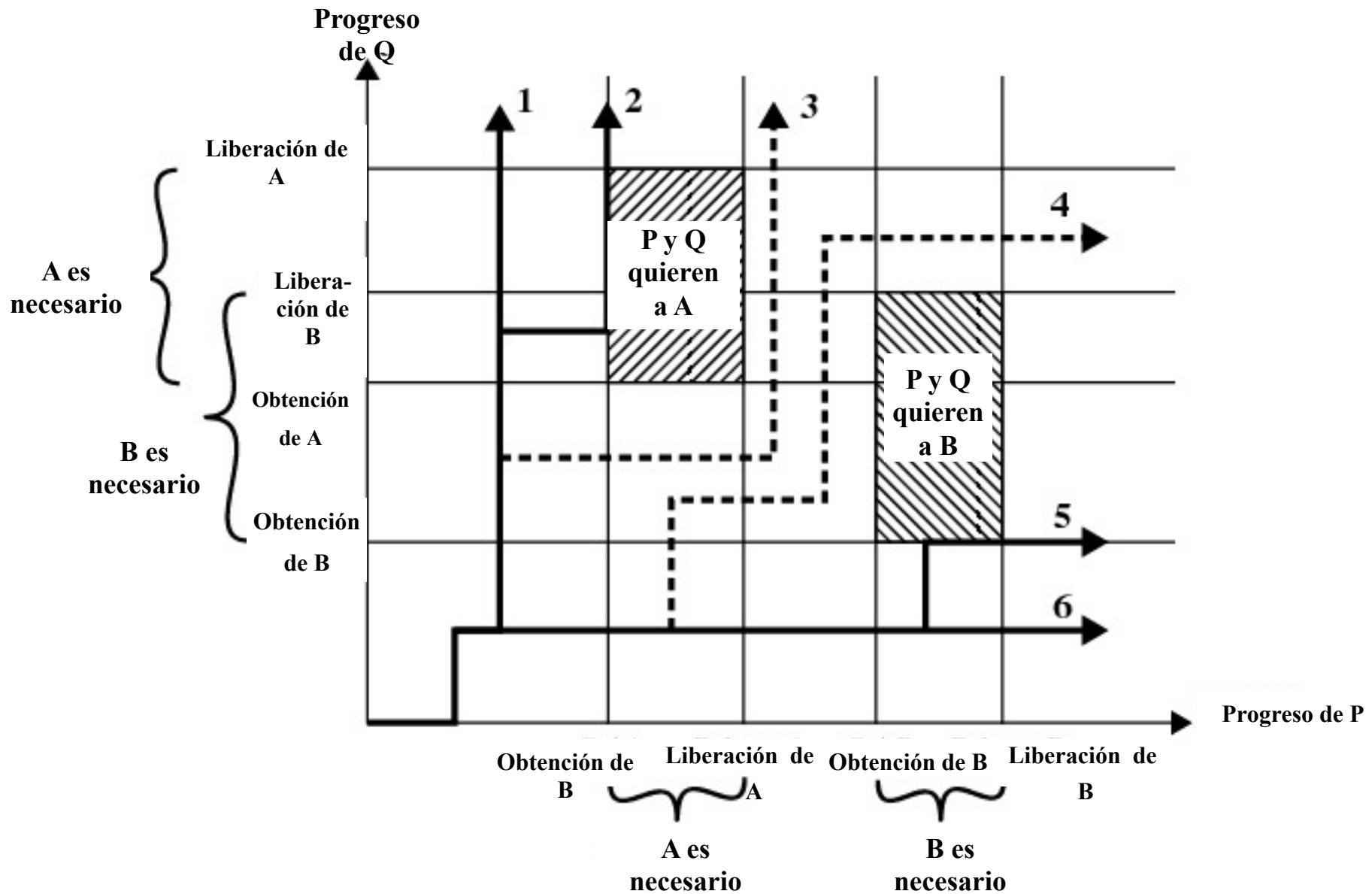


Figura 6.3. Ejemplo de sin interbloqueo [BACO98].



Recursos reutilizables

- Pueden ser usados por un proceso y no se agotan con el uso.
- Los procesos obtienen unidades de recursos que liberan posteriormente para que otros procesos las reutilicen.
- Procesadores, canales de E/S, memoria principal y secundaria, archivos, bases de datos y semáforos.
- El interbloqueo se produce si cada proceso retiene un recurso y solicita el otro.

Ejemplo de interbloqueo

Proceso P

Paso	Acción
p_0	Solicitar (D)
p_1	Bloquear (D)
p_2	Solicitar (T)
p_3	Bloquear (T)
p_4	Realizar función
p_5	Desbloquear (D)
p_6	Desbloquear (T)

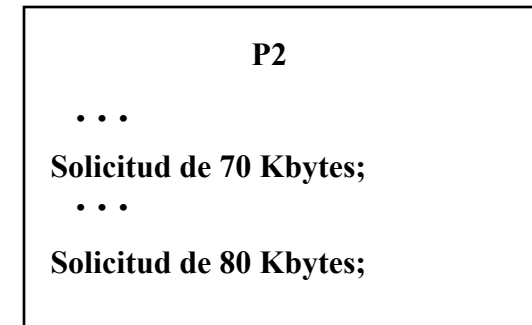
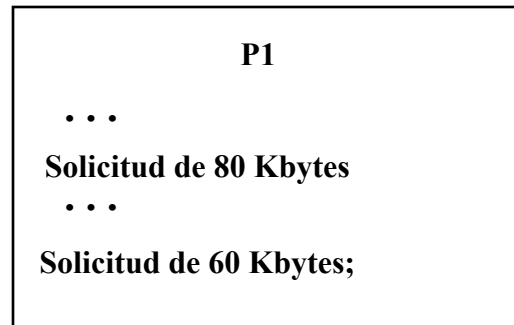
Proceso Q

Paso	Acción
q_0	Solicitar (T)
q_1	Bloquear (T)
q_2	Solicitar (D)
q_3	Bloquear (D)
q_4	Realizar función
q_5	Desbloquear (T)
q_6	Desbloquear (D)

Figura 6.4. Ejemplo de dos procesos compitiendo por recursos reutilizables.

Otro ejemplo de interbloqueo

- El espacio disponible es de 200 KB y se origina la siguiente secuencia de peticiones:



- Se produce un interbloqueo si ambos procesos avanzan hasta su segunda petición.

Recursos consumibles

- Puede ser creado (producido) y destruido (consumido) por un proceso.
- Interrupciones, señales, mensajes e información en buffers de E/S.
- El interbloqueo se produce si el *Receive* es bloqueante.
- Puede darse una combinación de sucesos poco habitual que origine el interbloqueo.

Ejemplo de interbloqueo

- El interbloqueo se produce si el *Receive* es bloqueante.

P1

...

Receive (P2);

...

Send (P2, M1);

P2

...

Receive (P1);

...

Send (P1, M2);

Condiciones de interbloqueo

- Exclusión mutua:
 - Sólo un proceso puede usar un recurso cada vez.
- Retención y esperar:
 - Un proceso solicita todos los recursos que necesita a un mismo tiempo.

Condiciones de interbloqueo

- No apropiación:
 - Si a un proceso que retiene ciertos recursos se le deniega una nueva solicitud, dicho proceso deberá liberar sus recursos anteriores.
 - Si un proceso solicita un recurso que actualmente está retenido por otro proceso, el sistema operativo puede retener el segundo proceso y exigirle que libere sus recursos.

Condiciones de interbloqueo

- Círculo vicioso de espera:
 - Puede prevenirse definiendo una ordenación lineal de los tipos de recursos.

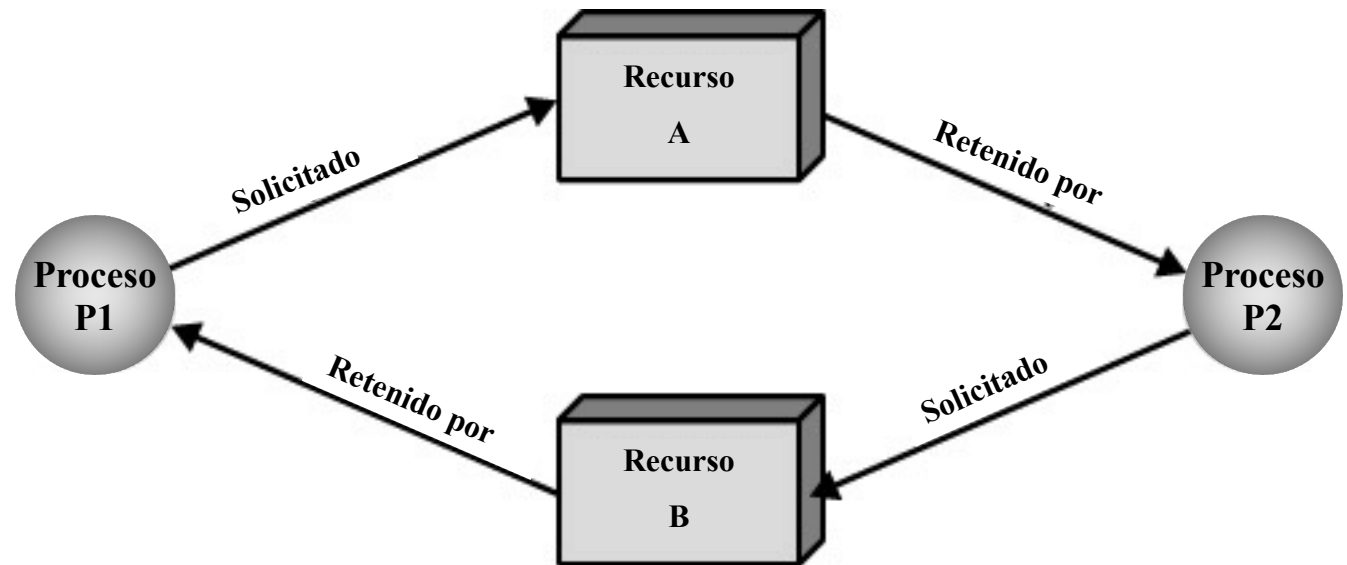


Figura 6.5. Círculo vicioso de espera.



Predicción del interbloqueo

- Se decide dinámicamente si la petición actual de asignación de un recurso podría, de concederse, llevar potencialmente a un interbloqueo.
- Necesita conocer las peticiones futuras de recursos.



Dos enfoques para la predicción del interbloqueo

- No iniciar un proceso si sus demandas pueden llevar a interbloqueo.
- No conceder una solicitud de incrementar los recursos de un proceso si esta asignación puede llevar a interbloqueo.

Negativa de asignación de recursos

- Denominada algoritmo del banquero.
- El estado del sistema es la asignación actual de recursos a los procesos.
- Un estado seguro es un estado en el cual existe al menos una secuencia que no lleva al interbloqueo.
- Un estado inseguro es un estado que no es seguro.

Determinación de un estado seguro: estado inicial

	R1	R2	R3
P1	3	2	2
P2	6	1	3
P3	3	1	4
P4	4	2	2

Matriz demanda

	R1	R2	R3
P1	1	0	0
P2	6	1	2
P3	2	1	1
P4	0	0	2

Matriz asignación

R1	R2	R3
9	3	6

Vector recursos

R1	R2	R3
0	1	1

Vector disponible

(a) Estado inicial

Determinación de un estado seguro: P2 terminado

	R1	R2	R3
P1	3	2	2
P2	0	0	0
P3	3	1	4
P4	4	2	2

Matriz demanda

	R1	R2	R3
P1	1	0	0
P2	0	0	0
P3	2	1	1
P4	0	0	2

Matriz asignación

R1	R2	R3
6	2	3

· **Vector disponible**

(b) P2 terminado

Determinación de un estado seguro: P1 terminado

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	3	1	4
P4	4	2	2

Matriz demanda

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	2	1	1
P4	0	0	2

Matriz asignación

R1	R2	R3
7	2	3

Vector disponible

(c) P1 terminado

Determinación de un estado seguro: P3 terminado

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	0	0	0
P4	4	2	2

Matriz demanda

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	0	0	0
P4	0	0	2

Matriz asignación

R1	R2	R3
9	3	4

Vector disponible

(d) P3 terminado

Determinación de un estado inseguro

	R1	R2	R3
P1	3	2	2
P2	6	1	3
P3	3	1	4
P4	4	2	2

Matriz demanda

	R1	R2	R3
P1	1	0	0
P2	6	1	2
P3	2	1	1
P4	0	0	2

Matriz asignación

R1	R2	R3
9	3	6

Vector recursos

R1	R2	R3
0	1	1

Vector disponible

(a) Estado inicial

Determinación de un estado inseguro

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	3	1	4
P4	4	2	2

Matriz demanda

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	2	1	1
P4	0	0	2

Matriz asignación

R1	R2	R3
7	2	3

Vector disponible

(b) P1 solicita una unidad de R1 y otra de R3

Predicción del interbloqueo

- Se debe presentar la máxima demanda de recursos por anticipado.
- Los procesos a considerar deben ser independientes, no hay condiciones de sincronización.
- Debe haber un número fijo de recursos a repartir.
- Los procesos no pueden finalizar mientras retengan recursos.

Detección del interbloqueo

	R1	R2	R3	R4	R5
P1	0	1	0	0	1
P2	0	0	1	0	1
P3	0	0	0	0	1
P4	1	0	1	0	1

Matriz Solicitud Q

	R1	R2	R3	R4	R5
P1	1	0	1	1	0
P2	1	1	0	0	0
P3	0	0	0	1	0
P4	0	0	0	0	0

Matriz asignación A

R1	R2	R3	R4	R5
2	1	1	2	1

Vector recursos

R1	R2	R3	R4	R5
0	0	0	0	1

Vector disponible

Figura 6.9. Ejemplo de detección de interbloqueo.

Técnicas una vez detectado el interbloqueo

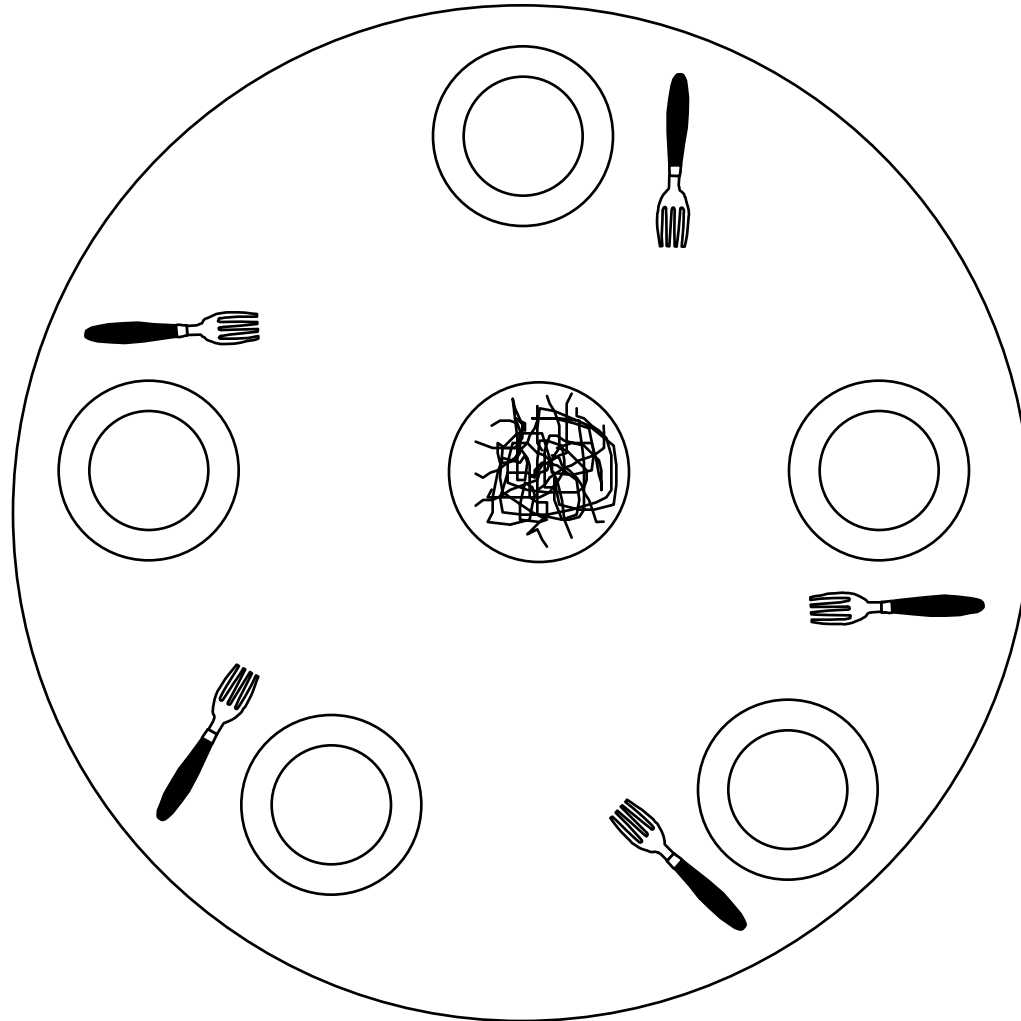
- Abortar todos los procesos interbloqueados.
- Retroceder cada proceso interbloqueado hasta algún punto de control definido previamente y volver a ejecutar todos los procesos:
 - Puede repetirse el interbloqueo original.
- Abortar sucesivamente procesos interbloqueados hasta que deje de haber interbloqueo.
- Apropiarse de recursos sucesivamente hasta que deje de haber interbloqueo.



Criterio de selección de los procesos interbloqueados

- La menor cantidad de tiempo de procesador consumido hasta ahora.
- El menor número de líneas de salida producidas hasta ahora.
- El mayor tiempo restante estimado.
- El menor número total de recursos asignados hasta ahora.
- La prioridad más baja.


El problema de la cena de los filósofos





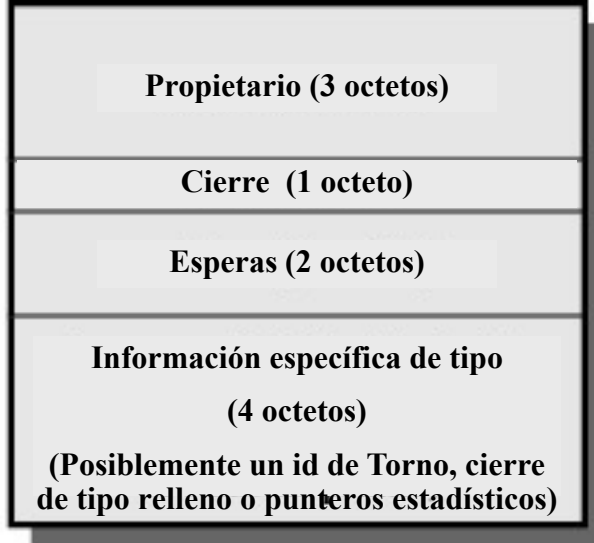
Mecanismos de concurrencia en UNIX

- Tubos (*pipes*).
- Mensajes.
- Memoria compartida.
- Semáforos.
- Señales.

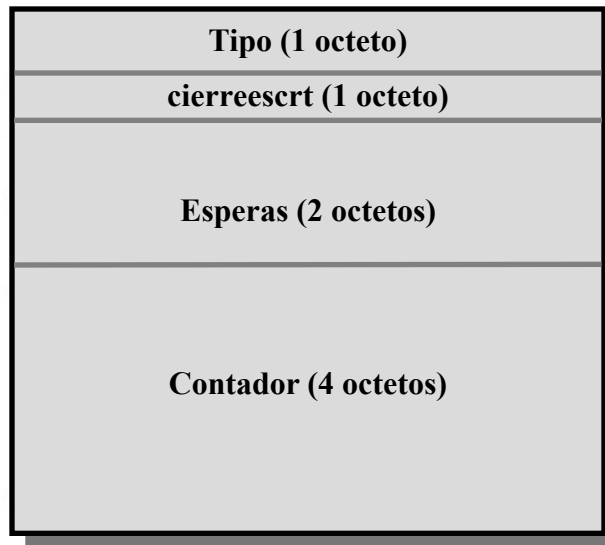


Primitivas de sincronización de hilos en Solaris

- Cierres de exclusión mutua (*mutex*).
- Semáforos.
- Cierres de múltiples lectores, un escritor (lectores/escritores).
- Variables de condición.

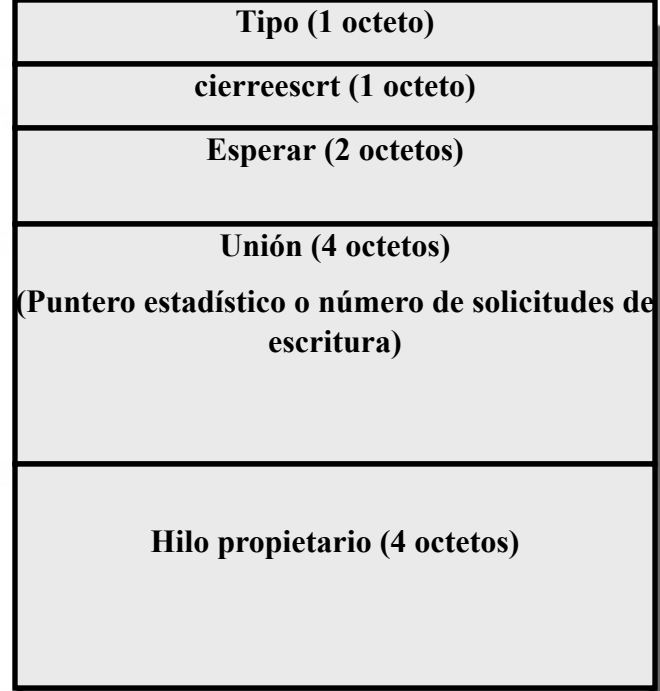


(a) Cierre MUTEX

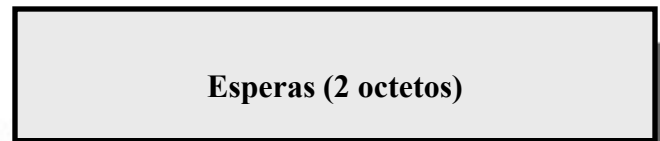


(b) Semáforo

Figura 6.13. Estructuras de datos de sincronización en Solaris.



(c) Cierre de lectores/escritores



(d) Variable de condición

Figura 6.13. Estructuras de datos de sincronización en Solaris.

Mecanismos de concurrencia en Windows 2000

- Proceso.
- Hilo.
- Archivo.
- Entrada de consola.
- Notificación de cambio de archivo.
- Mutante.
- Semáforo.
- Suceso.
- Temporizador.