Interbloqueig

Unitat 4 - Nel Banqué Torné

---

1. Introducció

Els sistemes informàtics sovint gestionen diversos recursos simultàniament. Aquest fet pot portar problemes quan diversos processos competeixen pel mateix recurs. El sistema operatiu té com a funció principal gestionar aquests recursos i evitar conflictes, assegurant-ne un ús coordinat. Tot i així, poden aparèixer situacions de bloqueig mutu que dificulten el funcionament normal.

---

2. Problemes Amb Recursos Compartits

Exemple 1: Creació d'un fitxer

Un procés que crea un fitxer necessita accés exclusiu a la **taula del sistema de fitxers**. Aquest mecanisme garanteix que no es corrompi la informació. Si dos processos intenten accedir-hi alhora, un d'ells serà bloquejat fins que l'altre hagi acabat.

Exemple 2: Situació d'interbloqueig

- Procés A obté accés a un escàner i sol·licita la memòria USB.
- Procés B obté accés a la memòria USB i després sol·licita l'escàner.
  Conseqüència: Cap dels dos pot avançar, ja que estan esperant que l'altre alliberi el recurs.

---

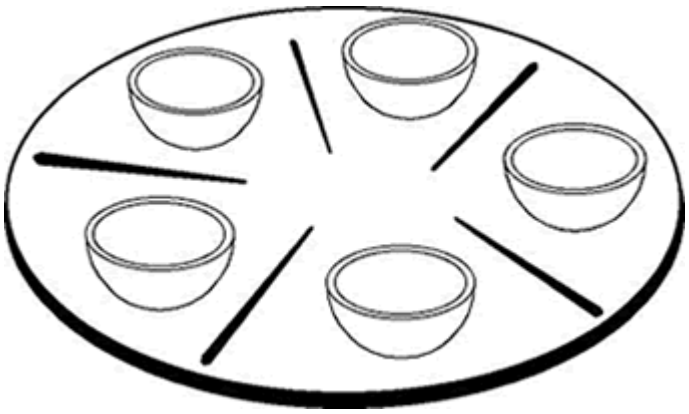
3. Definició i Causes

Definició d'Interbloqueig

Un interbloqueig es produeix quan dos o més processos no poden avançar perquè cadascun d'ells espera que l'altre alliberi un recurs.

Exemple Clàssic: Problema dels Filòsofs

En una taula amb cinc filòsofs, cadascun necessita un ganivet i una forquilla per menjar. Si tots agafen un ganivet i esperen una forquilla, cap podrà completar la seva acció.



---

4. Tipus de Recursos

- **Apropiatius:** Aquests recursos es poden retirar d'un procés sense que això suposi cap problema. Ex.: La memòria en sistemes amb **swapping** o **paging**.
- **No apropiatius:** No poden ser retirats fins que el procés en qüestió els alliberi. Ex.: Una impressora o un escàner.

Gariebé totes les crides a sistemes soliciten un recurs (open, close, fork, etc).

---

5. Condicions Necessàries (Coffman et al.)

1. **Exclusió Mútua:** Cada recurs només pot ser usat per un procés a la vegada.
2. **Retenció i Espera:** Els processos mantenen els recursos que tenen assignats mentre en sol·liciten de nous.
3. **No Apropiació:** Els recursos no poden ser retirats per força; han de ser alliberats pel procés que els utilitza.
4. **Espera Circular:** Hi ha una cadena de processos on cadascun espera un recurs que està en mans del següent procés de la cadena.

Si totes aquestes condicions es compleixen alhora, es pot generar un interbloqueig.

---

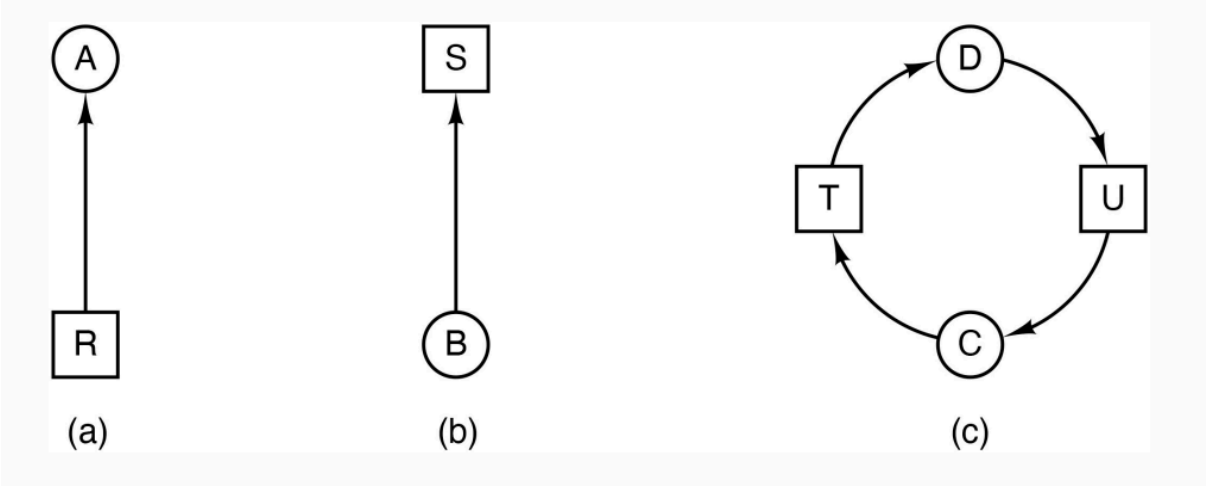
6. Grafs d'assignació

Graf d'Assignació de Recursos

Definirem les parts del graf de manera que:

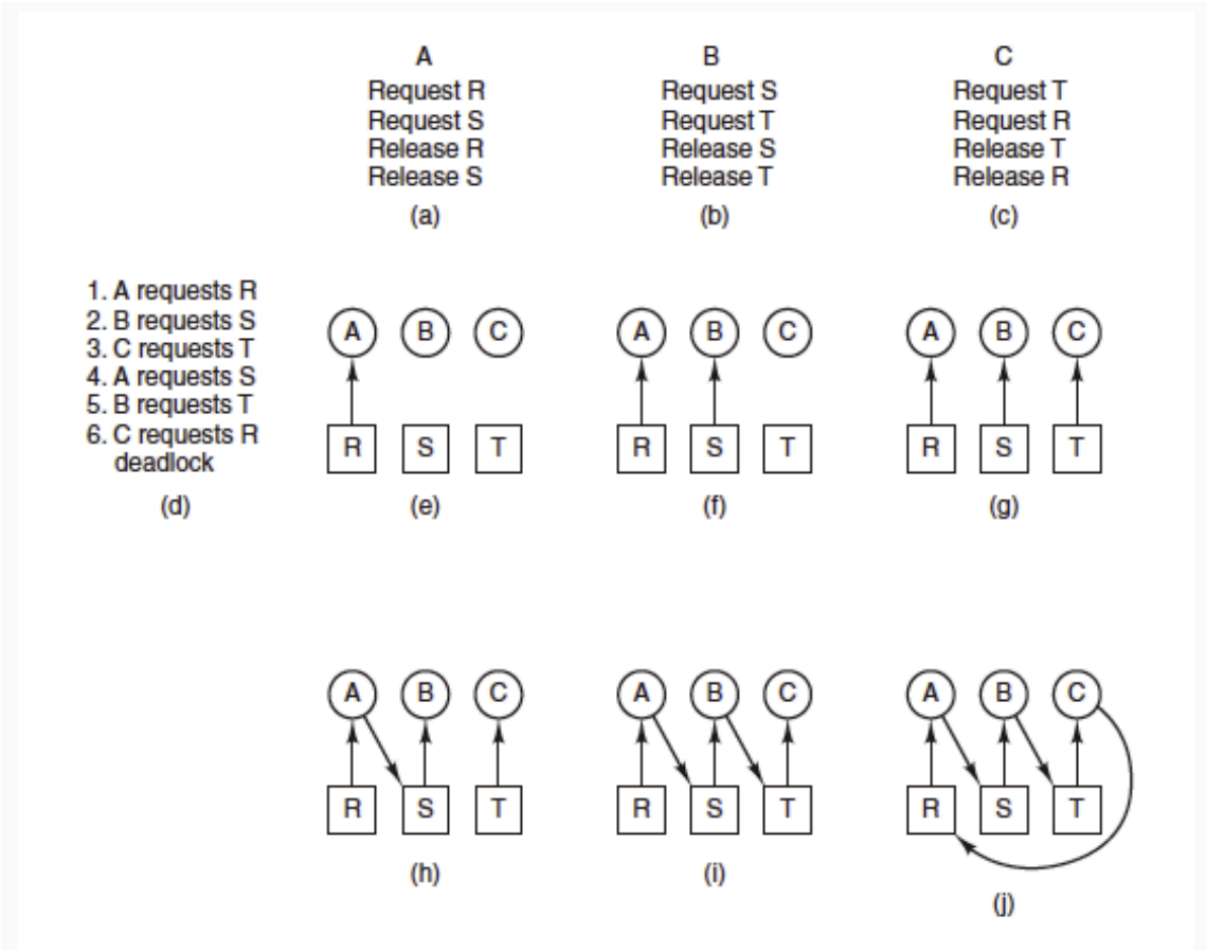
- Nodes: Processos i recursos.
- Arestes: Peticions o assignacions de recursos.

Un graf d'assignació de recursos és un graf dirigit que representa les relacions entre els processos i els recursos en un sistema:

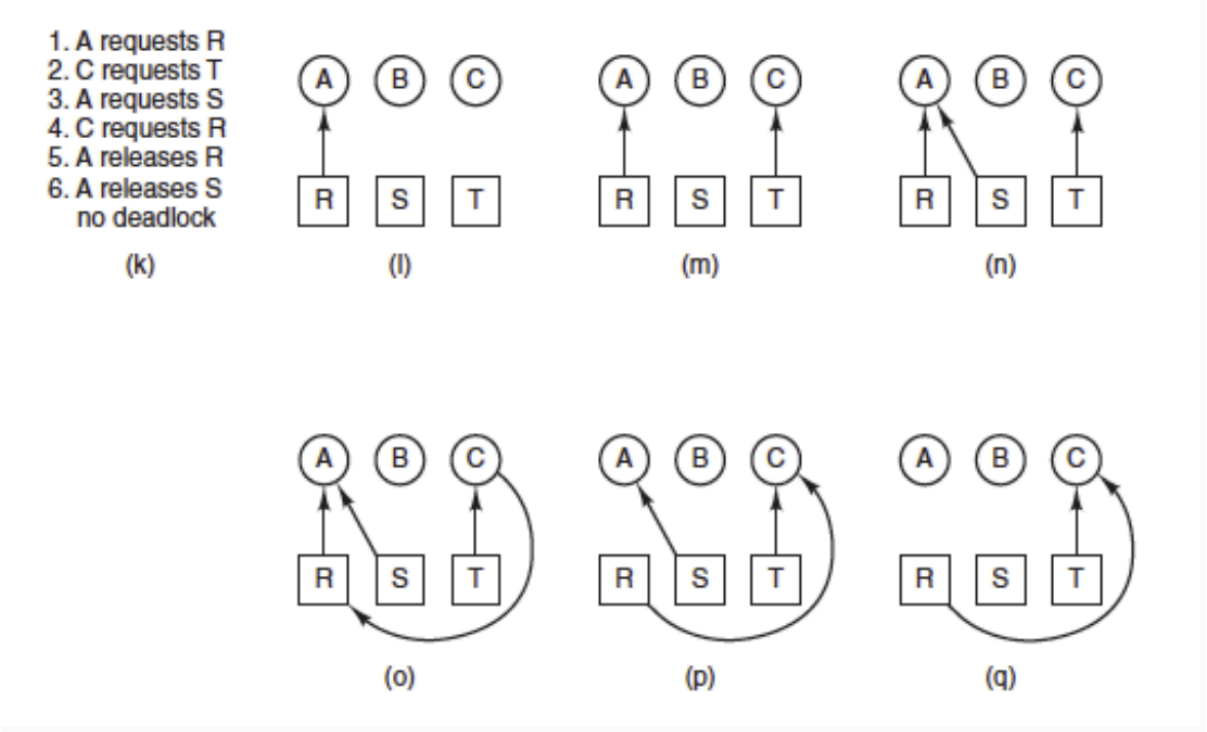


Important: Un cicle en un graf és indicatiu d'un interbloqueig.

Exemples de modelització de recursos (Causa Interbloqueig)



Exemples de modelització de recursos (Evitar interbloqueig)



---

7. Estratègies per Gestionar Interbloquejos

1. Ignorar el Problema

En alguns sistemes, especialment els senzills o amb poques interaccions entre processos, els interbloquejos són rars. En aquests casos, pot ser més pràctic ignorar-los.

- **Exemples:** Sistemes on els recursos són poc freqüentment compartits o amb pocs processos en execució simultània.
- **Desavantatge:** Si es produeix un interbloqueig, el sistema pot quedar inutilitzable fins que es reinicii manualment.

---

2. Detecció i Recuperació

Detecció

La detecció consisteix a identificar cicles en els grafs d'assignació de recursos. Per això es poden utilitzar algorismes dissenyats per trobar patrons que indiquen interbloquejos:

Graf d'assignació de recursos: Representa processos com nodes i peticions/assignacions de recursos com arestes. Un cicle en aquest graf implica un possible interbloqueig.

Algorisme de detecció de cicles: Es basa en explorar els nodes i arcs del graf. Quan un node es troba més d'una vegada en la mateixa ruta, es detecta un cicle. Aquí hi ha un exemple amb explicació:

$$A = \begin{bmatrix} 2 & 1 & 0 & 0 \end{bmatrix} \; E = \begin{bmatrix} 4 & 2 & 3 & 1 \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ 0 & 1 & 2 & 0 \end{bmatrix} \qquad R = \begin{bmatrix} 2 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 \end{bmatrix}$$

- Pas 1: Buscar un procés que es pugui satisfer totes les seves sol·licituts.
  - $P1$ no es pot satisfer ja que sol·licita 1 instancia de $R4$ i no ni ha cap disponible.
  - $P2$ no es pot satifer ja que sol·licita 1 instancia de $R3$ i no ni ha cap disponible.
  - $P3$ es pot satisfer ja que sol·licita 2 instancies de $R1$ i 1 de $R2$ i aquestes estan disponibles a A.
- Pas 2: Sumar la fila i-èssima de C a A.
  - $A = A + C_3 = [2, 1, 0, 0] + [0, 1, 2, 0] = [2, 2, 2, 0]$
  - Marquem el procés $P3$. En aquest moment, $P3$ pot finalitzar i alliberar els recursos que té assignats. $A = [4, 3, 2, 0]$

Amb aquests recursos alliberats, podem satisfer les sol·licituds de $P1$ i $P2$. Per tant no hi ha interbloqueig. Si $P3$ hagués sol·licitat 1 instacia de $R4$ llavors no es podria satisfer cap sol·licitud i tindríem un interbloqueig.

Vector disponible -> A

Vector total -> E

Matriu Nec -> R

Recuperació

Una vegada detectat un interbloqueig, cal alliberar els recursos bloquejats. Les opcions més comunes són:

1. **Finalitzar processos:** Acabar forçosament un o més processos involucrats en l'interbloqueig.
2. **Reordenar prioritats:** Assignar recursos segons un ordre diferent que permeti desbloquejar el sistema, podent dependre de la importància dels processos.
3. **Alliberar recursos parcialment:** Permetre als processos retornar alguns recursos sense finalitzar-los.

---

3. Prevenció

La prevenció consisteix a dissenyar el sistema de manera que les condicions necessàries per a un interbloqueig no es puguin complir simultàniament. Això es basa en les quatre condicions de Coffman:

1. **Exclusió mútua:**
    - Es poden utilitzar tècniques com el **spooling**, on es crea una cua per gestionar recursos compartits de manera controlada.
    - Ex.: En comptes de permetre l'ús exclusiu d'una impressora, s'enquadernen les tasques i s'imprimeixen seqüencialment.
2. **Espera circular:**
    - Assignar recursos seguint un ordre estricte.
    - Ex.: Els processos han de sol·licitar recursos seguint un ordre jeràrquic predefinit.
3. **No apropiació:**
    - Permetre l'expropiació de recursos crítics. Això implica retirar recursos d'un procés sense el seu consentiment si és necessari per evitar un interbloqueig.
4. **Espera i manteniment:**
    - Evitar que un procés mantingui recursos mentre n'espera de nous. Una solució és obligar els processos a sol·licitar tots els recursos necessaris alhora.

---

8. Algorisme de Prevenció Dinàmica

Algorisme del Banquer

Aquest algorisme és una estratègia avançada que calcula l'impacte potencial de cada sol·licitud de recurs abans d'assignar-lo.

- **Funcionament:**
    - S'analitza si una petició mantindrà el sistema en un **estat segur**.
    - Si el sistema continua segur, es concedeix el recurs. Si no, la petició es rebutja temporalment.
- **Avantatges:**
    - Redueix el risc d'interbloquejos.
    - Manté la flexibilitat en la gestió de recursos.
- **Limitacions:**
    - Requereix molta memòria i pot ser lent en sistemes amb molts processos.

Exemple per a un recurs:

	$Assignat$	Max
A	0	6
B	0	5
C	0	4
D	0	7

Si hi ha diponibles 10 recursos. **Estat segur**, tots poden finalitzar amb èxit.

	$Assignat$	Max
A	1	6
B	1	5
C	2	4
D	4	7

Si hi ha diponibles 2 recursos. **Estat segur**, C primer i després la resta.

	$Assignat$	Max
A	1	6
B	2	5
C	2	4
D	4	7

Si hi ha diponibles 1 recurs. **Estat insegur**, ningú pot acabar.

Exemple per a M recursos:

L'algorisme del banquer es pot generalitzar per a sistemes amb múltiples recursos de cada tipus. En aquest cas, el sistema operatiu ha de mantenir una matriu A de mida $M \times N$ on M és el nombre de recursos i N és el nombre de processos. Els elements de la matriu A_{ij} indiquen el nombre d'instàncies del recurs R_i que el procés P_j té assignades.

1. Seleccioneu un procés P_i que no s'hagi finalitzat i que tingui una demanda màxima R_i que sigui menor o igual a A.
2. Marqueu el procés P_i i afegiu la fila i-èssima de C a A.
3. Repeteix els passos fins acabar en un (estat segur) o un (estat insegur).

---

9. Detecció d'Interbloquejos

Amb 1 Recurs per Tipus

En sistemes amb un únic recurs per tipus, es pot detectar un interbloqueig analitzant el graf d'assignació de recursos:

- **Graf d'esperes:** S'elimina la representació dels recursos i només es mantenen les connexions entre processos. Si es forma un cicle, hi ha un interbloqueig.
- **Exemple:**
  - Procés A té el recurs R i vol S.
  - Procés B té S i vol R.
  - Es forma un cicle entre A i B, indicant interbloqueig.

Amb M Recursos per Tipus

En sistemes més complexos, on hi ha múltiples instàncies de cada recurs:

- **Vectors i Matrius:**
  - **A (Disponibilitat):** Recursos lliures al sistema.

    - **C (Assignació):** Recursos assignats a cada procés.
    - **R (Peticions):** Recursos sol·licitats pels processos.
- **Algorisme:**
    - Verifica si un procés pot satisfer la seva petició R[i]≤AR[i] \leq AR[i]≤A.
    - Si es compleix, s'assignen els recursos i es recalcula AAA.
    - Si cap procés pot avançar, hi ha interbloqueig.
- **Exemple:**
    - Suposem tres processos i dues instàncies d'un recurs. Si els recursos assignats i les peticions impossibiliten satisfer cap procés, es detecta interbloqueig.