Tema 5. Gestió de la memòria

Què aprendràs

- Ordenar les memòries de l'ordinador per velocitat, preu i capacitat.
- Calcular l'adreça física d'un procés a partir de la base i l'adreça virtual.
- Explicar per què es perd memòria amb les particions, i distingir la fragmentació interna de l'externa.
- Calcular quantes pàgines necessita un procés i quanta memòria es perd a l'última.
- Traduir una adreça virtual a física amb una taula de pàgines.
- Explicar què és la memòria virtual i què passa en una fallada de pàgina.
- Veure quanta RAM i quant espai d'intercanvi fa servir el teu ordinador.

Currículum del mòdul 0222 (RD 1691/2007): RA2 a) · RA4 e).

La teva xuleta per a tot el tema

Tots els càlculs del tema surten d'aquestes quatre línies.

Què vols saber	Com es calcula
Adreça física	$\mathbf{d_F = d_B + d_V}$ (base més virtual)
Quantes pàgines	mida del procés ÷ mida de pàgina, **arrodonint cap amunt**
Memòria perduda a l'última pàgina	pàgines × mida de pàgina − mida del procés
Traduir amb pàgines	d_V ÷ mida de pàgina: el **quocient** és la pàgina i el **residu**, el desplaçament. Després, **marc × mida de pàgina + desplaçament**

Les mides de memòria es compten de 1.024 en 1.024, com vas veure al tema 3. Per això aquí parlem de **KiB** i **MiB**.

---

1. Per què s'ha de gestionar la memòria

Tens el navegador, la música i un editor oberts alhora. Tots tres processos viuen a la mateixa RAM, i no hi caben com vulguin. Algú ha de posar ordre, i aquest algú és el sistema operatiu.

La gestió de la memòria té tres feines:

Feina	Què vol dir
Repartir	Decidir quin tros de RAM li toca a cada procés
Protegir	Impedir que un procés llegeixi o escrigui la memòria d'un altre
Estirar	Fer que hi càpiguen més processos dels que cabrien de debò, fent servir el disc

Al tema 2 vas veure que el sistema operatiu fa de **repartidor** i de **guardià**. Aquí veuràs com ho fa amb la memòria.

---

2. No tota la memòria és igual

A l'ordinador hi ha diverses memòries, i cada una va a un ritme diferent.

Memòria	On és	Velocitat	Capacitat típica	En apagar
Memòria cau *(cache)*	Dins el processador	La més ràpida	De KiB a unes desenes de MiB	Es buida
RAM	A la placa base	Molt ràpida	De 8 a 32 GiB	Es buida
Disc (SSD o HDD)	A dins la caixa	Lenta	Centenars de GiB o més	Es manté

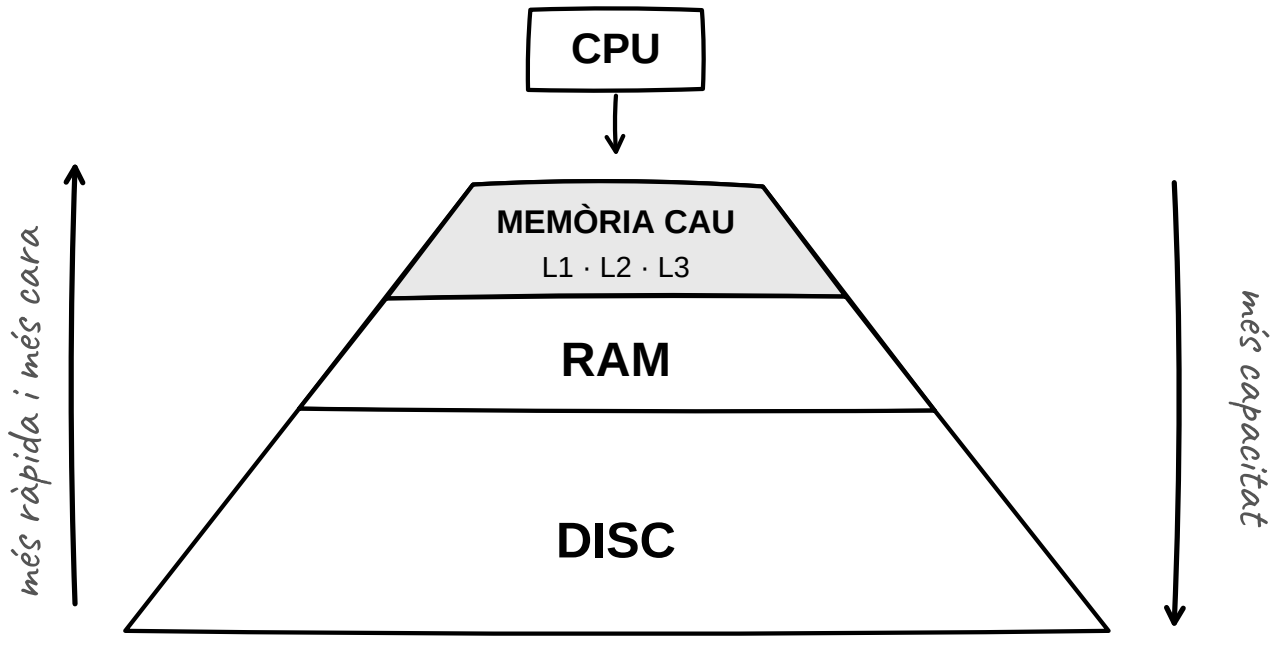

FIG 5.1 La jerarquia de memòria: com més a prop de la CPU, més ràpida i més petita.

La regla és aquesta: **com més ràpida és una memòria, més cara és i menys n'hi ha**. Per això no es fa tot l'ordinador amb memòria cau: costaria una fortuna i hi cabria molt poca cosa.

La **memòria cau** guarda les dades i instruccions que la CPU fa servir més sovint, perquè no hagi d'anar a buscar-les a la RAM cada vegada. N'hi ha de tres nivells, L1, L2 i L3. En els ordinadors d'avui, normalment tots tres van dins el mateix processador.

> **Recorda el tema 1.** La RAM és la taula de treball i el disc, l'armari. La memòria cau és el que tens a la mà mentre treballes.

Practica 5.1

a) Ordena les tres memòries de la més ràpida a la més lenta.

b) On és cada cosa?

Situació	Memòria
El document que estàs escrivint ara mateix	
La instrucció que la CPU repeteix milers de vegades per segon	
Una foto que no has obert des de fa un mes	

c) Si la memòria cau és la més ràpida, per què no es fa tota la memòria de l'ordinador amb memòria cau?

3. Adreces virtuals i adreces físiques

La RAM és una fila molt llarga de caselles, i cada casella té un número: la seva **adreça**.

Un programa no sap mai en quin lloc de la RAM el posaran. Per això treballa amb **adreces virtuals** (*virtual addresses*): compta des del seu propi començament, com si fos sol a la memòria. La primera casella del procés és la 0, la segona la 1, i així.

Quan el sistema operatiu carrega el procés, li troba un lloc a la RAM. On comença aquest lloc és l'**adreça base** (*base address*). L'adreça real a la RAM és l'**adreça física** (*physical address*), i es calcula així:

$\mathbf{d_F = d_B + d_V}$

- $\mathbf{d_B}$: adreça base, on comença el procés a la RAM.
- $\mathbf{d_V}$: adreça virtual, la que fa servir el programa.
- $\mathbf{d_F}$: adreça física, on és de debò.

Si et donen la física i la base i et demanen la virtual, és la mateixa fórmula girada: $\mathbf{d_V = d_F - d_B}$.

La traducció no la fa el programa ni l'has de fer tu mentre treballes: la fa la **MMU** (*Memory Management Unit*), un circuit que avui va dins el processador. La fa per a cada adreça, milions de vegades per segon.

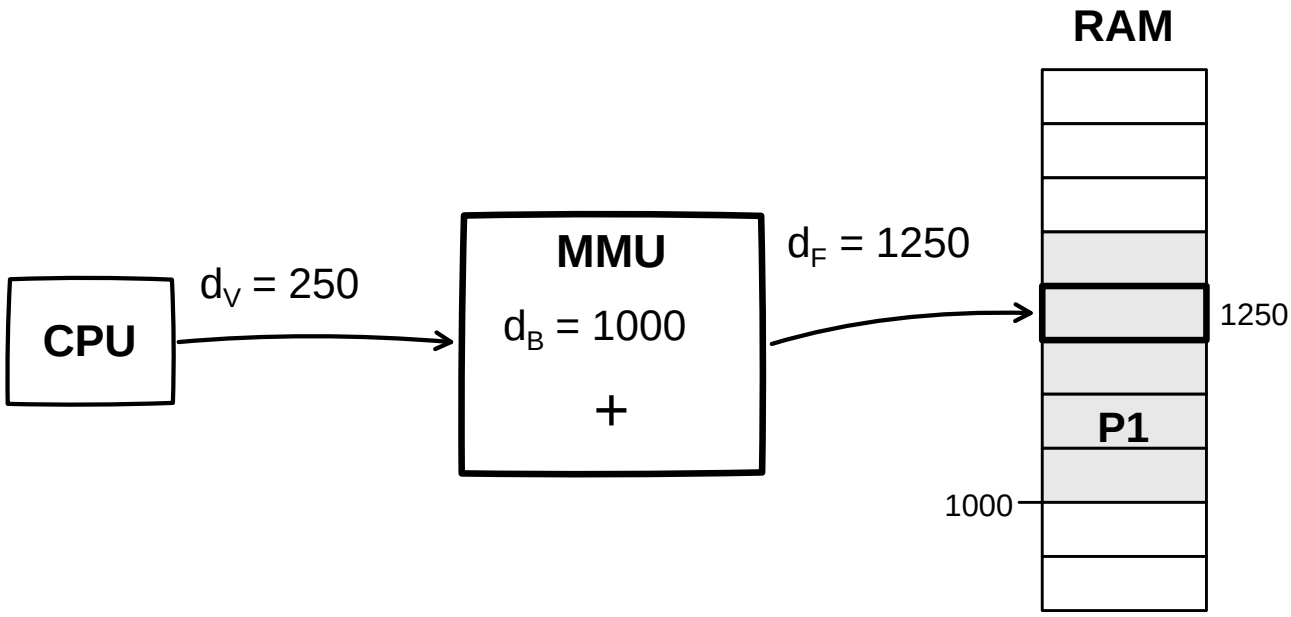

FIG 5.2 La MMU tradueix cada adreça virtual en una de física sumant-hi l'adreça base.

Exemple resolt 1 — De virtual a física

Un procés està carregat a partir de l'adreça **1000**. Vol llegir la seva adreça virtual **250**.

$$\mathbf{d_F = d_B + d_V = 1000 + 250 = 1250}$$

La casella que vol és la **1250** de la RAM.

> **Aquí és on la MMU protegeix.** Cada procés té una mida. Si demana una adreça virtual que surt del seu espai, la MMU no la tradueix, avisa el sistema operatiu i aquest tanca el programa. Així un procés no pot trepitjar la memòria d'un altre.

Guiat 5.2

Calcula l'adreça física. La primera ja està feta.

d_B	d_V	d_F
2000	300	2300
5000	45	
1200	800	
7350	650	

Practica 5.3

Un procés ocupa **1500** caselles i està carregat a partir de l'adreça **4000**. Les seves adreces virtuals, per tant, van de la 0 a la 1499.

d_V que demana	d_F
100	
1499	
0	
1600	

Per què la darrera no es pot traduir? Què fa la MMU?

4. Repartir la RAM en trossos seguits

La manera més antiga de repartir la memòria és donar a cada procés **un sol tros seguit** de RAM. Això es diu **assignació contigua** (*contiguous allocation*). Ha anat evolucionant així:

1. **Divisió simple.** Un tros per al sistema operatiu i la resta per a un únic procés. Només pot funcionar un programa a la vegada, com a MS-DOS.
2. **Particions estàtiques.** La memòria es parteix en trossos fixos, i a cada tros hi va un procés.
3. **Particions dinàmiques.** Cada procés rep just la memòria que necessita.

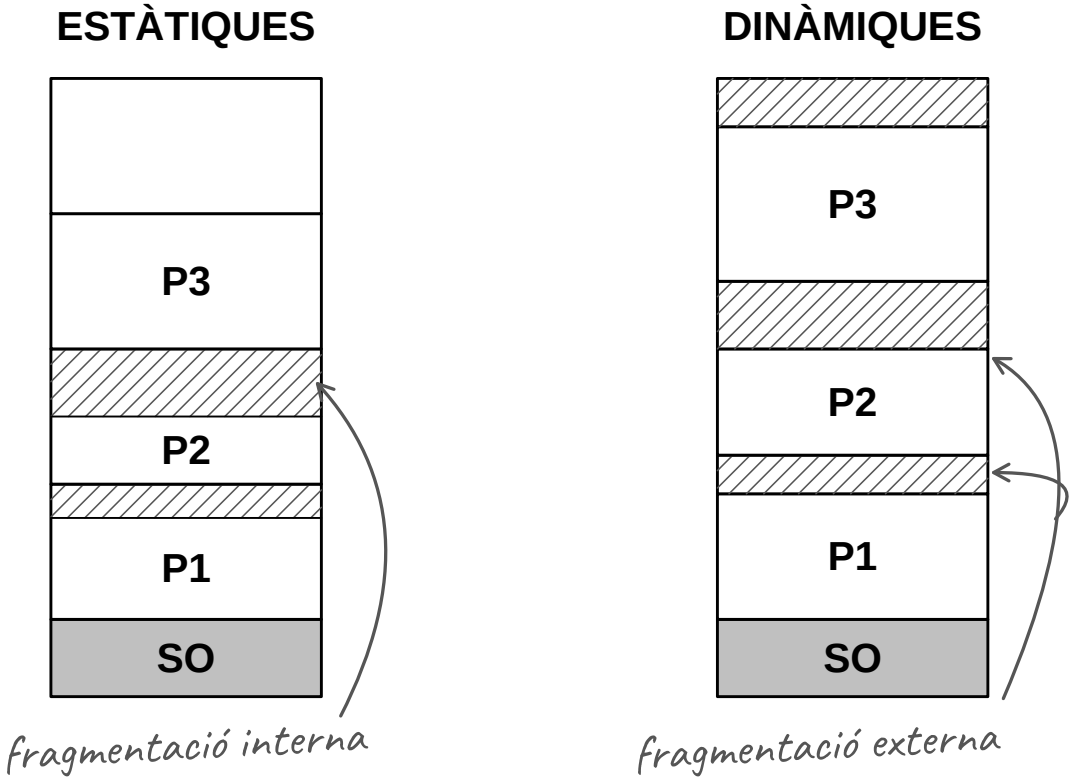

FIG 5.3 Particions estàtiques i dinàmiques. El ratllat és memòria perduda.

4.1. Particions estàtiques

El sistema operatiu parteix la memòria en **particions** (*partitions*) de mida fixa abans de començar. A cada partició hi cap **un sol procés**.

Tenen dos problemes:

- **Un procés més gran que la partició no hi cap enlloc**, encara que hi hagi molta memòria lliure.
- **Si el procés és més petit, el que sobra de la partició es perd**: ningú més no hi pot entrar. Aquesta memòria perduda **dins** la partició es diu **fragmentació interna** (*internal fragmentation*).

Exemple resolt 2 — Particions estàtiques

Hi ha **2 MiB** per als processos, partits en particions de **256 KiB**.

Pas 1. Quantes particions? 2 MiB són 2.048 KiB. 2.048 ÷ 256 = **8 particions**.

Pas 2. Arriba un procés de 180 KiB. Hi cap. Sobren 256 − 180 = **76 KiB** que ningú no podrà fer servir: fragmentació interna.

Pas 3. Arriba un procés de 300 KiB. És més gran que 256: **no hi cap a cap partició**.

Guiat 5.4

Hi ha **1.024 KiB** per als processos, partits en particions de **128 KiB**. Quantes particions hi ha? Omple la taula. La primera fila ja està feta.

Procés	Mida	Hi cap?	KiB perduts
A	100 KiB	Sí	28
B	128 KiB		
C	60 KiB		
D	200 KiB		

Quanta memòria es perd en total amb els processos que hi caben?

4.2. Particions dinàmiques

Per no perdre memòria dins les particions, el sistema operatiu dona a cada procés **exactament la memòria que necessita**, un procés darrere l'altre.

Funciona bé fins que els processos comencen a acabar. Cada un que acaba deixa un **forat**. Amb el temps, la memòria lliure queda repartida en molts forats petits. Hi pot haver prou memòria lliure en total, però **cap forat prou gran** per al procés que arriba. Aquesta memòria perduda **entre** els processos es diu **fragmentació externa** (*external fragmentation*).

La solució és la **compactació** (*compaction*): el sistema operatiu mou tots els processos cap a un costat perquè els forats s'ajuntin en un de sol. Funciona, però mentre la fa, l'ordinador no fa res més.

	Fragmentació interna	Fragmentació externa
On es perd	Dins la partició d'un procés	Entre els processos, en forats
Quan passa	Particions estàtiques	Particions dinàmiques
Com es resol	No es pot recuperar	Compactant

Exemple resolt 3 — Particions dinàmiques

La memòria està així, de baix a dalt:

Tros	Mida
Sistema operatiu	—
P1	200 KiB
Forat	**60 KiB**
P2	150 KiB
Forat	**90 KiB**
P3	100 KiB
Forat	**50 KiB**

Arriba **P4, de 120 KiB**.

Pas 1. Quanta memòria lliure hi ha? 60 + 90 + 50 = **200 KiB**. En total, P4 hi cabria.

Pas 2. Hi ha algun forat de 120 KiB o més? El més gran fa 90. **No hi cap.** És fragmentació externa.

Pas 3. Compactem. Els tres forats s'ajunten en un de 200 KiB. P4 hi entra, i en sobren 200 − 120 = **80 KiB**.

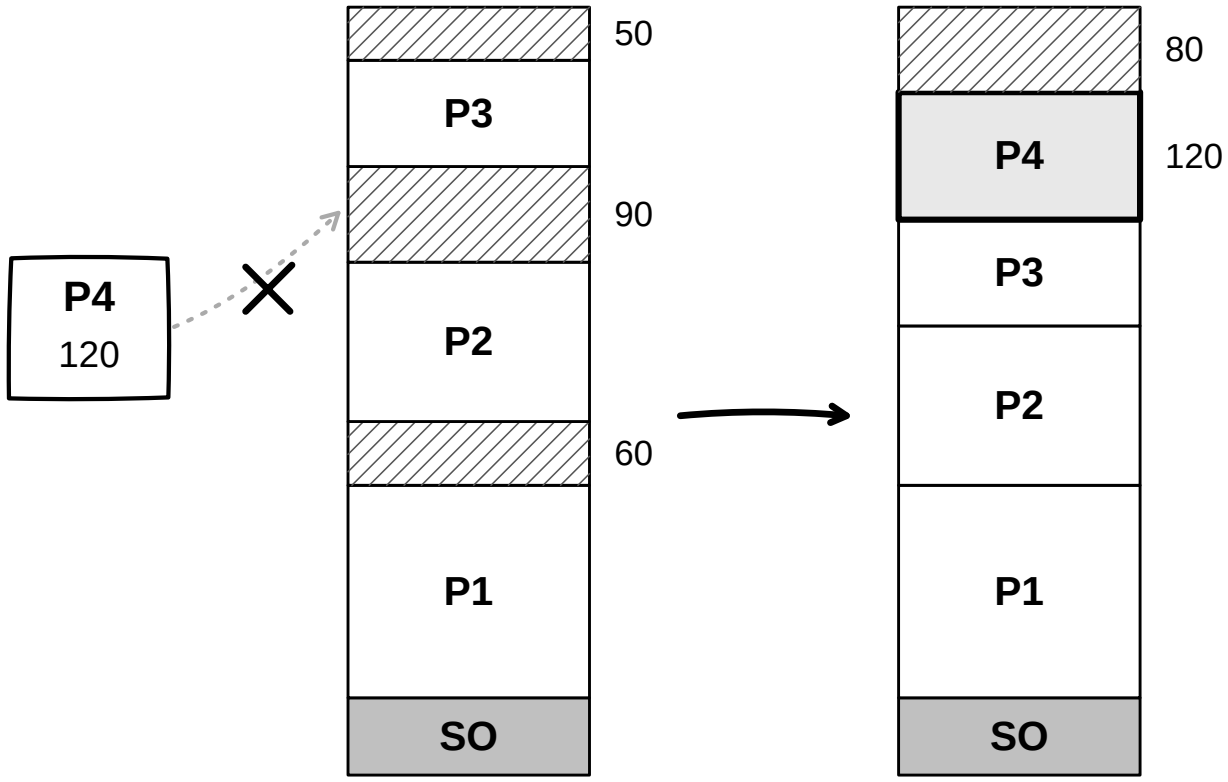

FIG 5.4 La mateixa memòria abans i després de compactar.

Practica 5.5

A la memòria hi ha quatre forats lliures, de **70**, **40**, **110** i **30 KiB**.

a) Quanta memòria lliure hi ha en total?

b) Arriba un procés de 100 KiB. Hi cap sense moure res? On?

c) Arriba un procés de 150 KiB. Hi cap? Per què?

d) Què hauria de fer el sistema operatiu perquè hi càpiga el de 150 KiB, i quanta memòria lliure en quedaria?

5. La paginació

Totes les maneres d'abans tenien el mateix problema: el procés havia d'anar **en un sol tros seguit**. La **paginació** (*paging*) ho resol trossejant-ho tot en peces petites i iguals.

- El **procés** es parteix en **pàgines** (*pages*), totes de la mateixa mida. Normalment, de **4 KiB**.
- La **RAM** es parteix en **marcs** (*frames*), de la mateixa mida que les pàgines.
- Cada pàgina va a **qualsevol marc lliure**. **No cal que vagin seguides.**
- Per saber on ha anat cada pàgina, el sistema operatiu guarda una **taula de pàgines** (*page table*) per a cada procés.

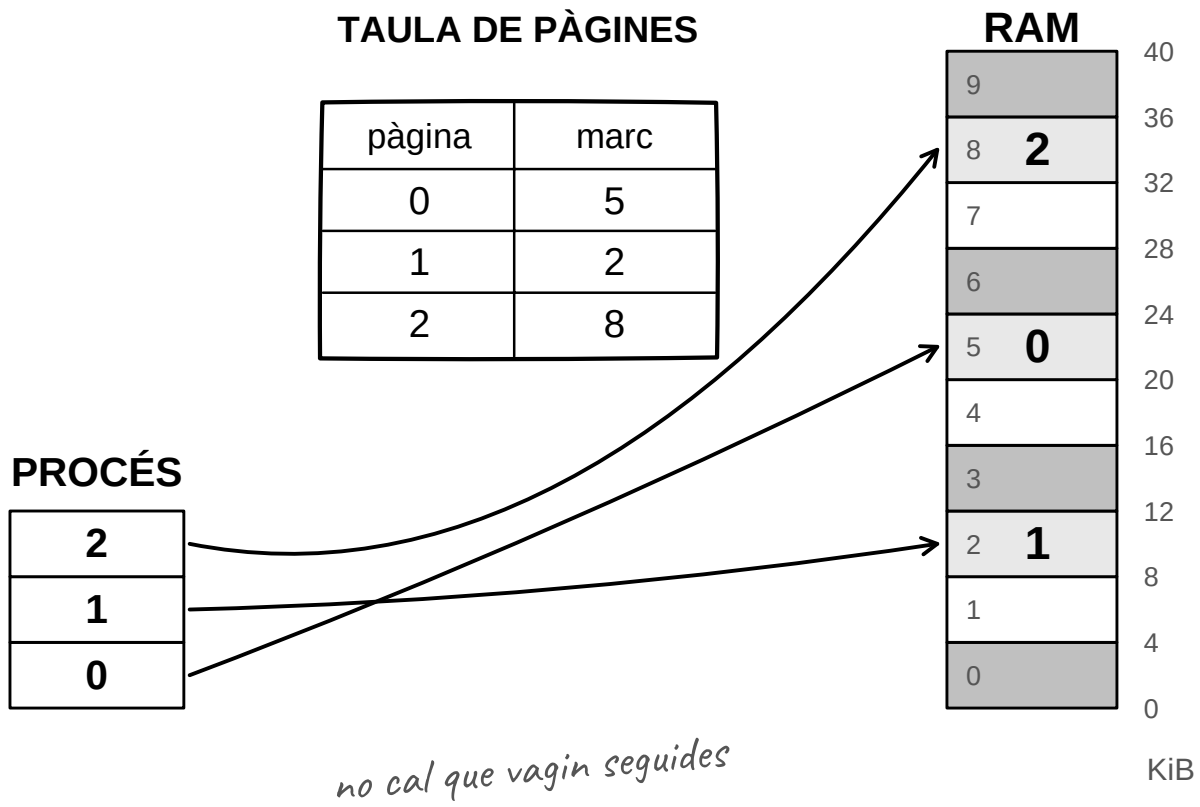

FIG 5.5 Les pàgines d'un procés repartides per marcs de la RAM, i la taula que les troba.

Què s'hi guanya:

- **No hi ha fragmentació externa.** Qualsevol marc lliure serveix, estigui on estigui.

- **La fragmentació interna és molt petita.** Només es pot perdre memòria a l'**última pàgina** del procés, i com a molt, menys d'una pàgina.

I què costa: cada procés necessita la seva taula de pàgines, i les taules també ocupen memòria.

> **La segmentació.** Hi ha una altra manera de trossejar un procés: en **segments** (*segments*) de mida diferent, un per a cada zona (el codi, les dades, la pila…). Els sistemes operatius d'avui fan servir sobretot la paginació.

Exemple resolt 4 — Quantes pàgines

Pàgines de **4 KiB**. Un procés ocupa **10 KiB**.

Pas 1. Quantes pàgines? 10 ÷ 4 = 2,5. No hi pot haver mitja pàgina: s'arrodoneix **cap amunt**. Calen **3 pàgines**.

Pas 2. Quanta memòria ocupen? 3 × 4 = **12 KiB**.

Pas 3. Quanta se'n perd? 12 − 10 = **2 KiB**, a l'última pàgina.

> **Comprova sempre.** La memòria perduda ha de ser **més petita que una pàgina**. Si et surt 4 KiB o més, t'has passat una pàgina.

Guiat 5.6

Pàgines de **4 KiB**. La primera fila ja està feta.

Mida del procés	Pàgines	KiB perduts
12 KiB	3	0
13 KiB		
7 KiB		
1 KiB		
20 KiB		

5.1. Traduir una adreça amb la taula de pàgines

Amb paginació, la MMU tradueix les adreces en tres passes:

1. **Divideix** l'adreça virtual entre la mida de pàgina. El **quocient** diu **a quina pàgina** és, i el **residu**, a **quina distància del principi de la pàgina**: el **desplaçament** (*offset*).
2. **Busca la pàgina a la taula** i en treu el marc.
3. **Calcula l'adreça física**: marc × mida de pàgina + desplaçament.

Si el procés demana una pàgina que no és seva, no la troba a la taula, i la MMU no la deixa passar.

Exemple resolt 5 — Traducció

Pàgines de **4 KiB**. La taula de pàgines del procés és aquesta:

Pàgina	Marc
0	5
1	2
2	8

El procés demana l'adreça virtual **9 KiB**.

Pas 1. 9 ÷ 4 = **2**, i en sobra **1**. És a la **pàgina 2**, amb desplaçament **1 KiB**.

Pas 2. A la taula, la pàgina 2 va al **marc 8**.

Pas 3. 8 × 4 + 1 = **33 KiB**. Aquesta és l'adreça física.

> **Comprova sempre.** El marc 8 va de 32 a 36 KiB. El resultat ha de caure dins d'aquest marc: 33 hi cau.

Practica 5.7

Pàgines de **4 KiB**. Aquest procés ocupa 4 pàgines i té aquesta taula:

Pàgina	Marc
0	5
1	2
2	8
3	0

d_V	Pàgina	Desplaçament	Marc	d_F
2 KiB				
5 KiB				
14 KiB				
11 KiB				

El procés demana l'adreça virtual **17 KiB**. Què passa?

6. La memòria virtual

Ara ve el truc més gran. Què passa si els processos oberts necessiten **més memòria de la que hi ha a la RAM**?

El sistema operatiu fa servir una part del disc com si fos RAM. Aquesta part del disc es diu **espai d'intercanvi** (*swap*). A Linux sol ser una partició o un fitxer de swap; a Windows és el **fitxer de paginació** (*pagefile*).

- A la **RAM** hi van les pàgines que es fan servir **ara**.
- A l'**espai d'intercanvi** hi van les pàgines que fa estona que ningú no toca.
- El sistema operatiu **les mou d'un lloc a l'altre tot sol**, sense que el programa se n'assabenti.

Tot junt, RAM i espai d'intercanvi, és la **memòria virtual** (*virtual memory*). Cada procés veu un espai de memòria gran i seguit, encara que per sota estigui repartit entre la RAM i el disc.

6.1. La fallada de pàgina

Quan un procés necessita una pàgina que és al disc, passa això:

1. El procés fa servir una adreça d'una pàgina que **no és a la RAM**.
2. La MMU no la troba a la RAM i avisa el sistema operatiu. És una **fallada de pàgina** (*page fault*).
3. El sistema operatiu **busca la pàgina** a l'espai d'intercanvi.

4. **La porta a un marc lliure de la RAM** i actualitza la taula de pàgines. Si no hi ha cap marc lliure, abans n'envia una altra al disc.
5. El procés **continua on era**, com si no hagués passat res.

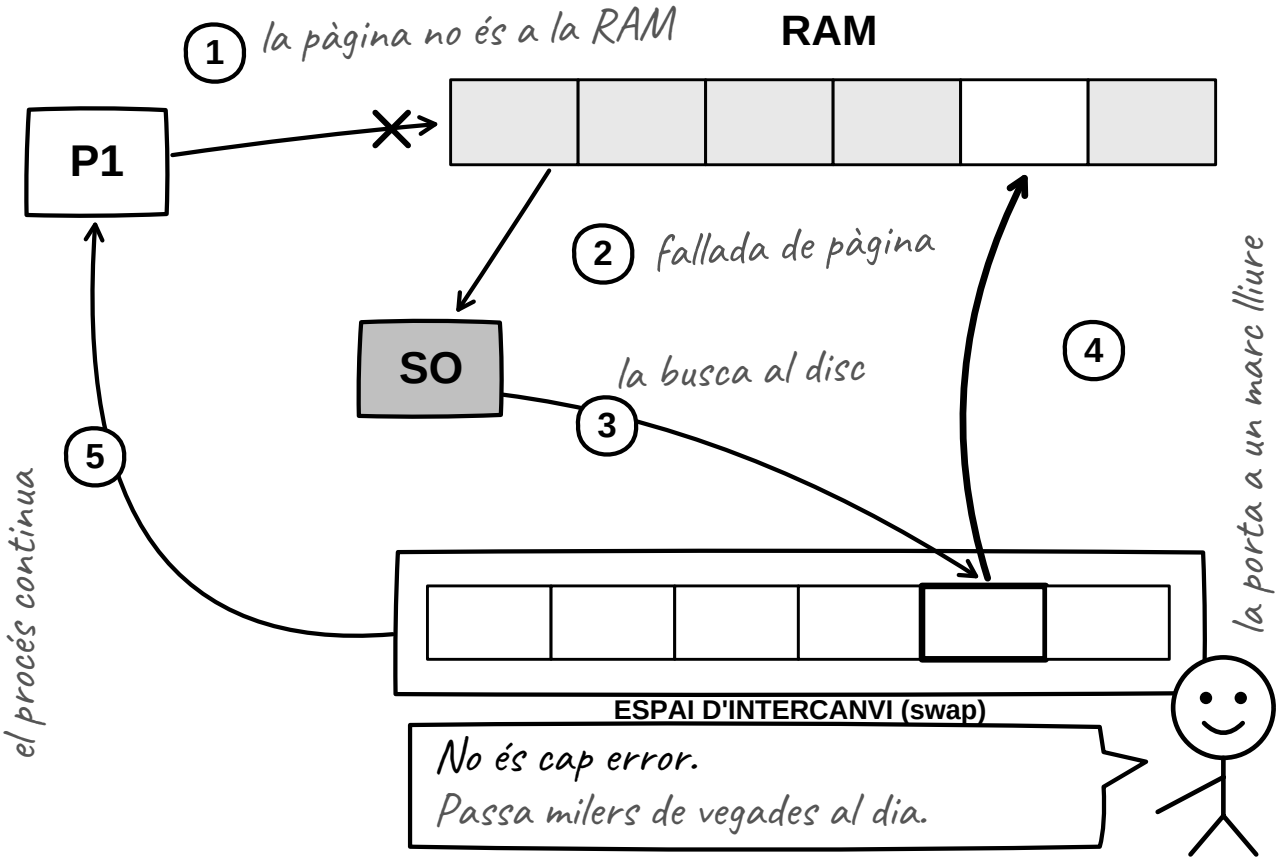

FIG 5.6 Una fallada de pàgina: la pàgina que falta ve del disc a un marc lliure.

Una fallada de pàgina **no és un error**. Passa milers de vegades cada dia i el programa no se n'assabenta. Mentre el disc porta la pàgina, el procés està **bloquejat**, com vas veure al tema 4.

Per què l'ordinador s'arrossega quan falta RAM. El disc és moltíssim més lent que la RAM: un disc dur, unes cent mil vegades, i fins i tot un SSD, unes mil. Si falta molta RAM, hi ha fallades de pàgina contínuament, i el sistema passa més temps movent pàgines que treballant. Això es diu **hiperpaginació** (*thrashing*). La solució de debò no és un disc més ràpid: és més RAM.

Practica 5.8

a) Ordena els passos d'una fallada de pàgina, de l'1 al 5.

	Pas	Ordre
a	El sistema operatiu busca la pàgina a l'espai d'intercanvi	
b	El procés continua on era	
c	El procés fa servir una adreça d'una pàgina que no és a la RAM	
d	El sistema operatiu porta la pàgina a un marc lliure i actualitza la taula	
e	La MMU no la troba a la RAM i avisa el sistema operatiu	

b) Mentre el disc porta la pàgina, en quin estat està el procés?

c) Un company té 8 GiB de RAM i l'ordinador li va molt lent quan obre molts programes. Diu que comprarà un disc més gran. Què li recomanes i per què?

7. La memòria del teu ordinador

7.1. A Linux, des del terminal

L'ordre **free -h** ensenya la RAM i l'espai d'intercanvi, amb unitats fàcils de llegir:

```
               total   used   free   shared  buff/cache  available
Mem:           15Gi    5,2Gi  3,1Gi  410Mi   7,2Gi       9,8Gi
Swap:          2,0Gi   0B     2,0Gi
```

- La fila **Mem** és la RAM i la fila **Swap**, l'espai d'intercanvi.
- **available** és la RAM que encara poden fer servir els programes. És la columna que importa.
- **buff/cache** és RAM que el sistema aprofita mentre ningú no la necessita, i que torna quan un programa la demana.

7.2. A Windows

A l'**Administrador de tasques** (Ctrl+Shift+Esc), la pestanya **Rendiment** i l'apartat **Memòria** et diuen quanta RAM hi ha, quanta es fa servir i quanta en queda disponible.

Practica 5.9 — A l'ordinador

a) Executa **free -h**. Quanta RAM total té l'ordinador i quanta hi ha disponible?

b) Quant espai d'intercanvi té, i quant en fa servir?

c) Obre molts programes i pestanyes i torna a executar **free -h**. Què ha canviat?

El que has de saber sí o sí

- **Com més ràpida és una memòria, més cara és i menys n'hi ha**: memòria cau, RAM, disc.
- Els programes fan servir **adreces virtuals**. La **MMU** les tradueix a **físiques**.
- $\mathbf{d_F = d_B + d_V}$. Si l'adreça surt de l'espai del procés, la MMU no la deixa passar.
- **Particions estàtiques**: trossos fixos, **fragmentació interna**.
- **Particions dinàmiques**: trossos a mida, **fragmentació externa**. Es resol **compactant**.
- **Paginació**: pàgines del procés en marcs de la RAM, **no cal que vagin seguides**. Sense fragmentació externa.
- **Pàgines necessàries** = mida del procés ÷ mida de pàgina, **arrodonint cap amunt**.
- **Traduir**: quocient = pàgina, residu = desplaçament, **marc × mida + desplaçament**.
- **Memòria virtual** = RAM + **espai d'intercanvi** al disc.
- Una **fallada de pàgina** no és un error: el sistema porta la pàgina del disc a la RAM.
- Si falta molta RAM, **hiperpaginació**: l'ordinador s'arrossega. La solució és més RAM.

Paraules noves

Paraula	Què és	Les meves notes
Memòria cau *(cache)*	Memòria molt ràpida i petita dins el processador.	
Adreça virtual	La que fa servir el programa, comptant des del seu inici.	
Adreça base	On comença el procés a la RAM.	
Adreça física	On és de debò a la RAM.	
MMU	Circuit que tradueix adreces virtuals a físiques.	
Fragmentació interna	Memòria perduda dins la partició d'un procés.	
Fragmentació externa	Memòria lliure repartida en forats massa petits.	
Compactació	Ajuntar els forats movent els processos.	
Pàgina *(page)*	Tros de mida fixa d'un procés.	
Marc *(frame)*	Tros de RAM de la mida d'una pàgina.	
Taula de pàgines	Diu a quin marc és cada pàgina.	
Espai d'intercanvi *(swap)*	Part del disc que fa de RAM.	
Memòria virtual	RAM i espai d'intercanvi junts.	
Fallada de pàgina *(page fault)*	Quan una pàgina que es necessita no és a la RAM.	

Repàs del tema

A. Vertader o fals

Frase	V o F	Per què
La memòria cau té més capacitat que la RAM		
L'adreça física és la base més l'adreça virtual		
Les particions estàtiques provoquen fragmentació externa		
Amb paginació, les pàgines d'un procés han d'anar seguides a la RAM		
Una fallada de pàgina vol dir que el programa s'ha penjat		
L'espai d'intercanvi és al disc		

B. Preguntes curtes

1. Quina diferència hi ha entre fragmentació interna i externa?

2. Per a què serveix la MMU?

3. Per què amb paginació no hi ha fragmentació externa?

4. Per què un ordinador amb poca RAM va lent quan obres molts programes?

C. Problemes

Problema 1. Hi ha **512 KiB** per als processos, en particions estàtiques de **64 KiB**. Quantes particions hi ha? Arriben processos de **50**, **64**, **20** i **100 KiB**. Quins hi caben, quanta memòria es perd amb cada un i quanta en total?

Problema 2. Pàgines de **8 KiB**. Un procés ocupa **30 KiB**. Quantes pàgines necessita i quanta memòria perd? La seva taula de pàgines és: pàgina 0 → marc 3, 1 → 7, 2 → 1, 3 → 4. Tradueix les adreces virtuals **20 KiB** i **26 KiB**.

Problema 3. Completa: un procés amb base **3500** demana l'adreça virtual **700**; un altre amb base **5200** acaba a l'adreça física **6000**. Quina és la física del primer i la virtual del segon?

Problema 4. Explica pas a pas què passa quan un procés necessita una pàgina que és a l'espai d'intercanvi, i en quin estat queda el procés mentrestant.

Les meves notes