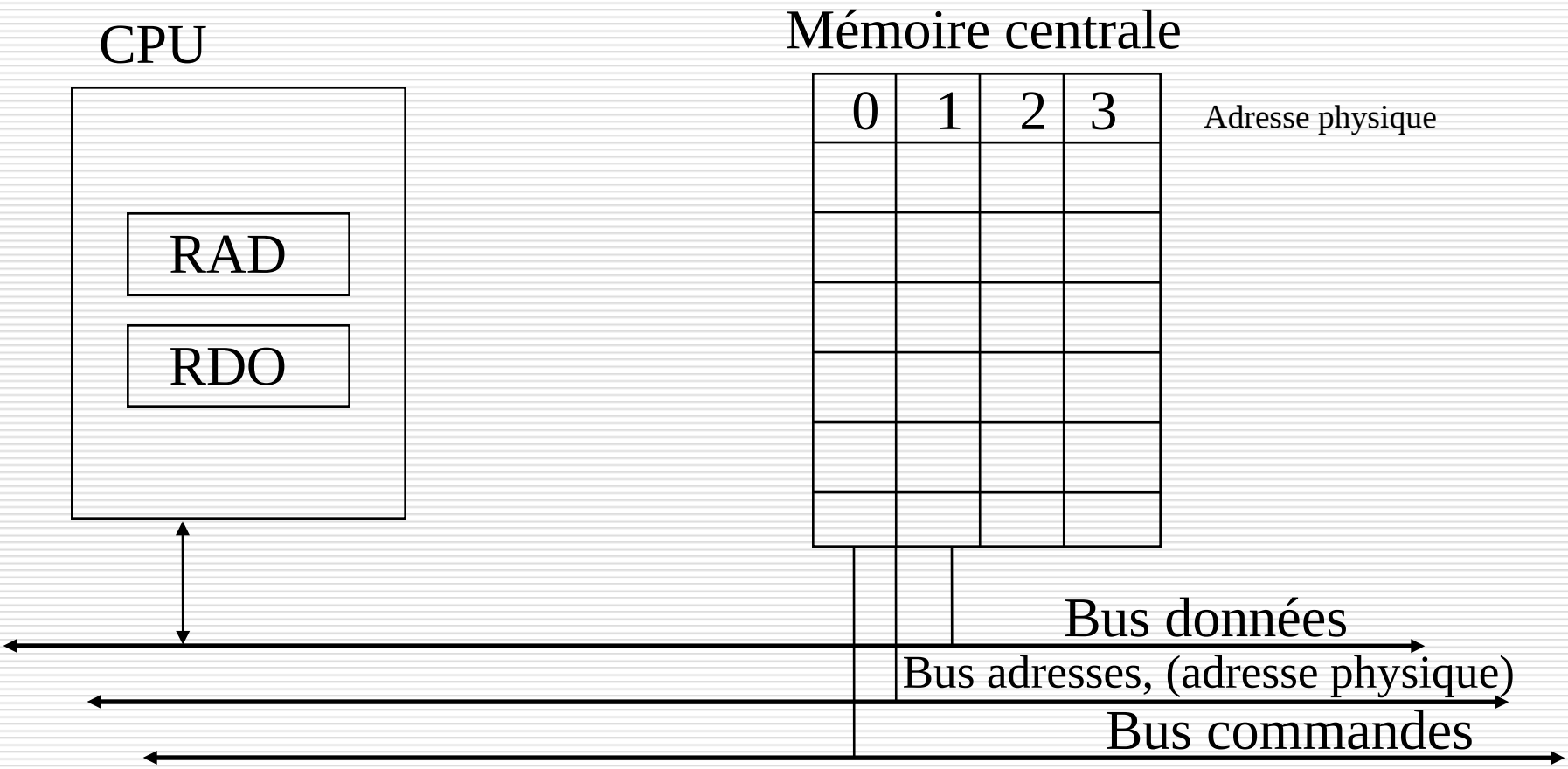
Gestion de la mémoire centrale

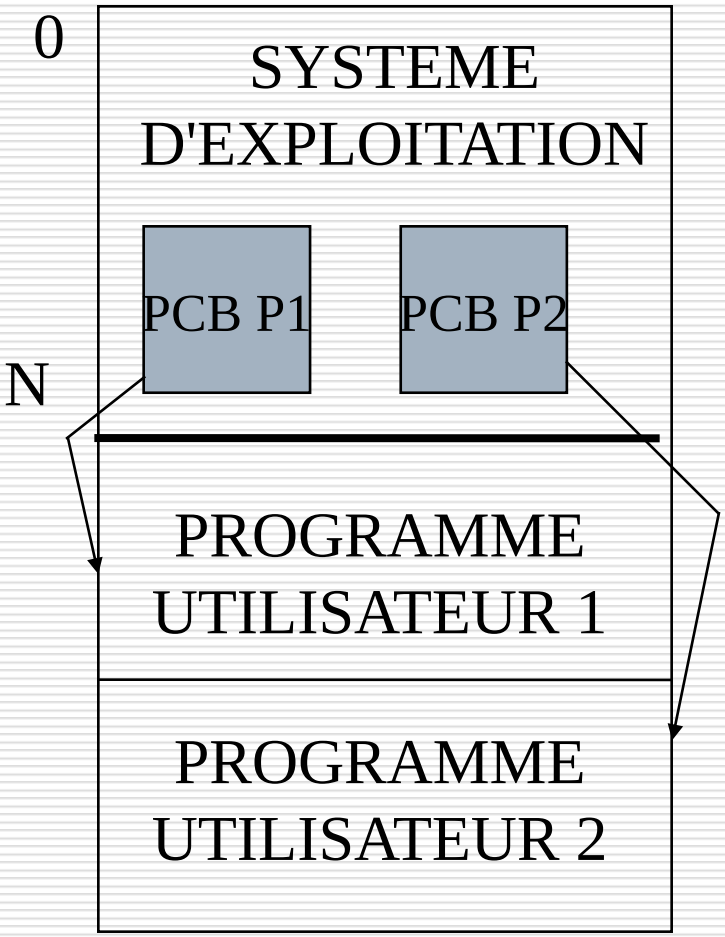
- Allocation de la mémoire physique : la pagination
- Gestion de la mémoire virtuelle
- Un exemple : linux

Rappels : Fonction de mémorisation

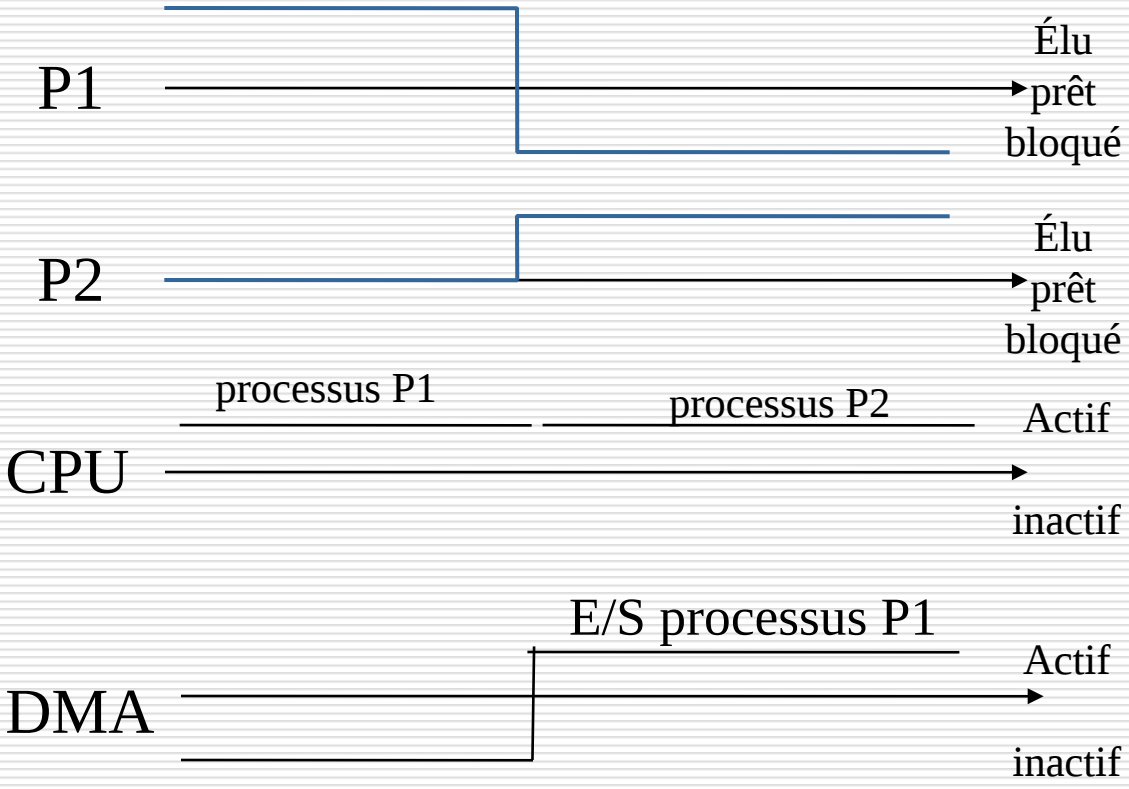


MC :Ensemble linéaire de mots d'adresses physiques contigües

Multiprogrammation : cas idéal



P1 : 100 ms calcul / 100 ms E/S
P2 : 100 ms calcul / 100 ms E/S
0 % d'inactivité du processeur



Multiprogrammation et gestion mémoire

le cnam

Trois problèmes à résoudre vis-à-vis de la mémoire :

- Définir un espace d'adressage indépendant pour chaque processus
- Protéger les espaces d'adressages des processus entre eux
- Allouer de la mémoire physique à chaque espace d'adressage

L'espace d'adressage d'un processus correspond à l'ensemble des adresses auxquelles il peut accéder au cours de son exécution

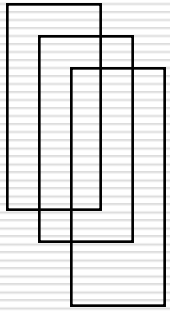
Allocation de la mémoire physique : la pagination

Multiprogrammation et gestion mémoire

programme

ensemble insécable de mots contigus

ensemble sécable de mots contigus



espace d'adressage linéaire

→ Allocation en partitions de tailles variables

espace d'adressage paginé

espace d'adressage segmenté

→ Allocation en partitions de tailles fixes

Allocation en partitions de tailles variables

- La mémoire est découpée en partitions de tailles variables, c'est-à-dire adaptables à la taille des programmes.

- La mémoire est formée d'un ensemble de zones libres et de zones occupées (partitions allouées). Une zone libre est caractérisée par son adresse d'implantation (adr) et sa taille (taille).

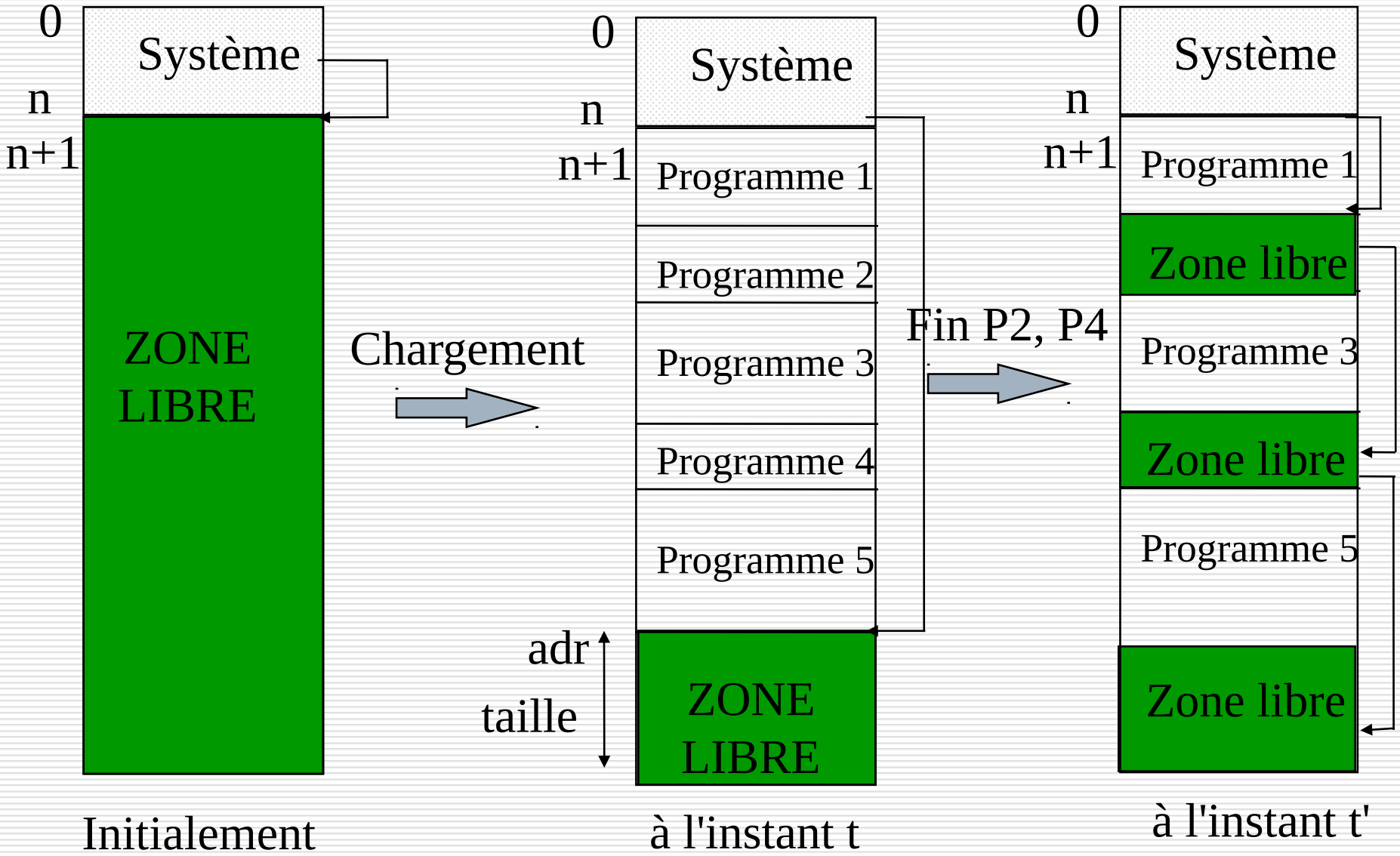
- Allouer un programme P de taille Taille(P) :
  - Trouver une zone libre telle que :

    Taille (Zone Libre) >= Taille (P)

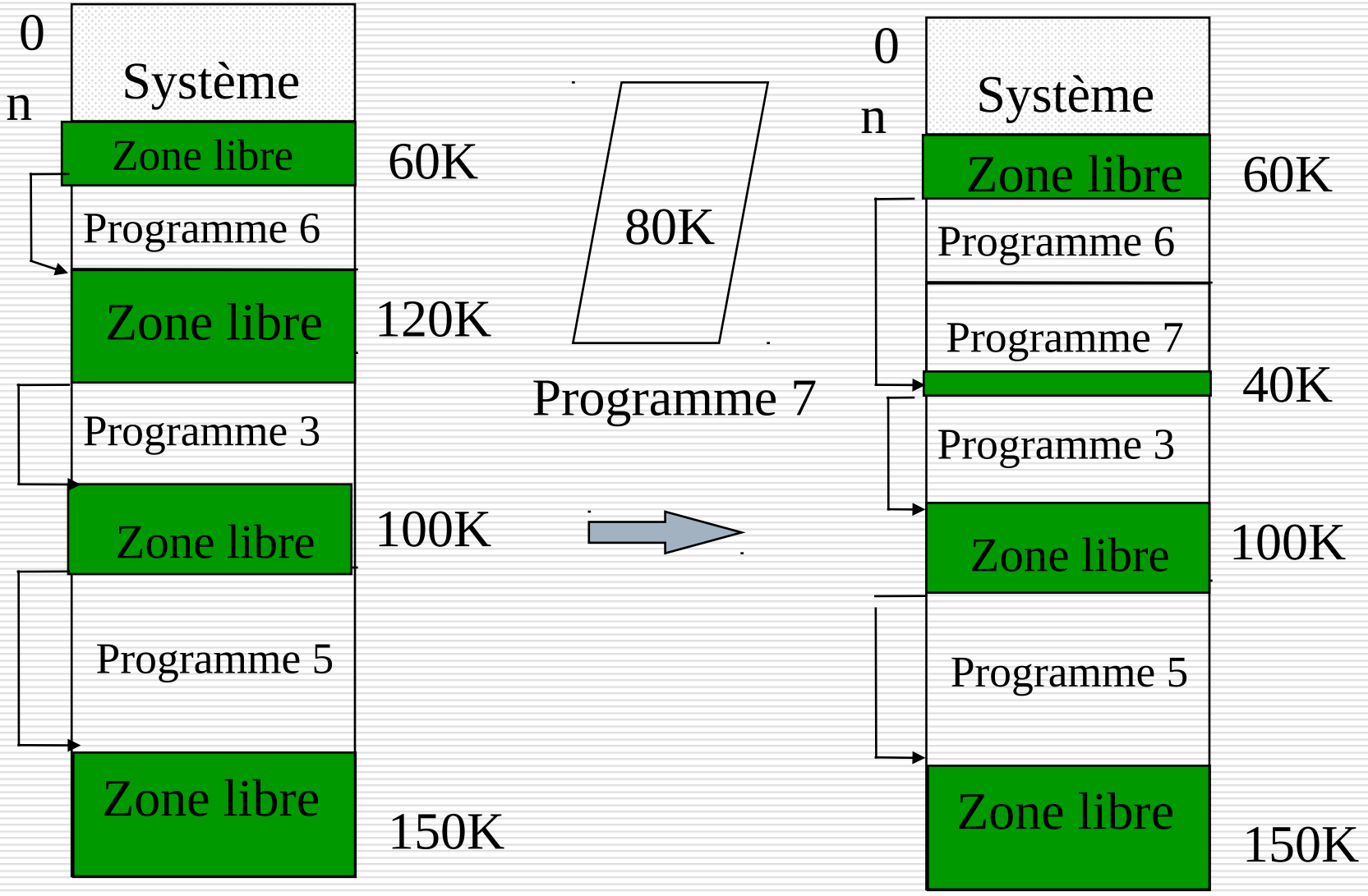
Le système maintient une liste des zones libres en MC

- **First Fit : la première zone qui convient**
- **Best Fit : celle générant le moins de perte**

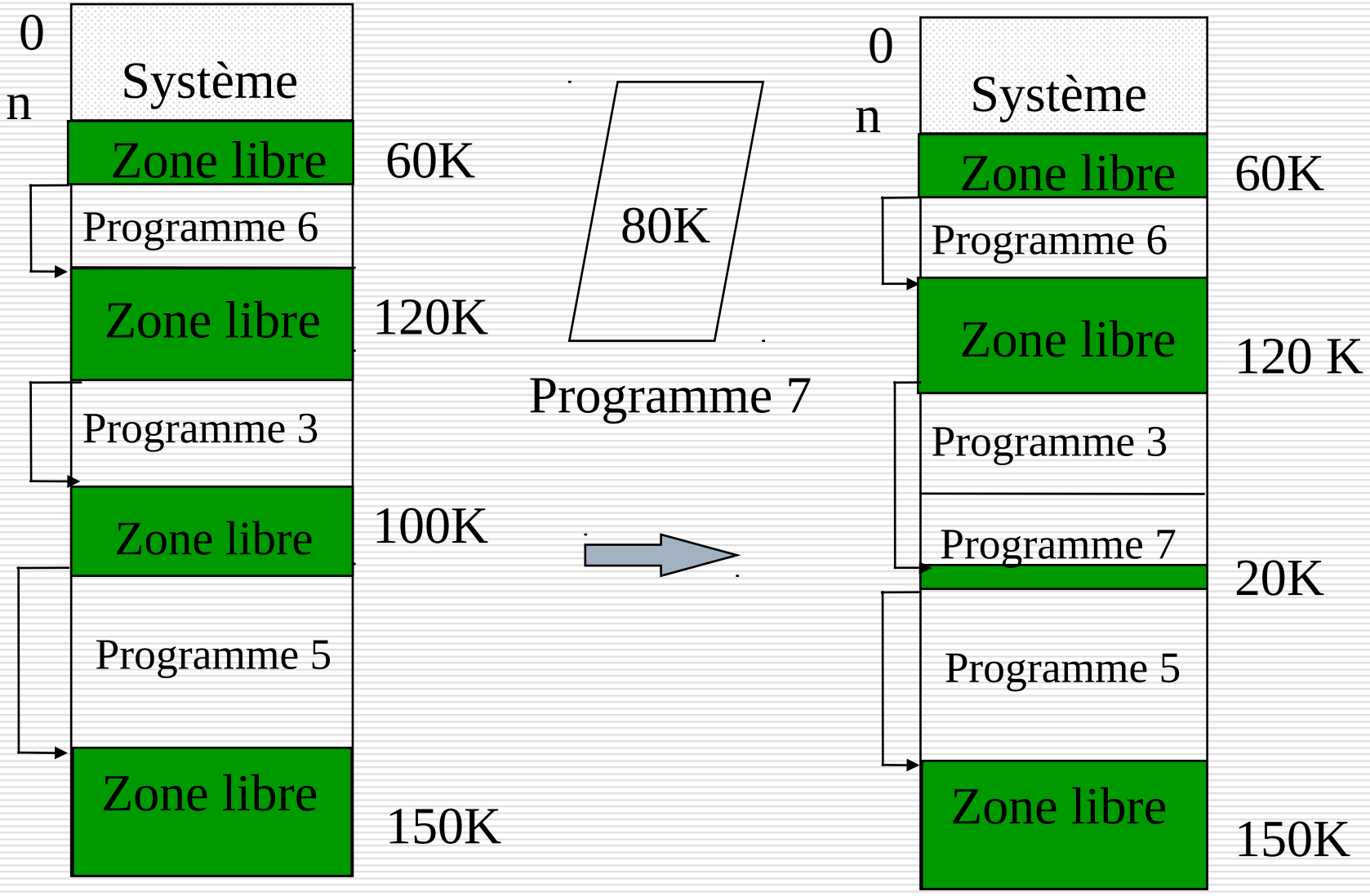
Allocation en partitions de tailles variables



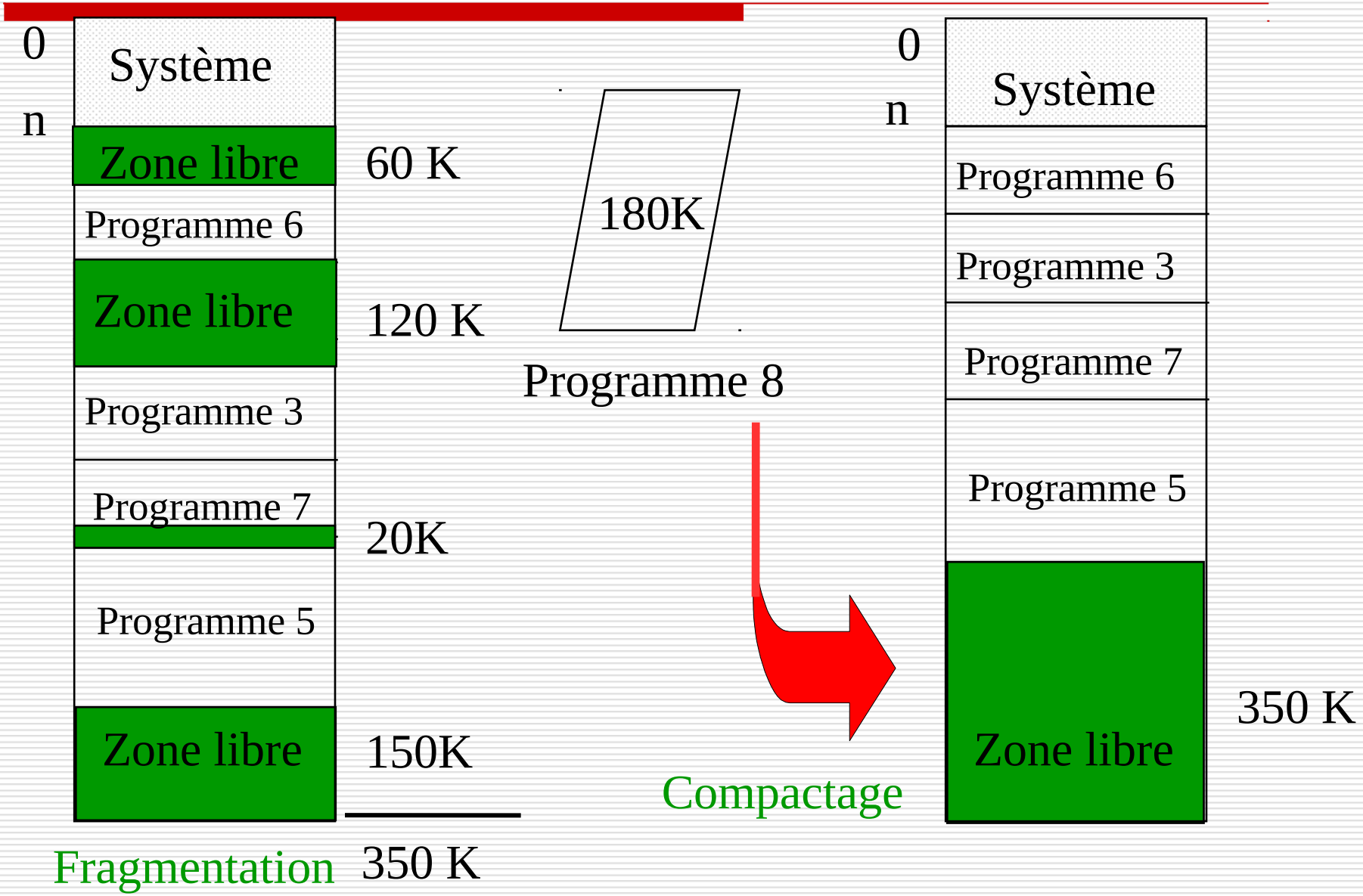
Allocation First Fit



Allocation Best Fit



Compactage de la mémoire



Compactage de la mémoire

le cnam

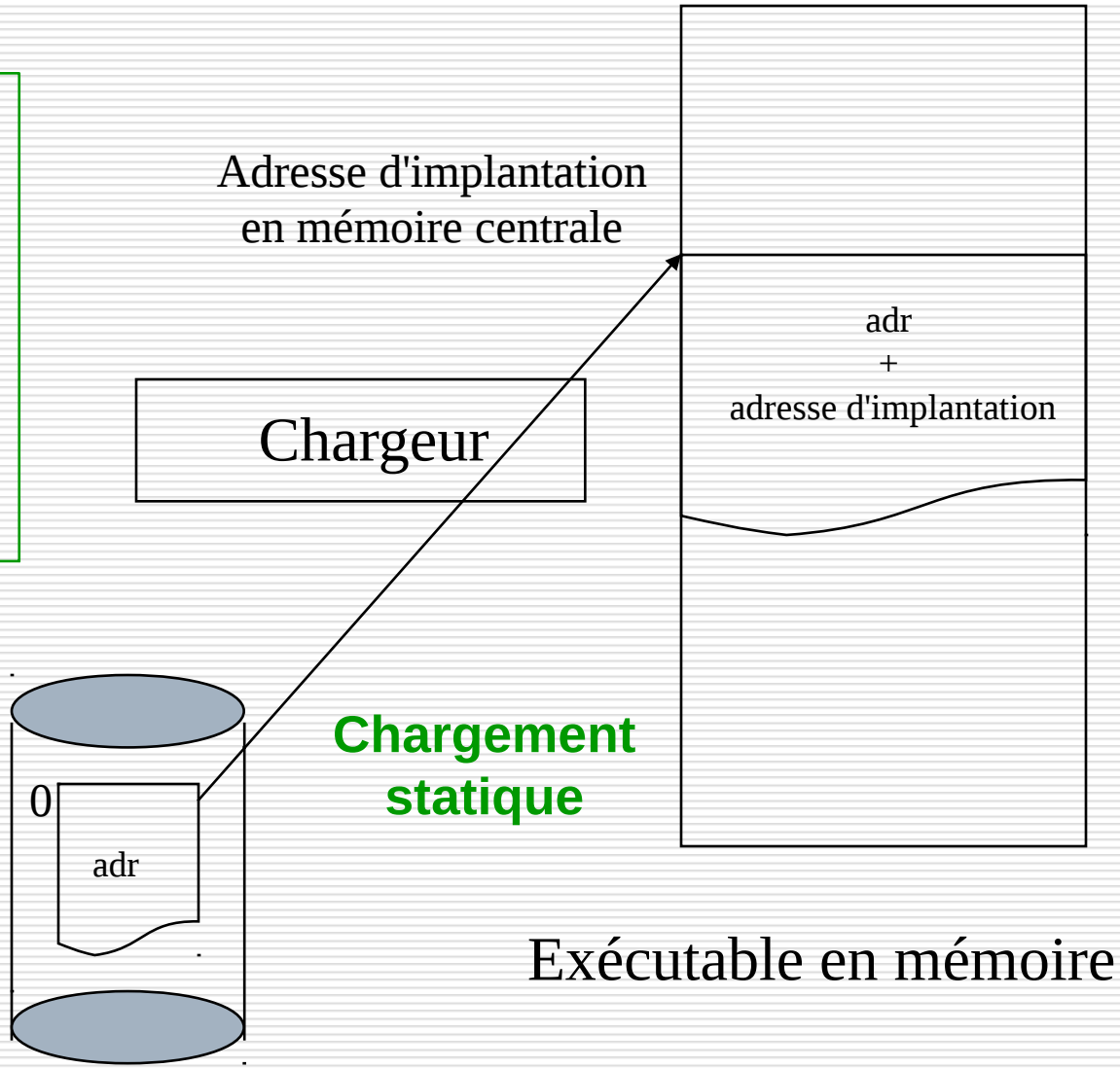
- Allocations et désallocations successives des programmes créent des trous (zones libres de taille insuffisante) en mémoire centrale : **fragmentation**

- Le **compactage** consiste à déplacer les programmes en mémoire centrale de manière à ne créer qu'une seule et unique zone libre.

- Le compactage est une opération très coûteuse. Elle suppose une **translation des adresses dynamique**.

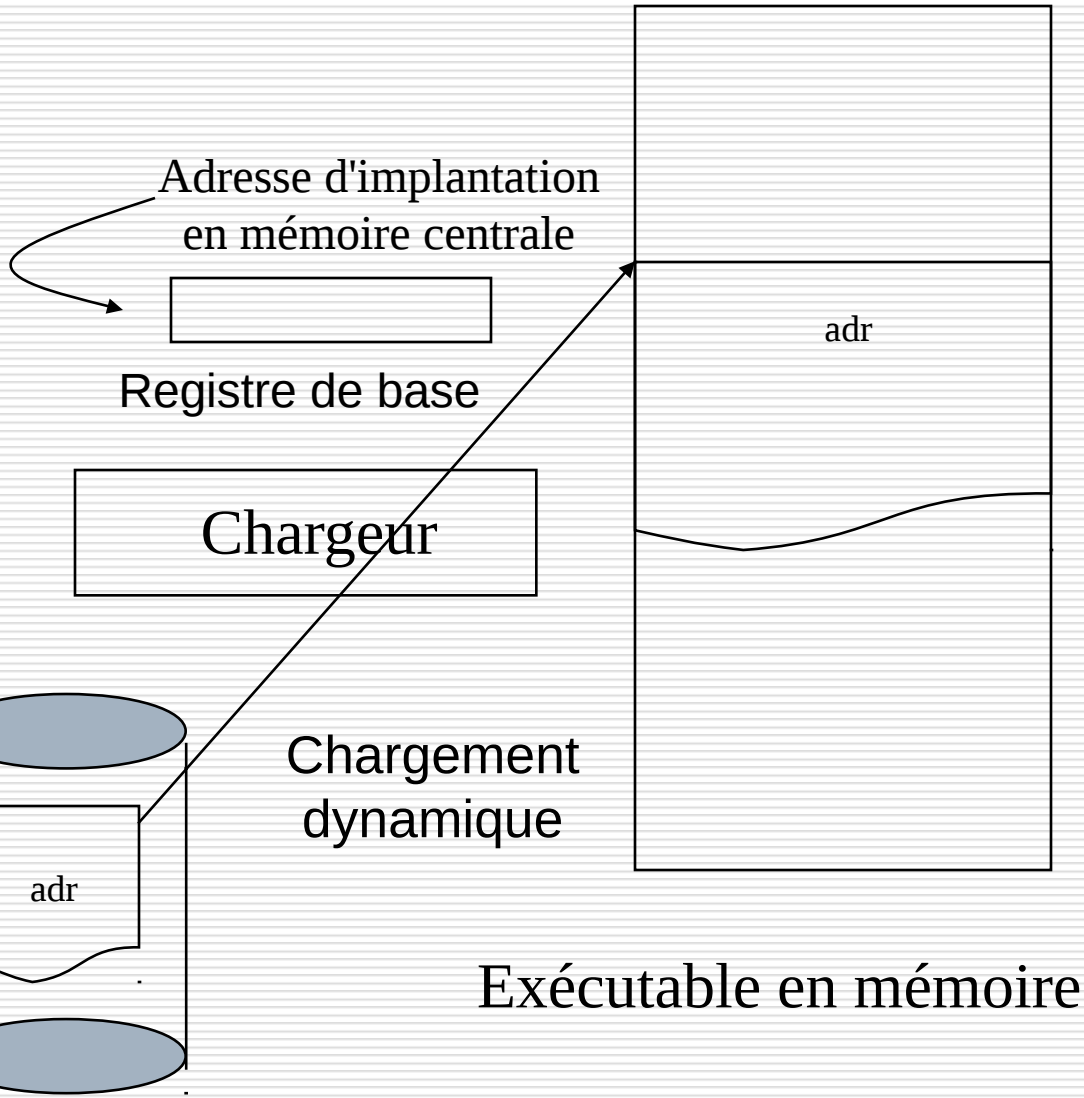
Chargement statique

Les adresses du programme sont translatées de la valeur de l'adresse d'implantation en MC au chargement



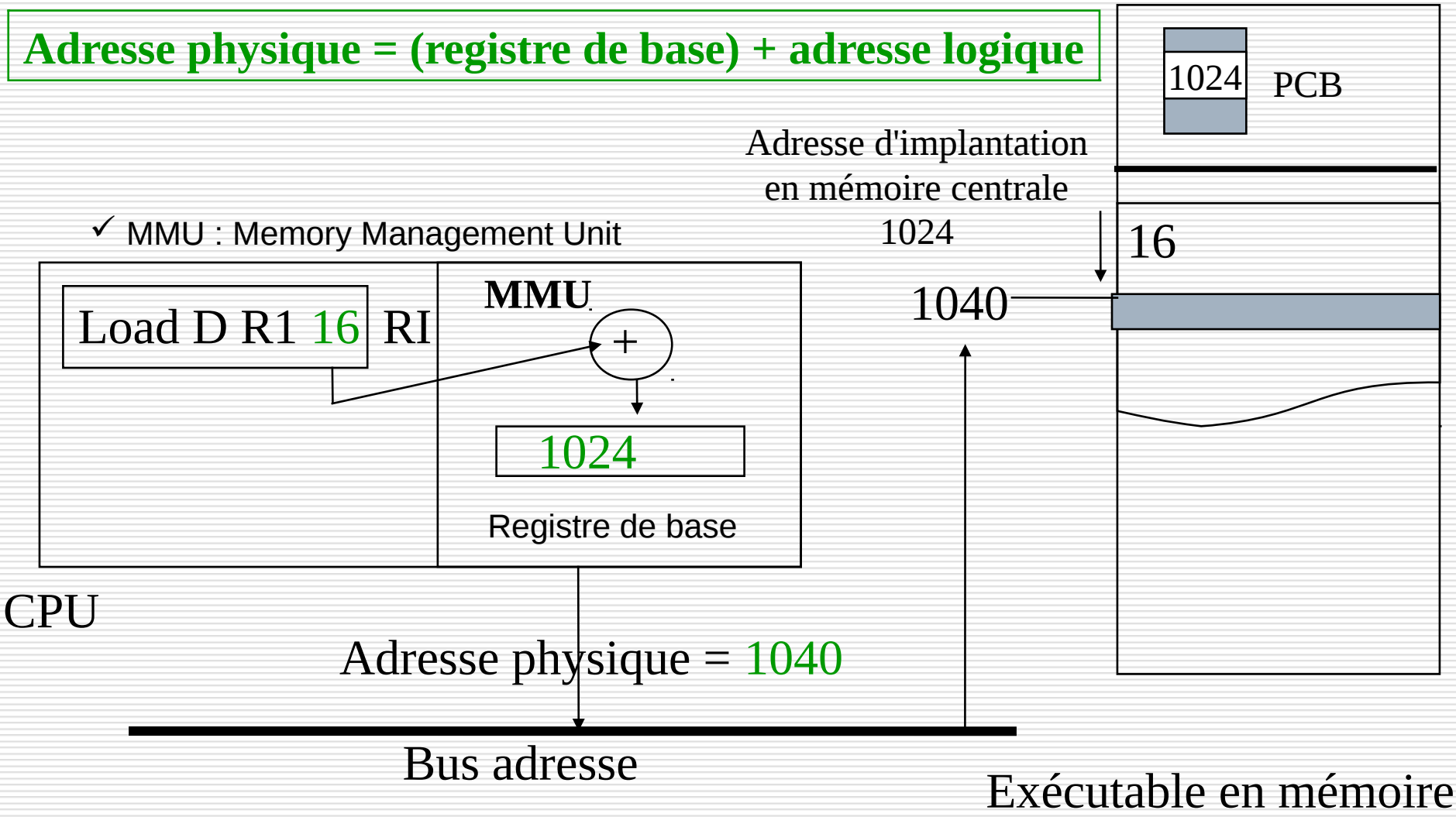
Chargement dynamique

Les adresses du programme sont translatées de la valeur de l'adresse d'implantation en MC à l'exécution,

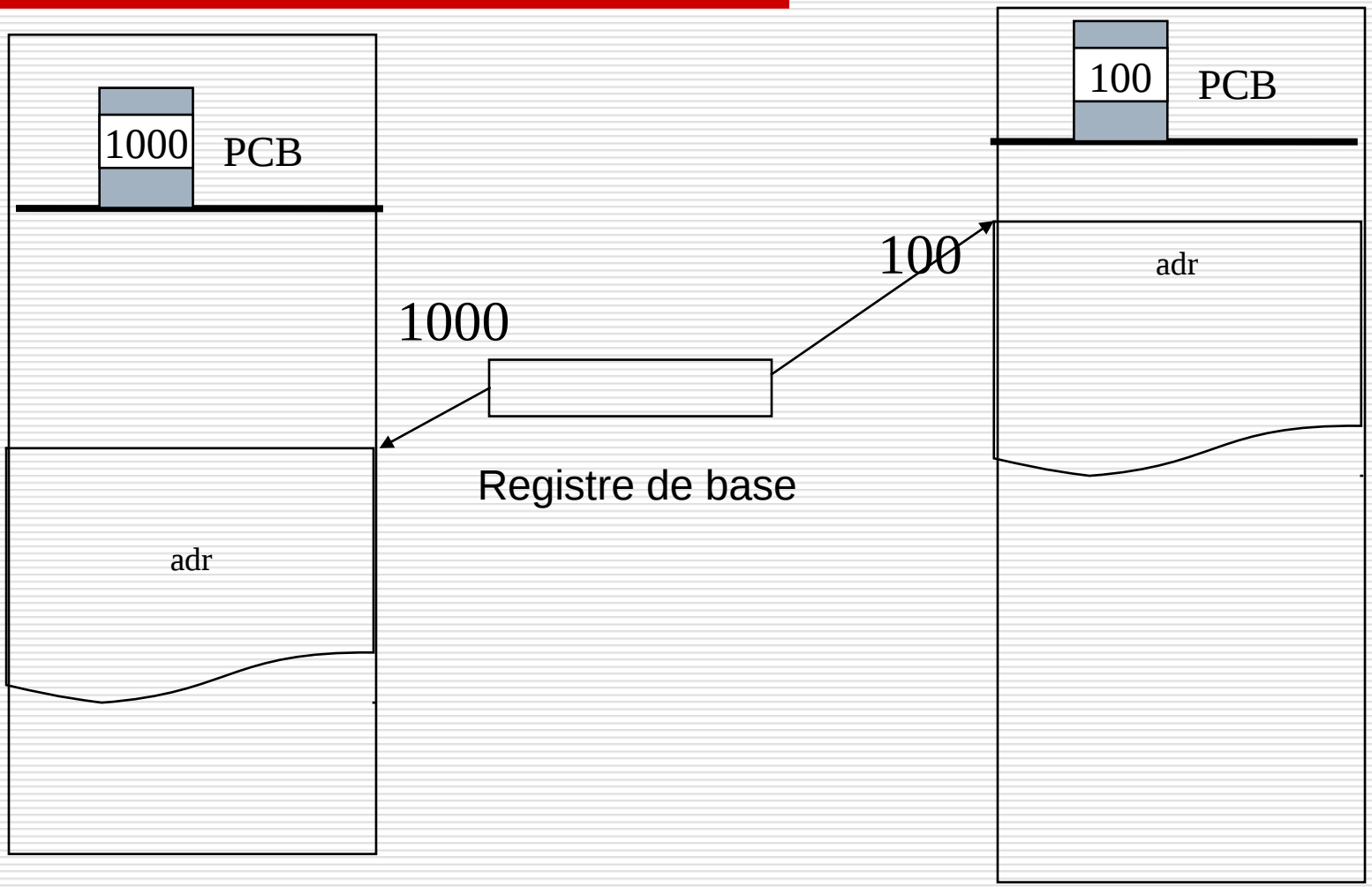


Conversion d'adresse

Adresse physique = (registre de base) + adresse logique



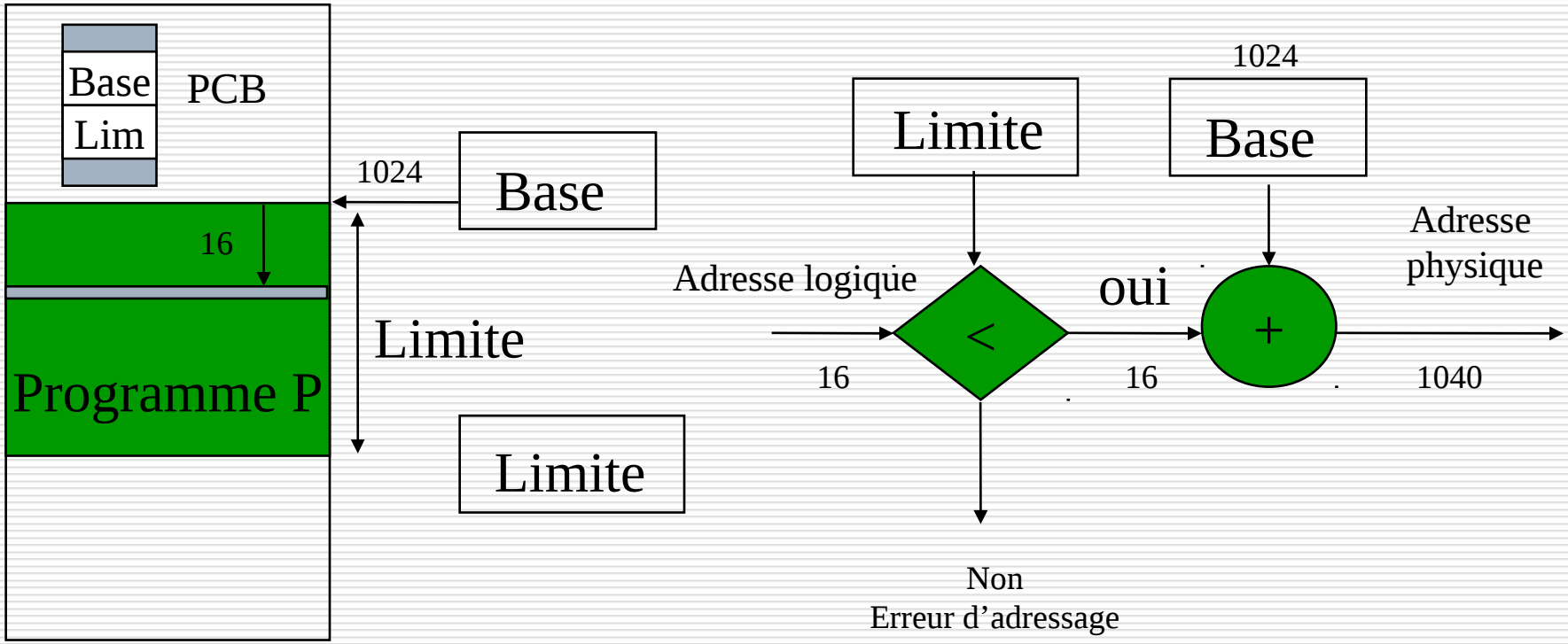
Compactage



Déplacer un programme ↔ modifier la valeur du registre de base

Protection si allocation d'un seul tenant

Toute adresse générée par l'exécution du programme doit appartenir à l'intervalle des adresses qui lui sont allouées



Registres Bornes De la MMU

L'espace d'adresse logiques est différent de l'espace d'adresse physique

Faiblesses de l'allocation d'un seul tenant

le cnam

- Nécessité d'une opération de compactage de la mémoire

- Exigence d'allouer le programme en une zone d'un seul tenant

Diviser le programme en portions de taille fixe et égale à l'unité d'allocation de la mémoire centrale : les **pages**

La pagination

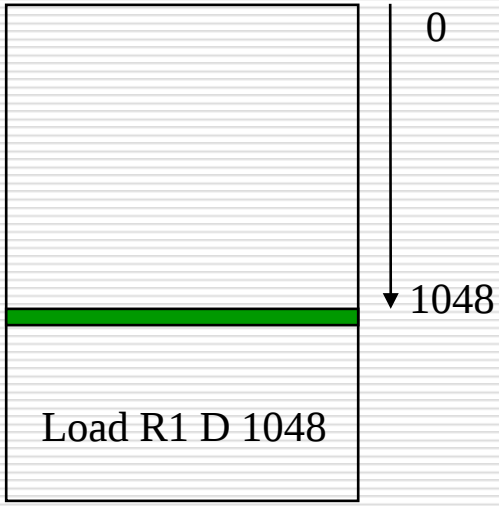
- L‘espace d’adressage linéaire du programme est coupé en portions de taille fixe et égale à l'unité d'allocation de la mémoire centrale : **les pages**.

- Chaque adresse devient une **adresse paginée** formée d’un couple :

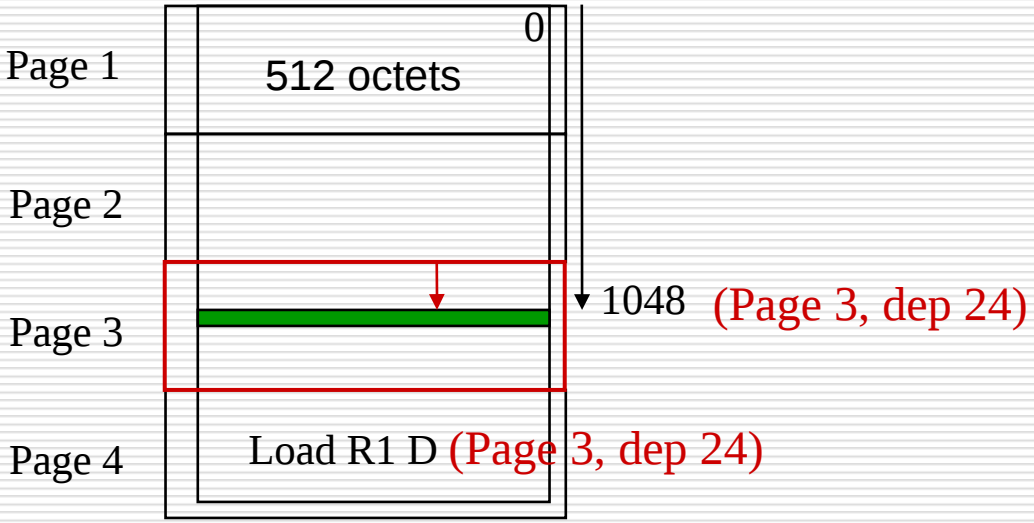
  (numéro de page, déplacement dans la page)

La pagination

L'espace d'adressage linéaire du programme est découpé en **pages**. Chaque adresse devient une **adresse paginée** formée d'un couple (numéro de page, déplacement dans la page)



Espace d'adressage
du programme
LINEAIRE
Adresse linéaire
(déplacement depuis 0)



Espace d'adressage
du programme
PAGINE
Adresse paginée (logique)
(n° de page, déplacement dans la page depuis 0)

La pagination

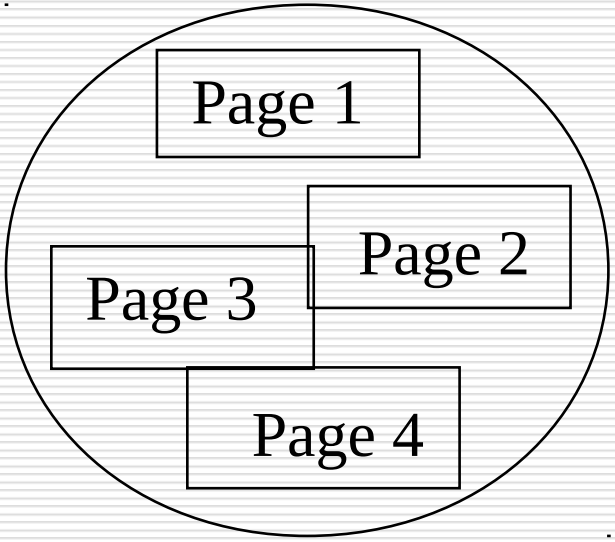
- L'espace d'adressage du programme est découpé en morceaux linéaires de même taille : la **page**.
- L'espace de la mémoire physique est lui-même découpé en morceaux linéaires de même taille : la **case ou cadre de page**
- La **taille d'une case est égale à la taille d'une page**
- Charger un programme en mémoire centrale:

placer les pages dans n'importe quelle case disponible.

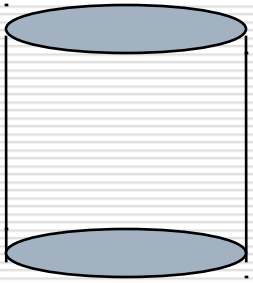
La pagination

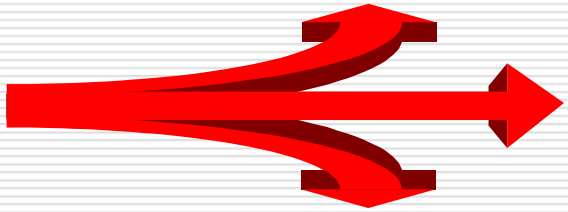
L'espace d'adressage du processus est découpé en **pages**
L'espace de la mémoire physique est découpé en **cases**
La taille d'une case est égale à la taille d'une page

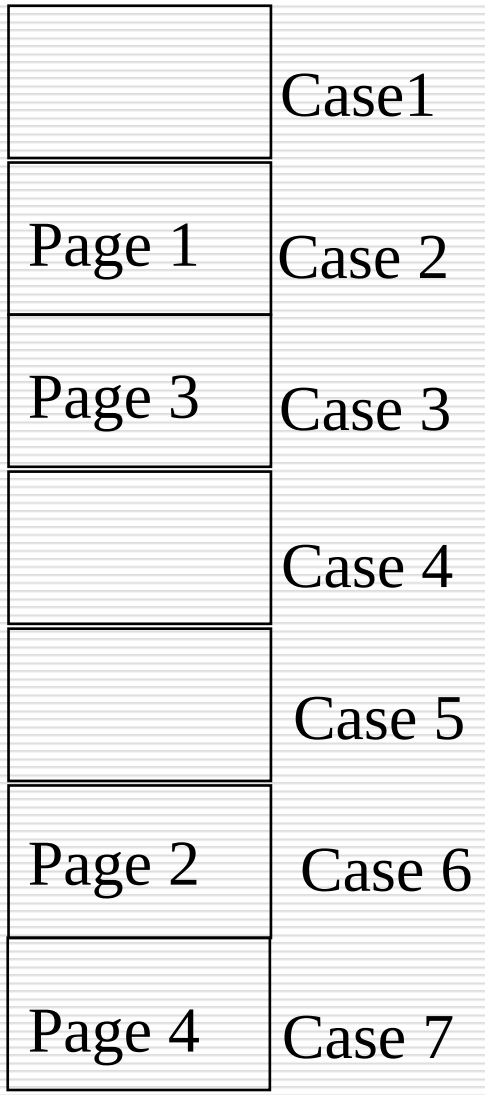
→ **Les pages sont placées dans n'importe quelle case libre de la mémoire centrale**



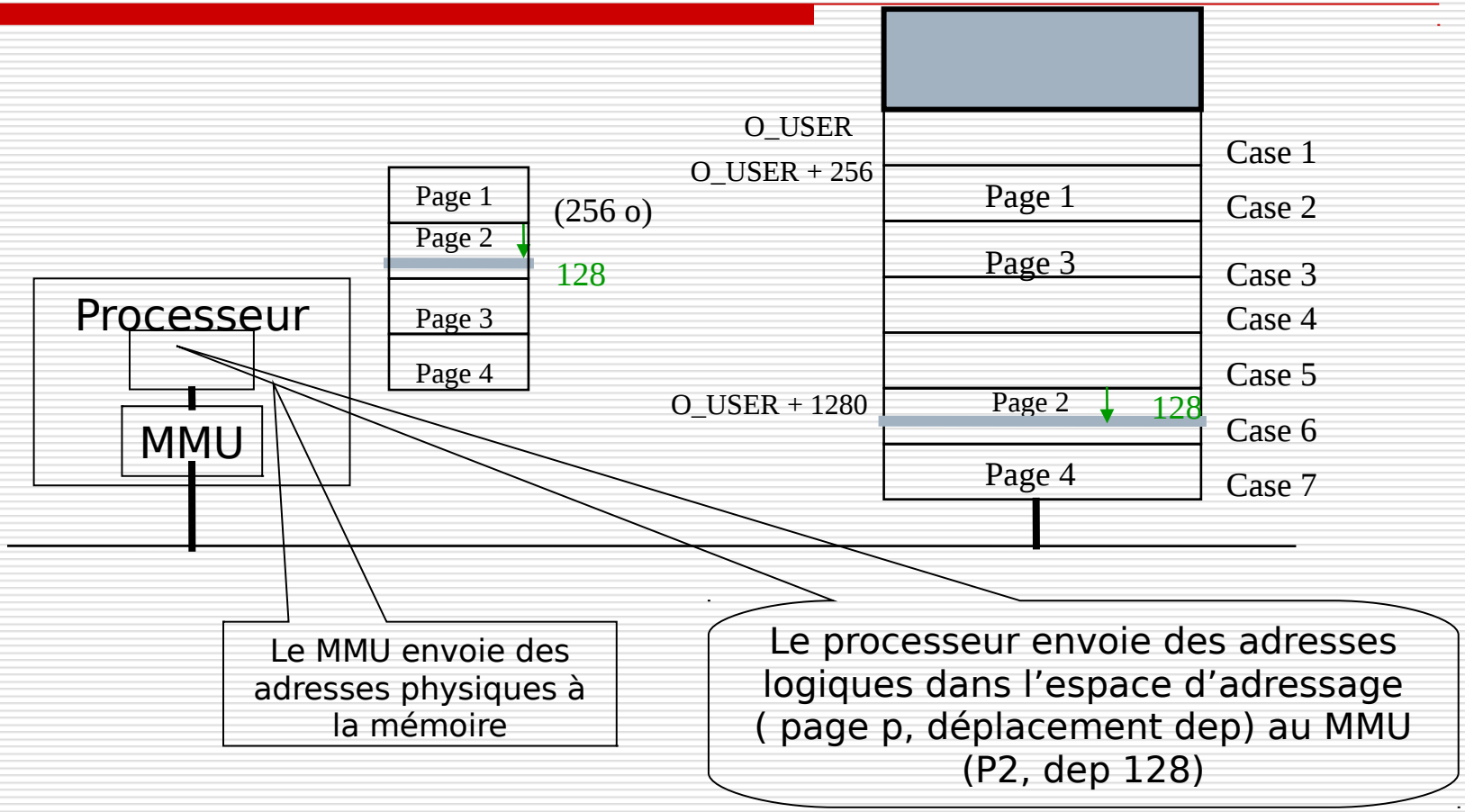
Espace d'adressage du programme







Conversion d'adresse

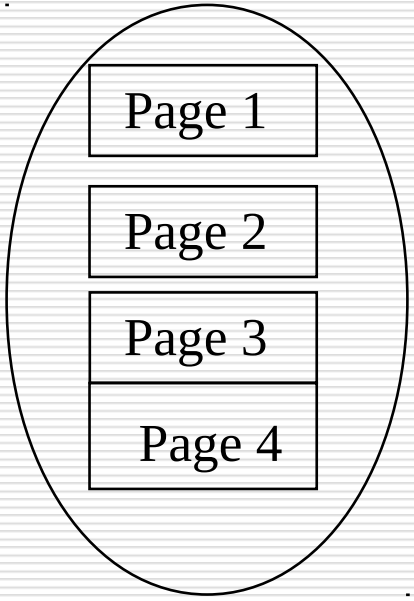


Il faut convertir l'adresse paginée en son équivalent adresse physique

Ad. physique = ad. impl. case contenant la page + deplacement dep

→ Nécessité d'une table des pages

La mémoire paginée : table des pages



Espace d'adressage

Table des pages de l'espace d'adressage

Numéro page	adresse case
1	Adr C2 (O_User + 256)
2	Adr C6 (O_User + 1280)
3	Adr C3 (O_User + 512)
4	Adr C7 (O_User + 1536)

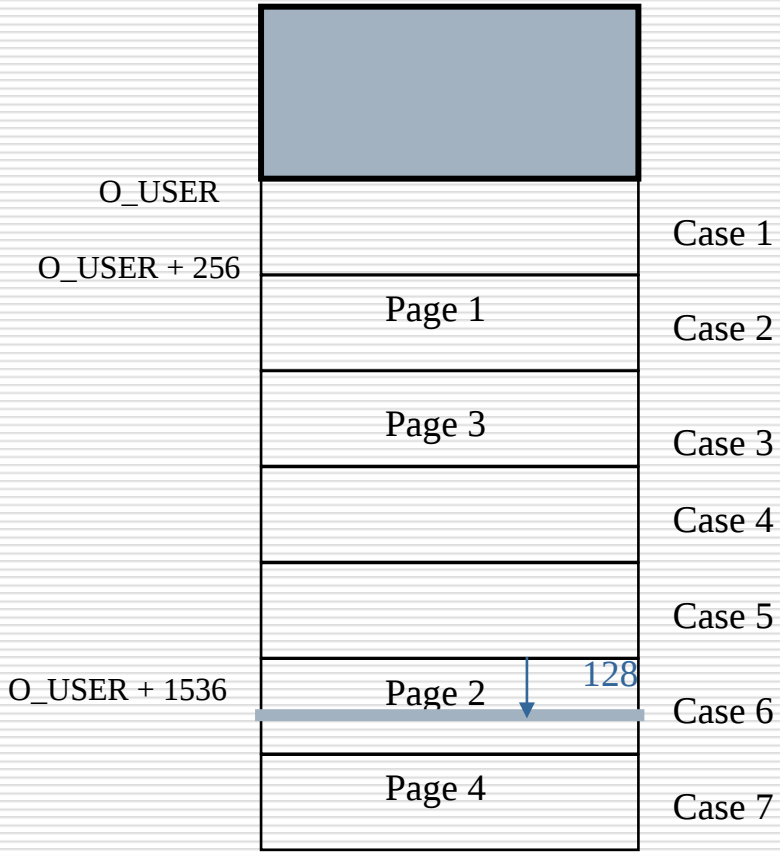
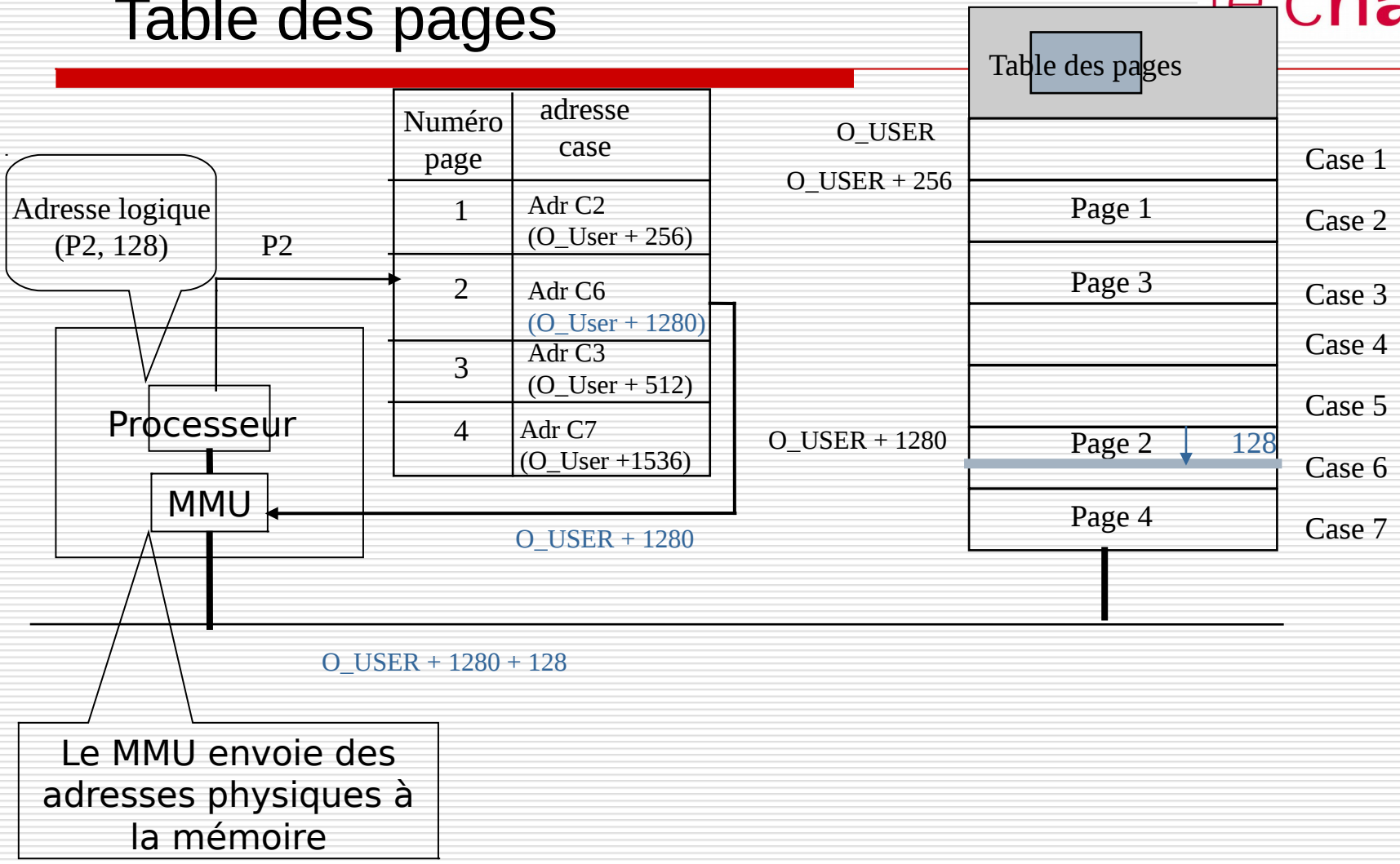


Table des pages



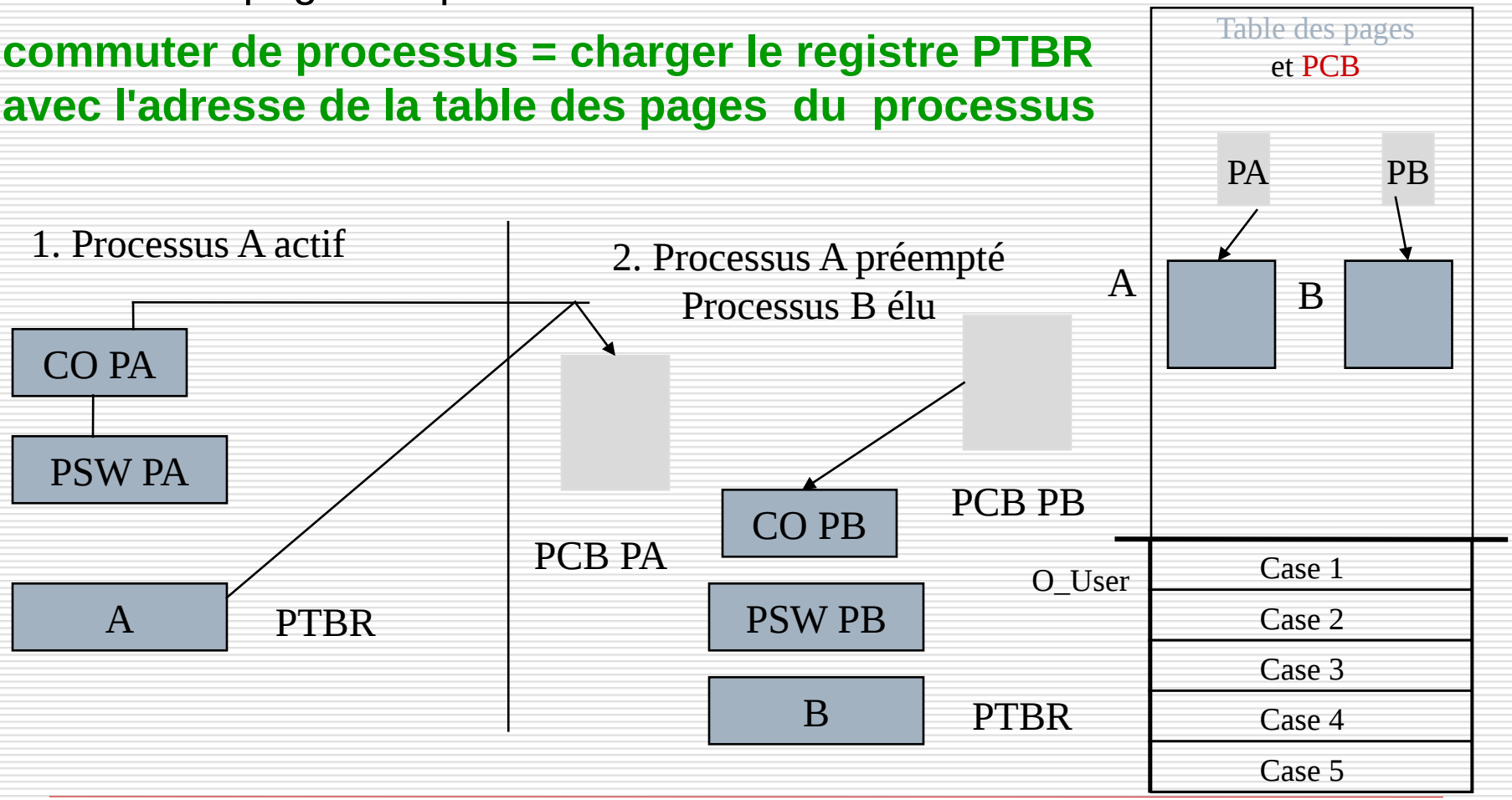
Ad. physique = ad. Impl. case contenant la page (O_User + 1280) + dépl. (128)

Table des pages : structure de données

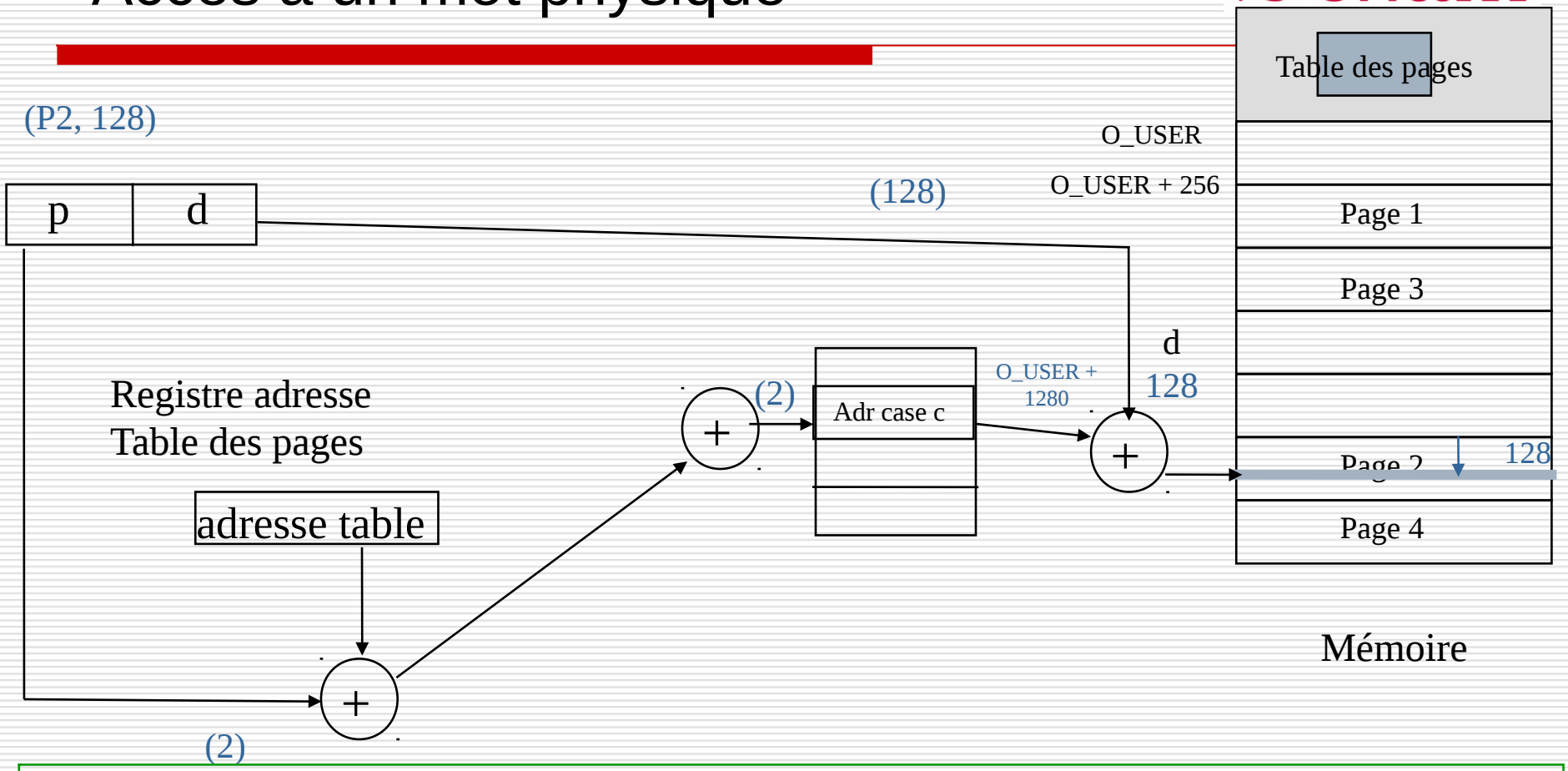
Tables pointées depuis le PCB de chaque processus.

Un registre repère à tout moment l'adresse en MC de la table des pages du processus actif :

commuter de processus = charger le registre PTBR avec l'adresse de la table des pages du processus



Accès à un mot physique

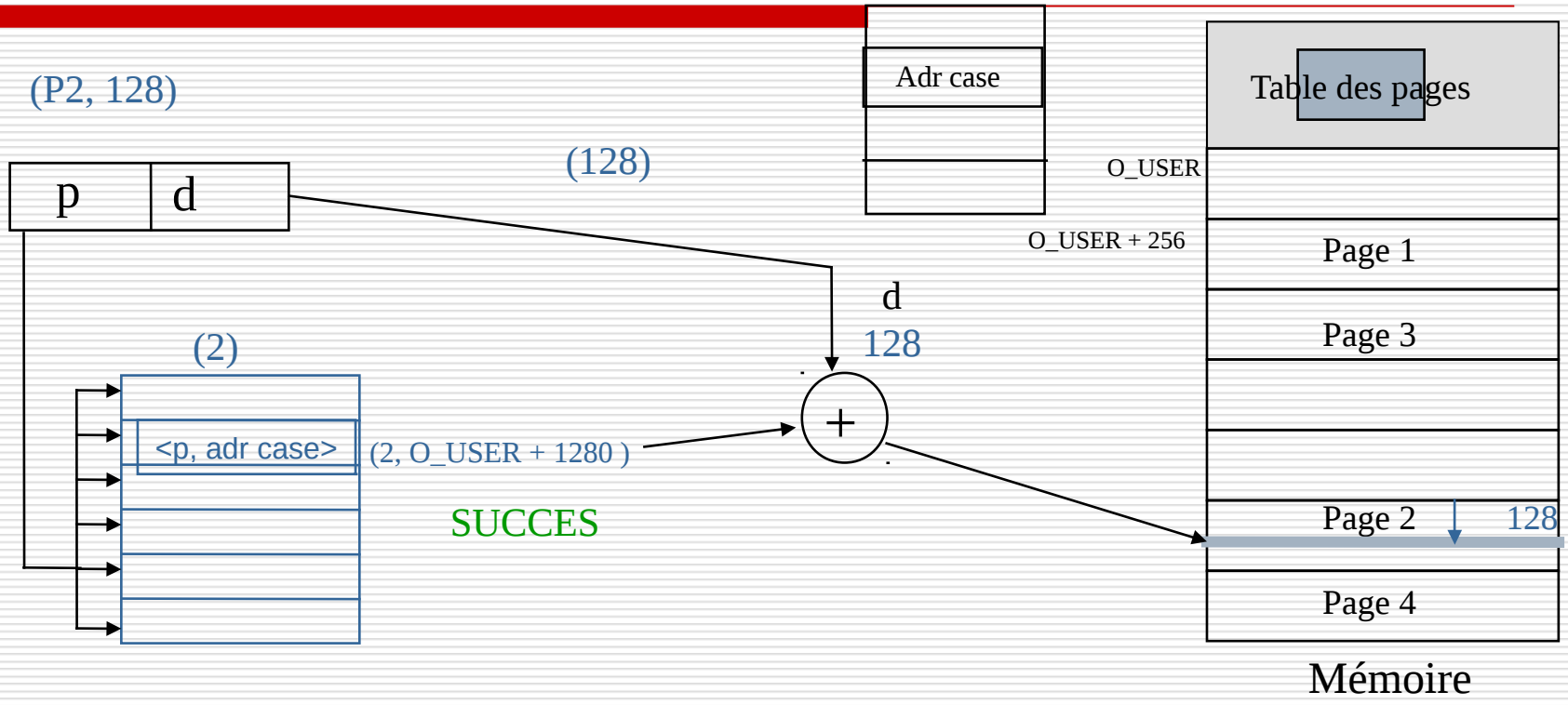


Conversion adresse paginée en adresse physique, deux accès en lecture à la MC :

- 1 accès pour lire l'entrée de la table des pages
- 1 accès pour lire le mot cherché

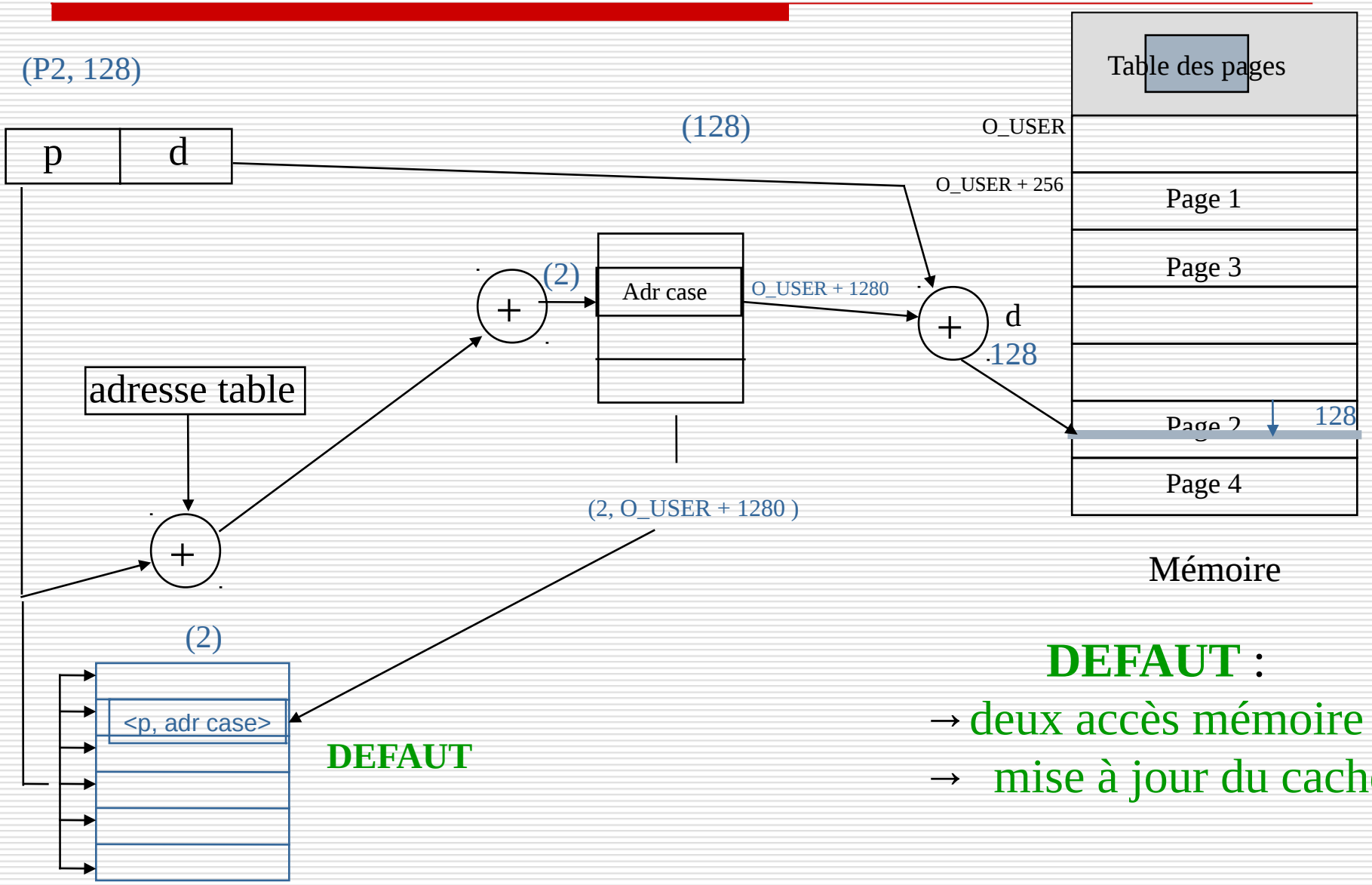
On utilise un **cache associatif** qui mémorise les couples les plus récents (p, c)

Accès à un mot physique



SUCCES : pas d'accès à la table des pages
→ **1 accès mémoire**

Accès à un mot physique



DEFAUT :

→ deux accès mémoire

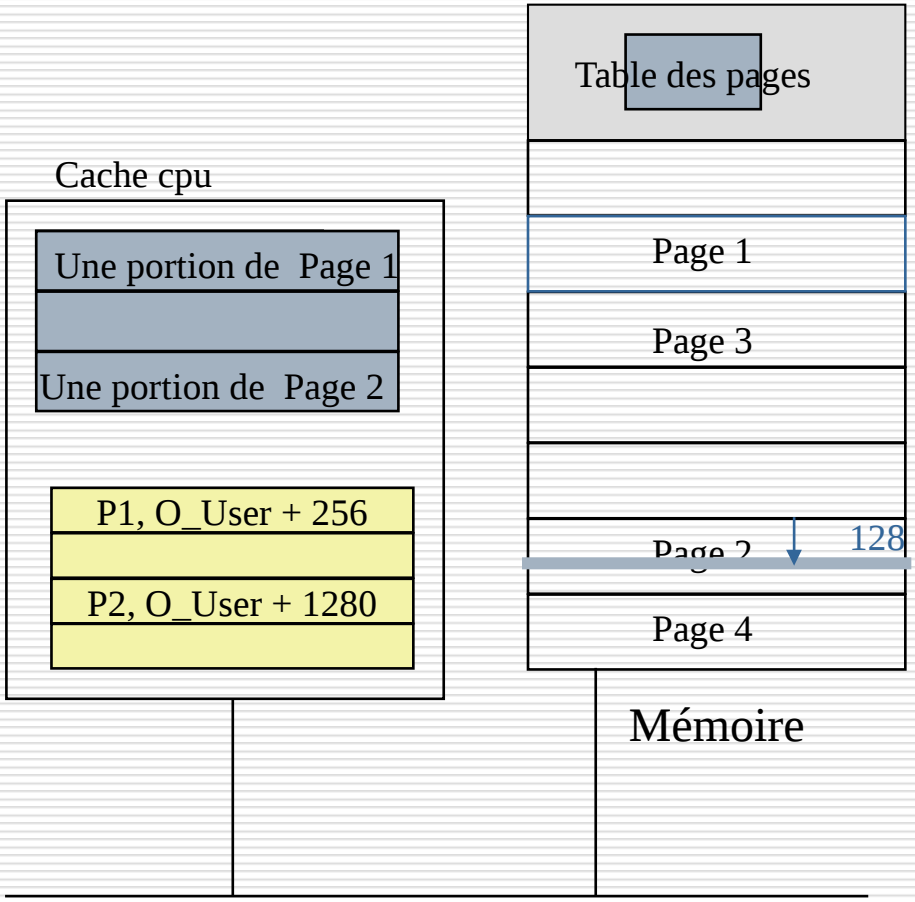
→ mise à jour du cache

Les caches du processeur et de la MMU

- Conversion d'une adresse logique en adresse physique : 2 accès mémoire

→ cache du processeur : il contient les instructions et données les plus récemment accédées en mémoire centrale

→ Cache de la MMU : contient les associations (N° page, case de la mémoire physique) les plus récemment formés



Les caches du processeur et de la MMU

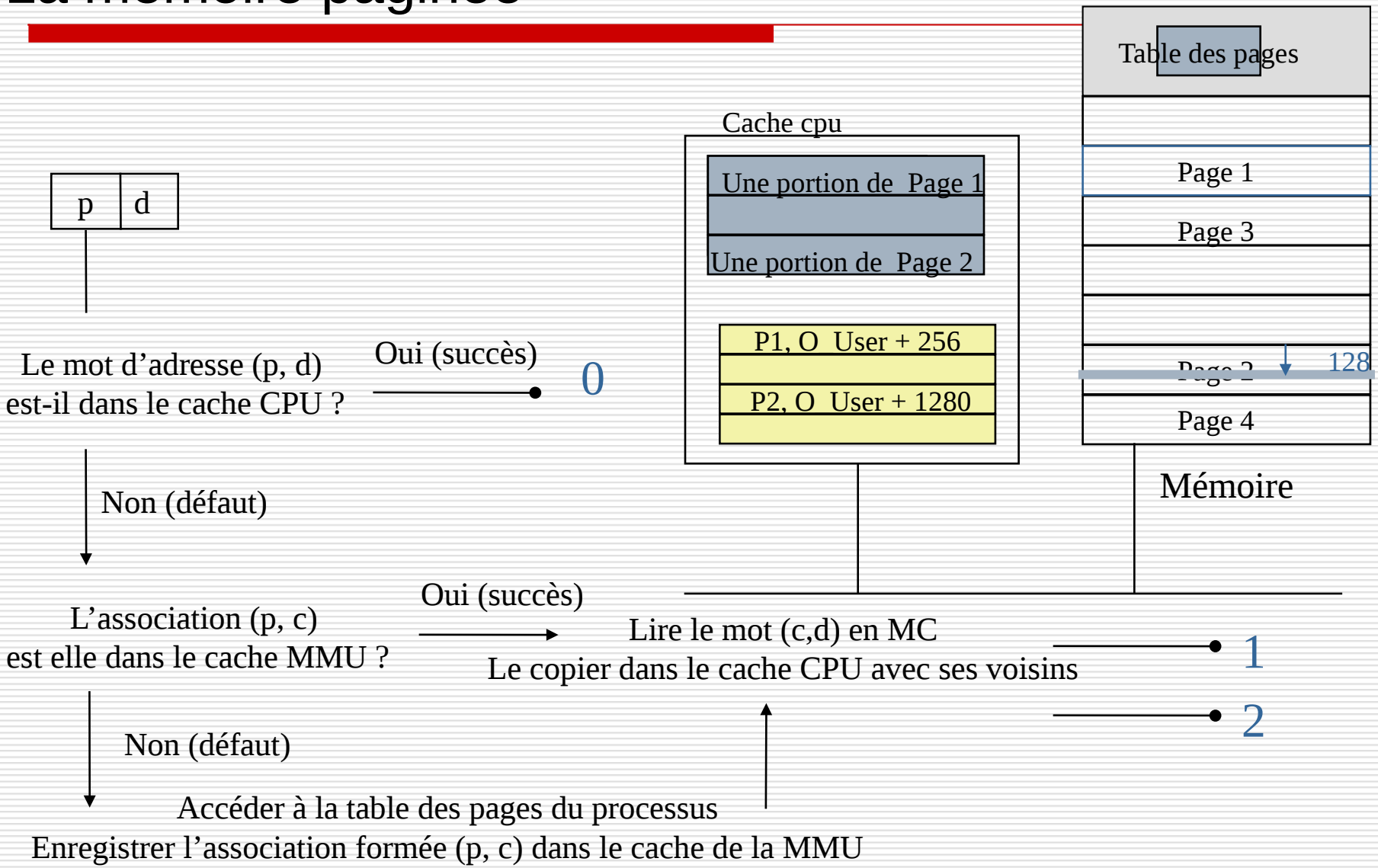
le cnam

L'utilisation de caches associatifs s'appuie sur les principes de localité d'exécution d'un programme :

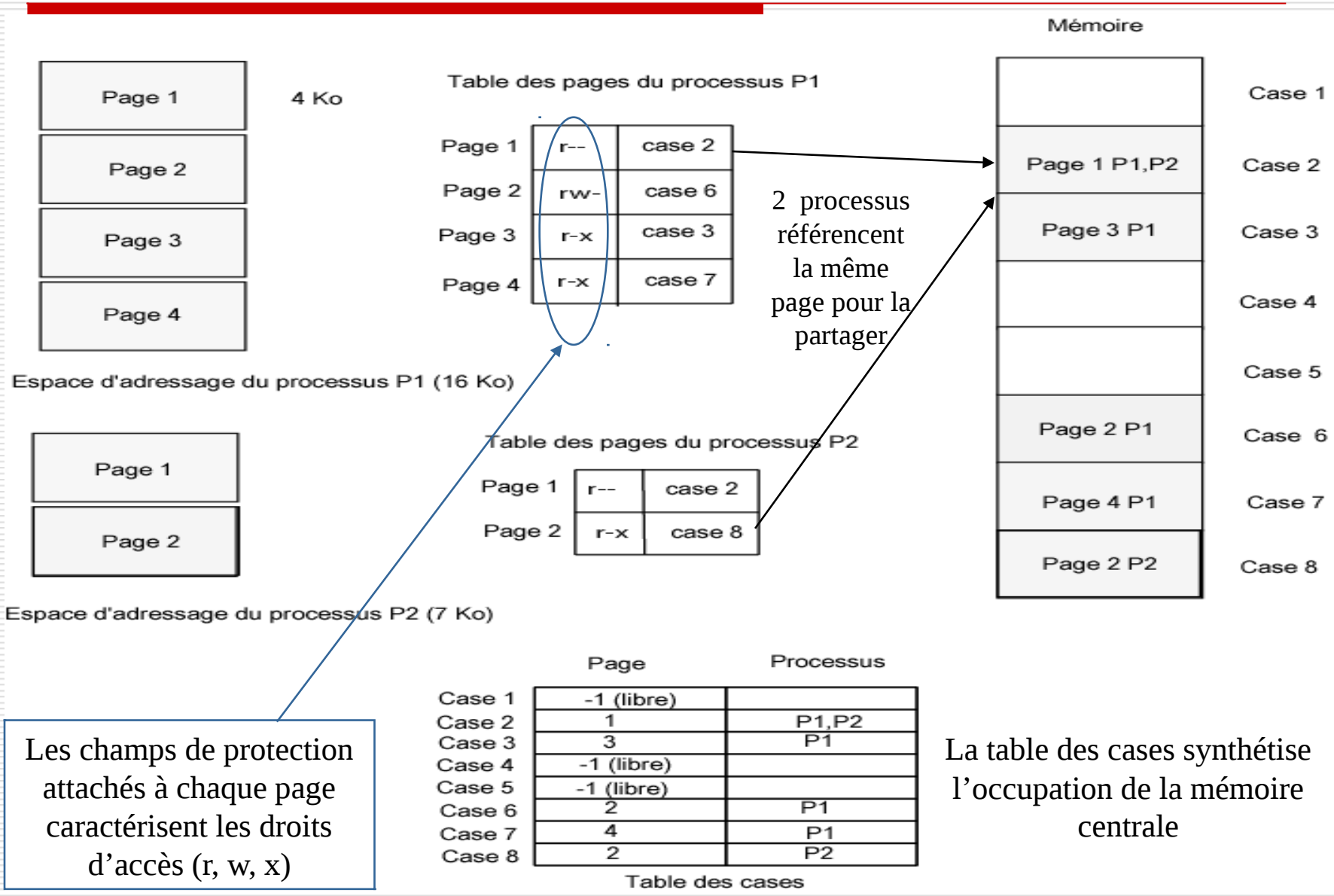
- **Localité temporelle** : si le processeur accède à l'instant t à l'adresse (p,d), la probabilité qu'il demande de nouveau accès à cette adresse (p,d) à l'instant t' très proche de t est grand
  - ↳ on enregistre le mot d'adresse (p,d) dans le cache du processeur
  - ↳ On mémorise l'association (p, c) dans le cache de la MMU avec c la case contenant la page p

- **Localité spatiale** : si le processeur accède à l'instant t à l'adresse (p,d), la probabilité qu'il demande accès à une adresse voisine dans cette même page (p,d') à l'instant t' très proche de t est grand
  - ↳ on enregistre le mot d'adresse (p,d) et des voisins dans le cache du processeur
  - ↳ On mémorise l'association (p, c) dans le cache de la MMU avec c la case contenant la page p

La mémoire paginée

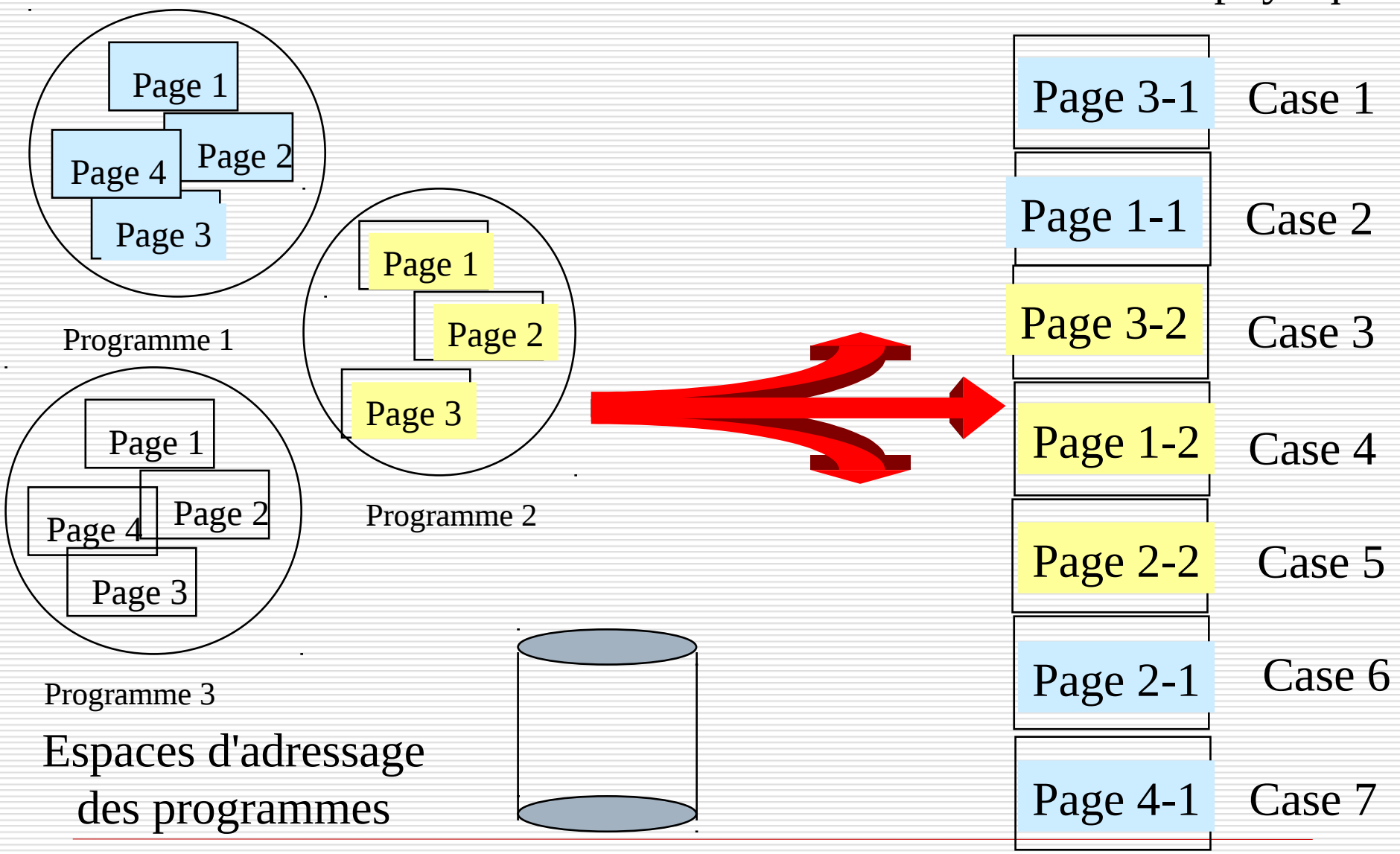


Protection et partage



La mémoire virtuelle

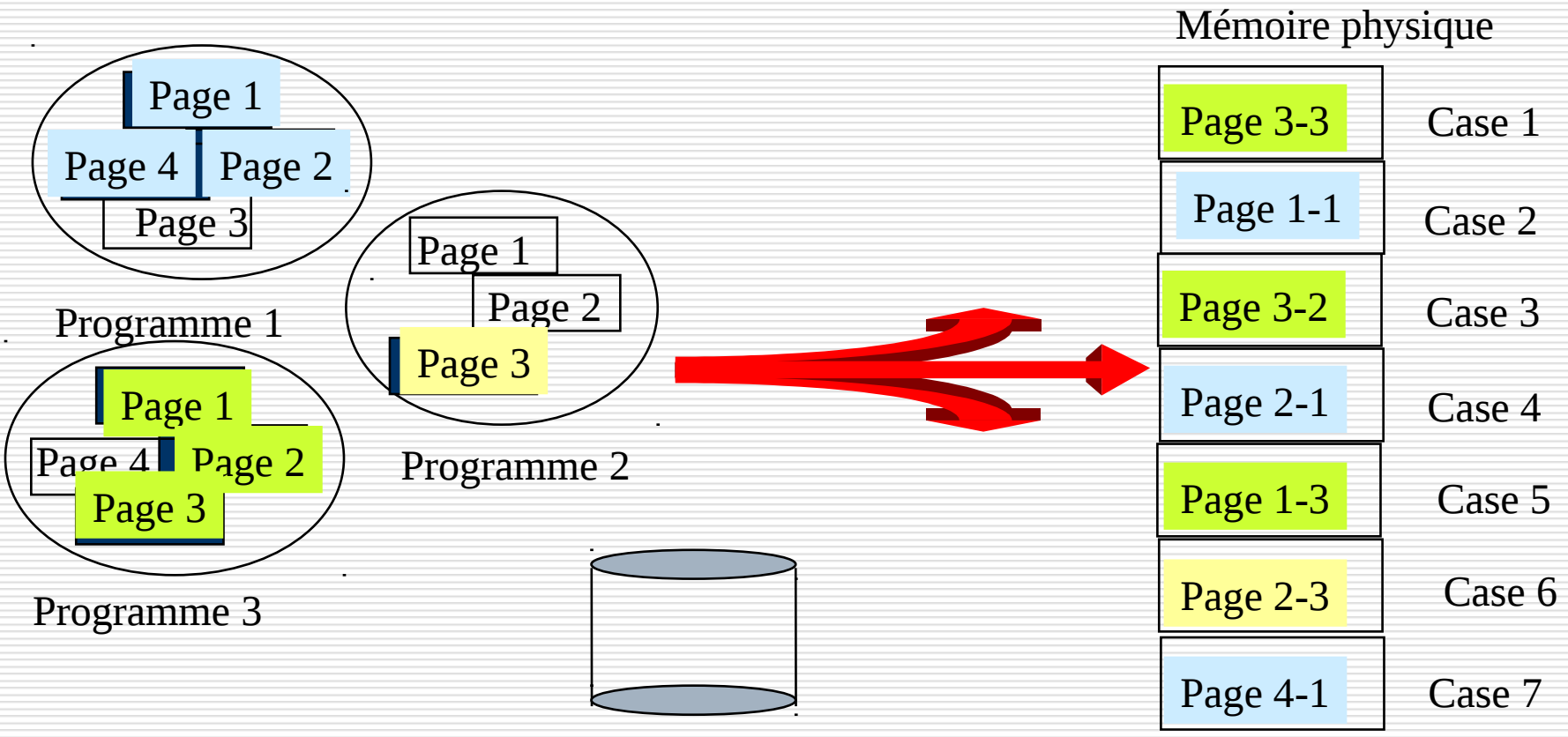
Mémoire virtuelle



Mémoire virtuelle

La capacité de la mémoire centrale est trop petite pour charger l'ensemble des pages des programmes utilisateurs :

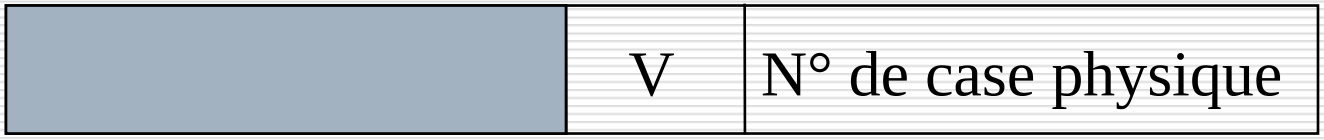
→ Ne charger que les pages utiles à un instant (principes de localité).



Bit de validation

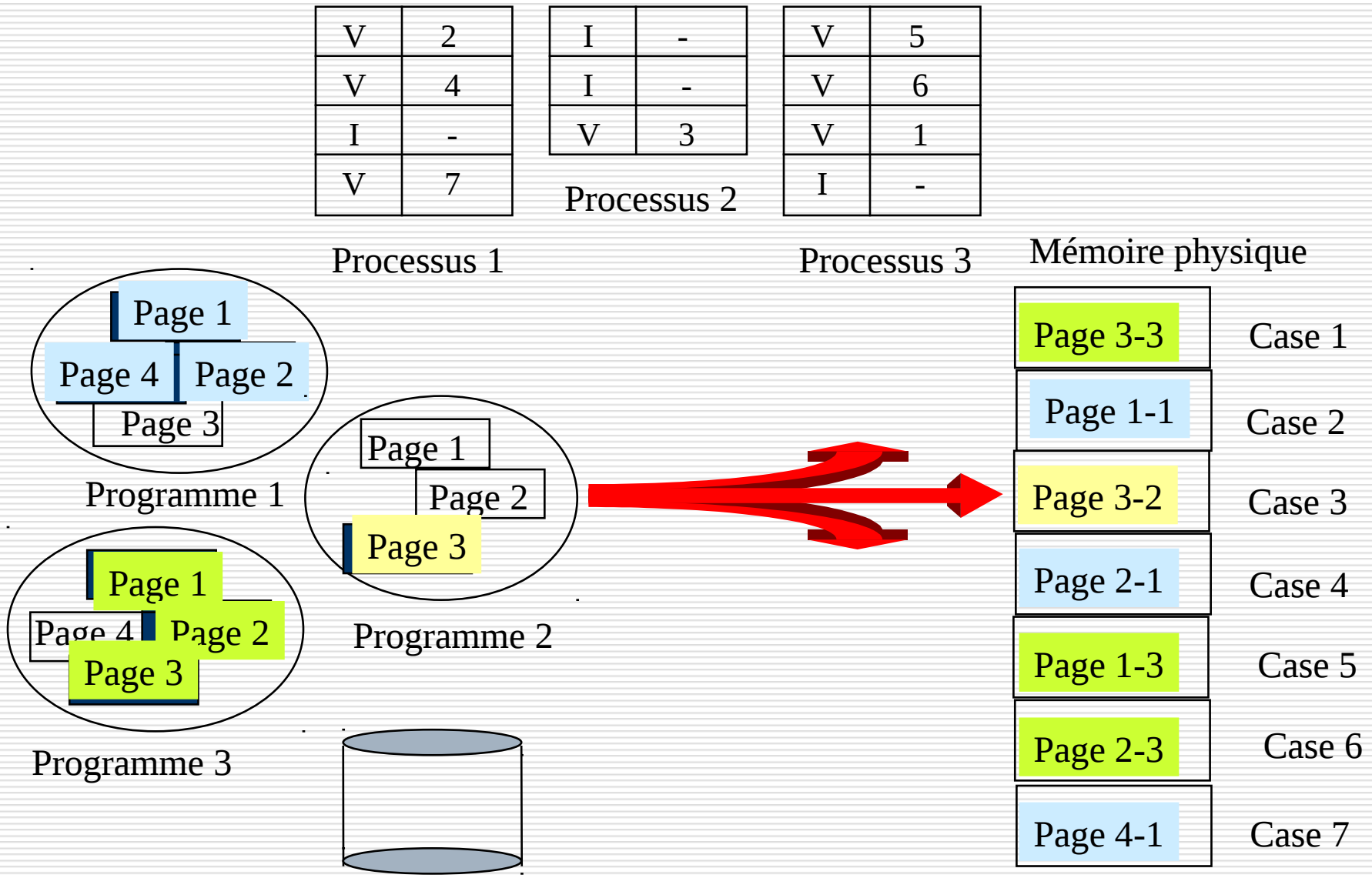
Ne charger que les pages utiles à un instant :

→ il faut pouvoir tester la présence d'une page en mémoire centrale



Bit validation à vrai si la page est présente en mémoire centrale

Bit de validation



Bit de validation et défaut de page

V	2
V	4
I	-
V	7

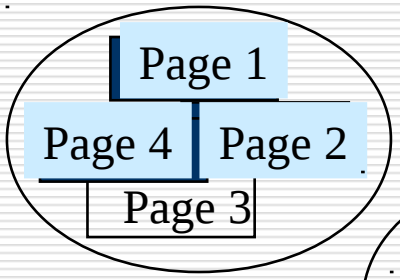
Processus 1

I	-
I	-
V	3

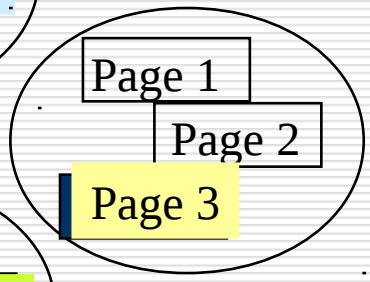
Processus 2

V	5
V	6
V	1
I	-

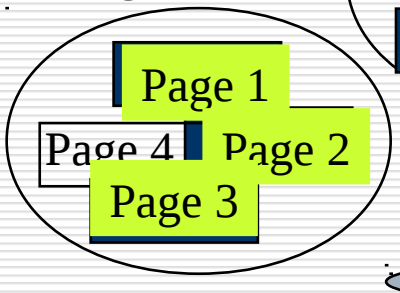
Processus 3



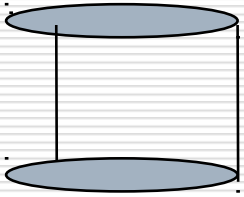
Programme 1

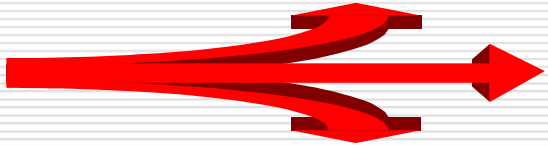


Programme 2



Programme 3





Processus 2 : accès à la page 2

DEFAUT DE PAGE

Mémoire physique

Page 3-3	Case 1
Page 1-1	Case 2
Page 3-2	Case 3
Page 2-1	Case 4
Page 1-3	Case 5
Page 2-3	Case 6
Page 4-1	Case 7

Bit de validation et défaut de page

le cnam

Ne charger que les pages utiles à un instant

il faut pouvoir tester la présence d'une page en mémoire centrale :

→ rôle du bit de validation

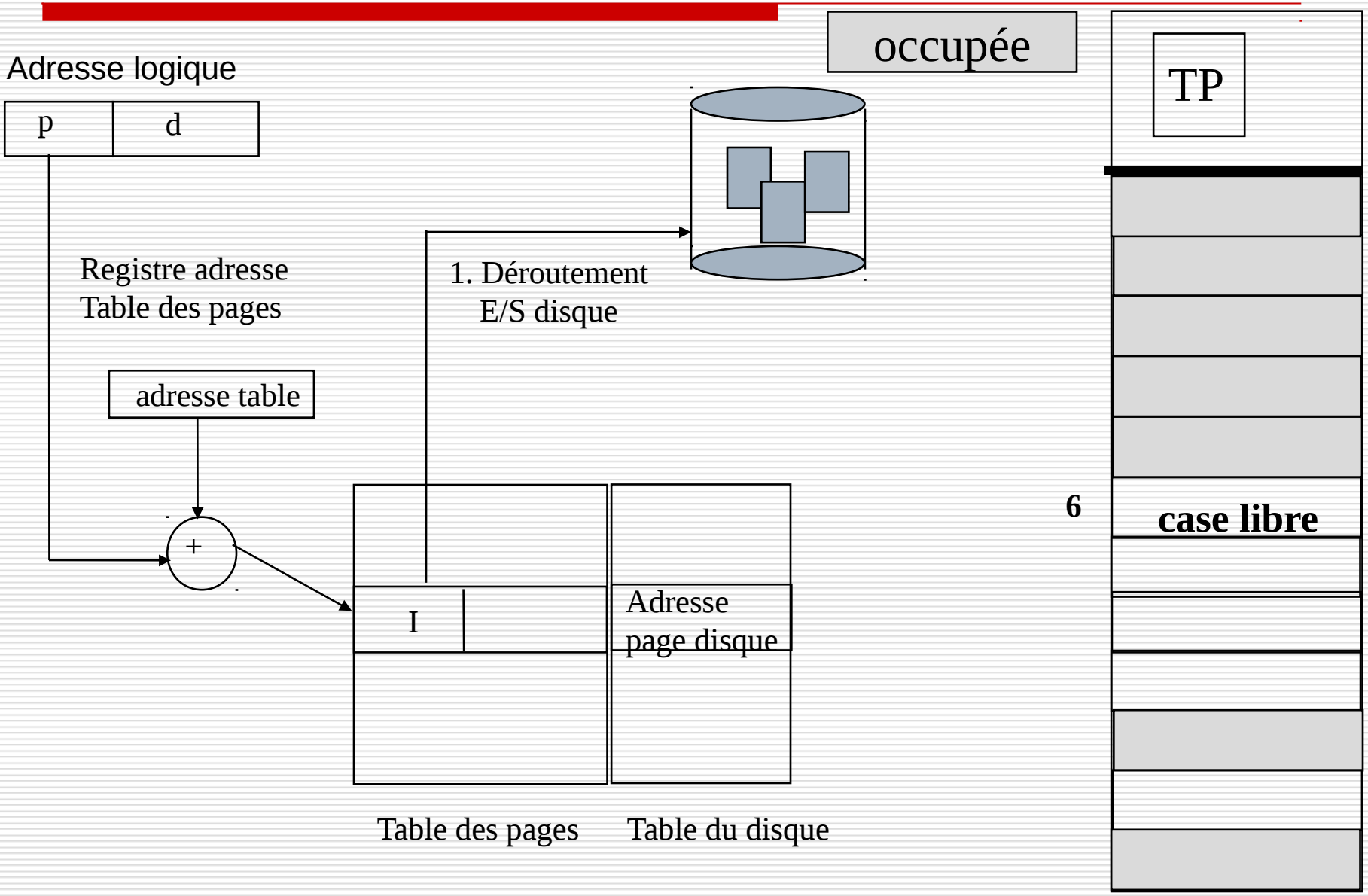
Si un processus cherche à accéder à une page non présente en mémoire centrale, il se produit un

déroutement de défaut de page :

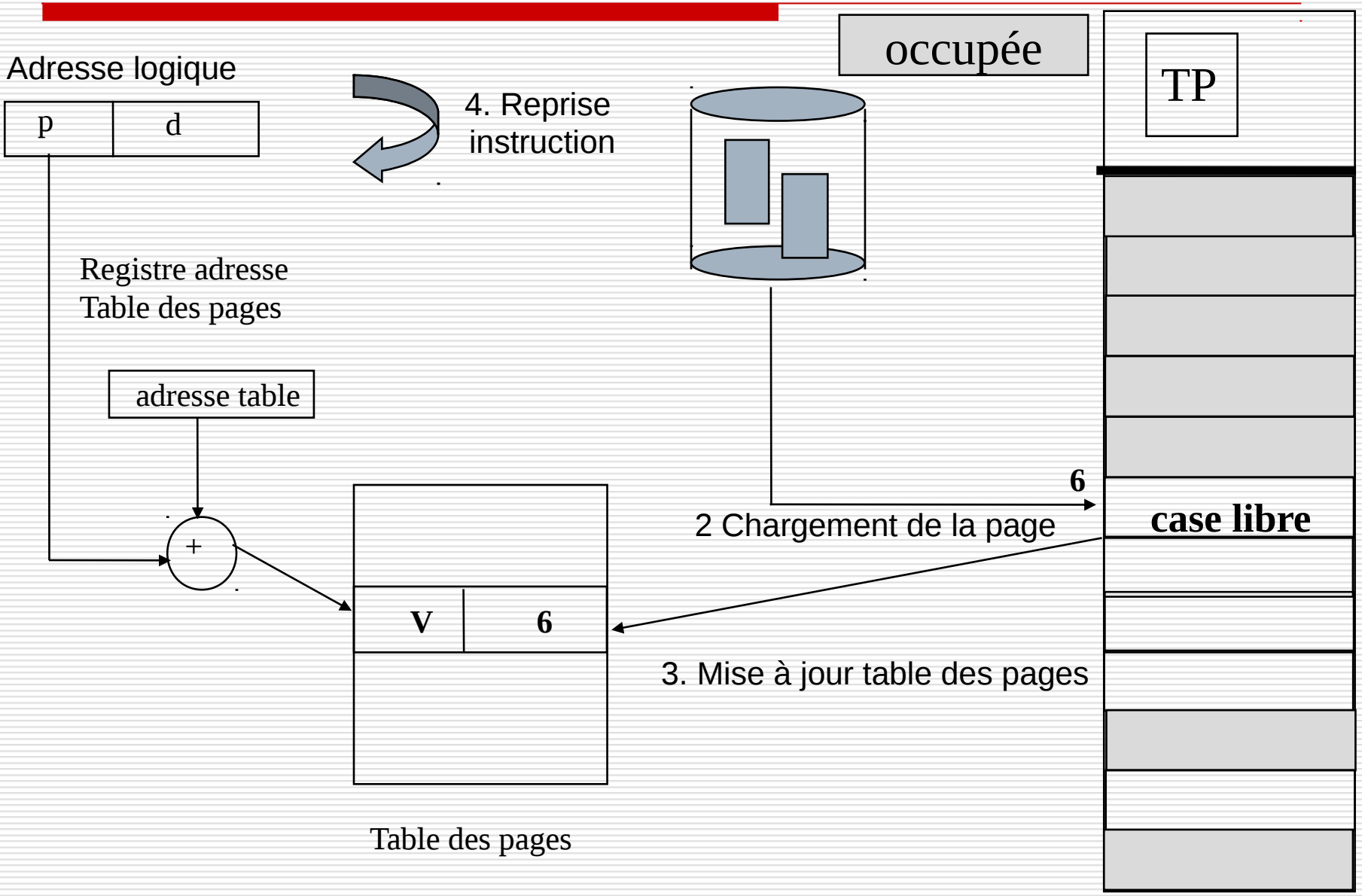
(1) Le système d'exploitation lance une entrée/sortie disque pour charger la page en mémoire dans une case libre.

(2) L'adresse de la page sur disque est stockée dans la table des pages.

Défaut de page



Défaut de page



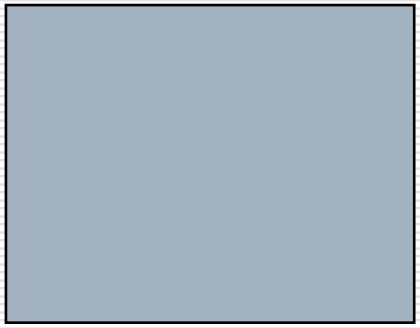
Chargement des pages : à la demande

le cnam

```
Pour i = 1 à 10
Faire
A(i) = 2;
B(i) = 5;
i = i + 1;
Fait
```

```
0 R1 ← 1
1 R2 ← 10
2 comp R1, R2
3 si R1 > R2 fin
4 A(R1)← 2
5 B(R1)← 5
6 R1 ← R1 + 1
7 aller à 2
```

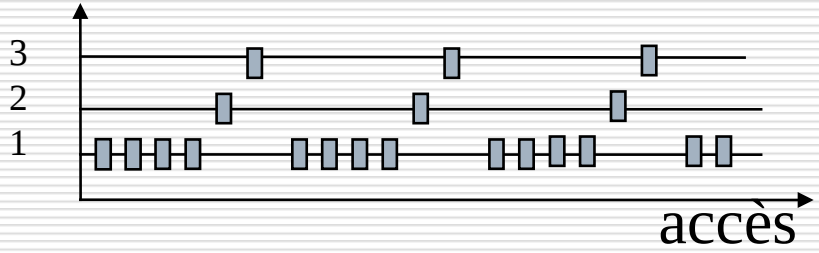
Page 1 code
Page de 10 mots



Page 2 vecteur A

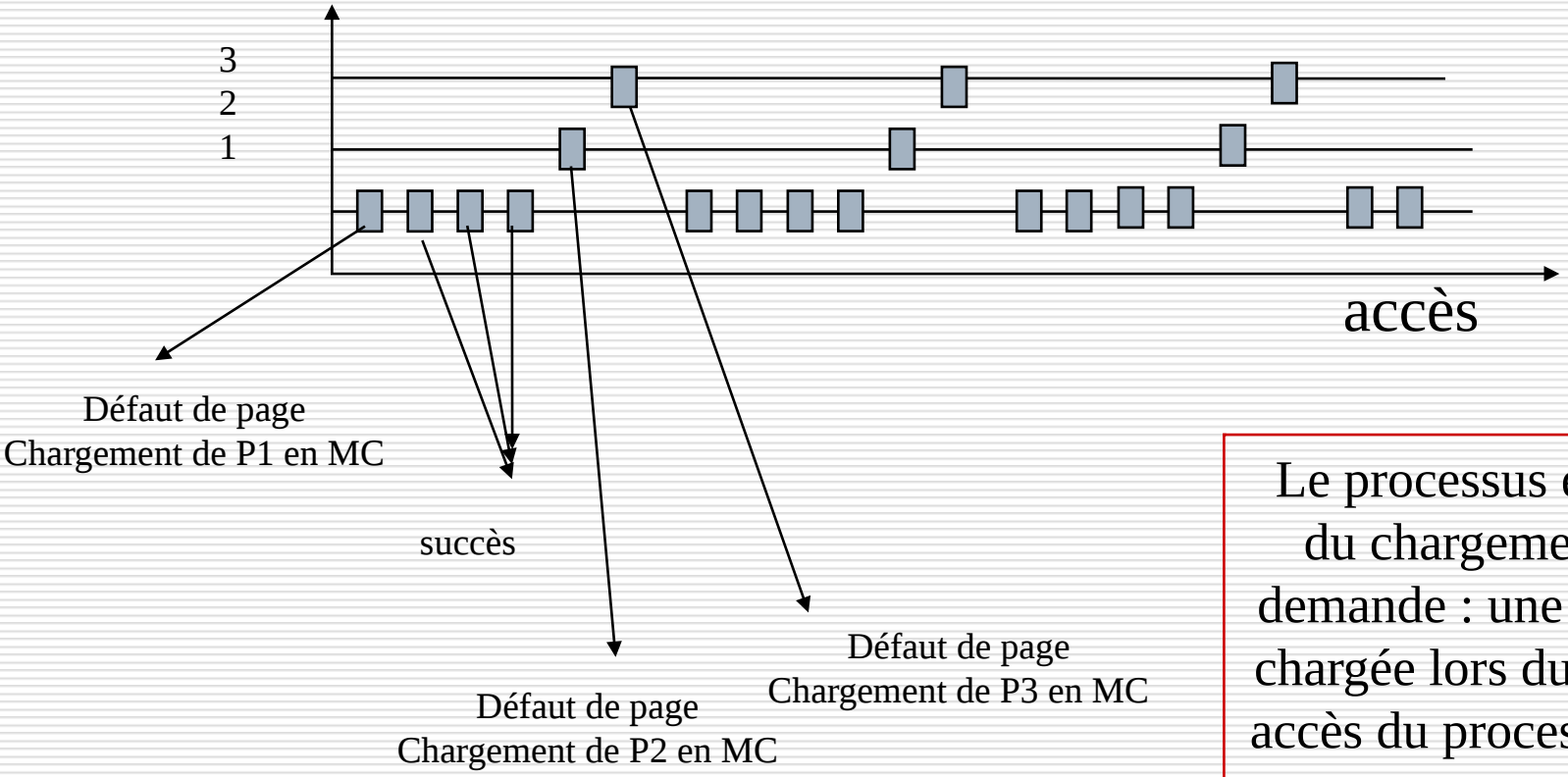
Page 3 vecteur B

Références aux pages : $11(112311)^{10}$



Chargement des pages : à la demande

Références aux pages : $11(112311)^{10}$



Le processus effectue du chargement à la demande : une page est chargée lors du premier accès du processeur à la page

Chargement de page

Lors d'un défaut de page, la page manquante est chargée dans une case libre

MAIS la totalité des cases de la mémoire centrale peuvent être occupées

Le système d'exploitation utilise un algorithme pour choisir une case à libérer.

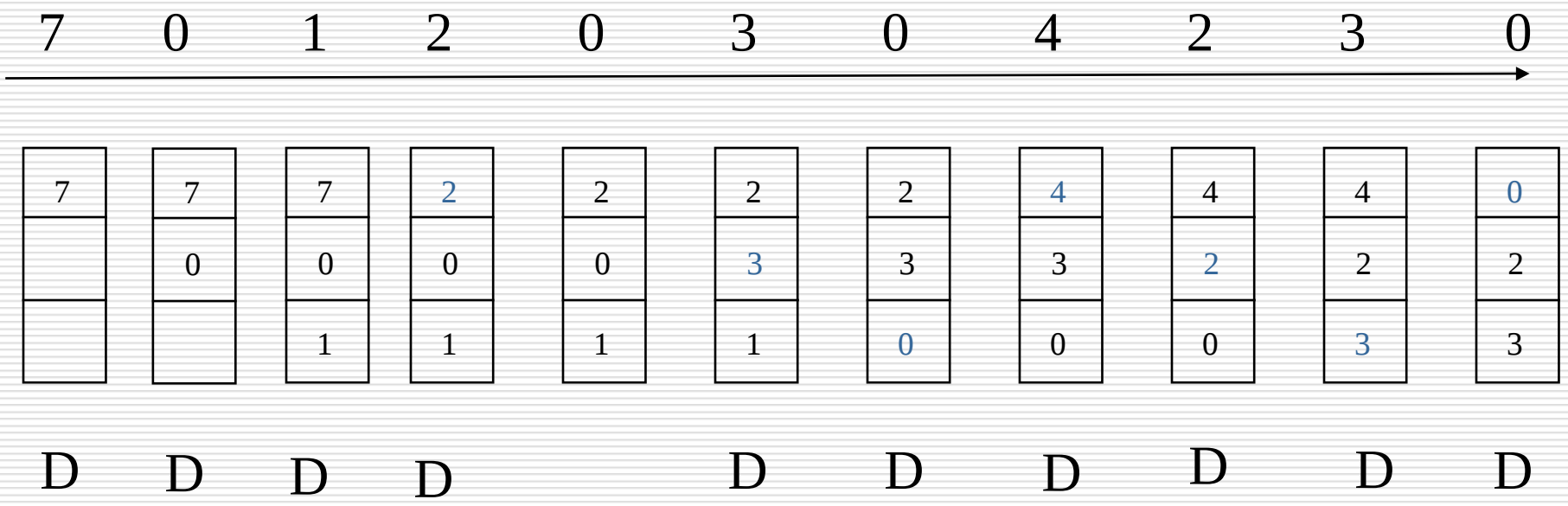
L'optimal est de retirer une page devenue inutile

- Au hasard
- FIFO (First In, First out)
- LRU (Least Recently Used) : moins récemment utilisée

Remplacement de page : FIFO

FIFO : la page la plus anciennement chargée est la page remplacée

Chaine de référence

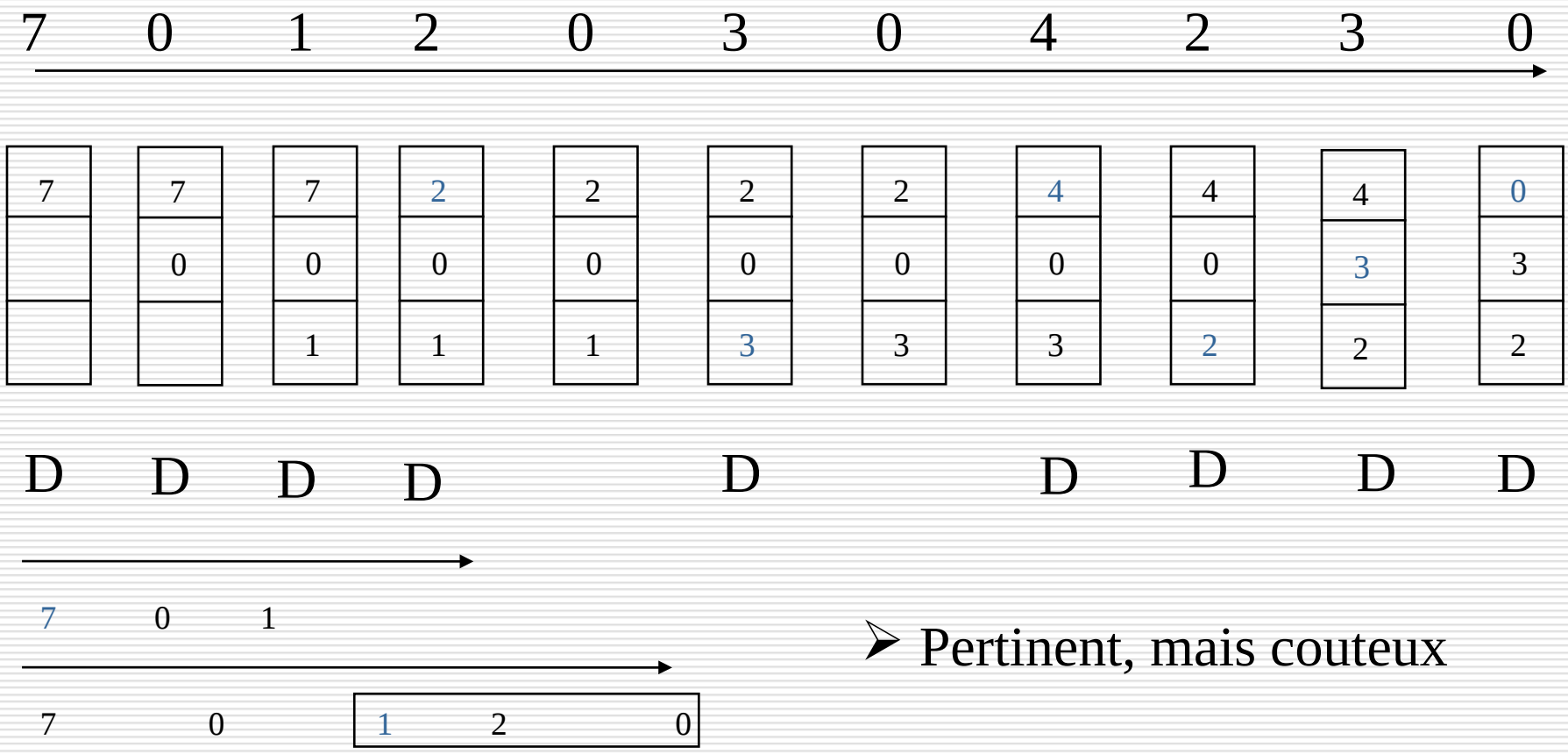


- Facile mais peu pertinent

Remplacement de page : LRU

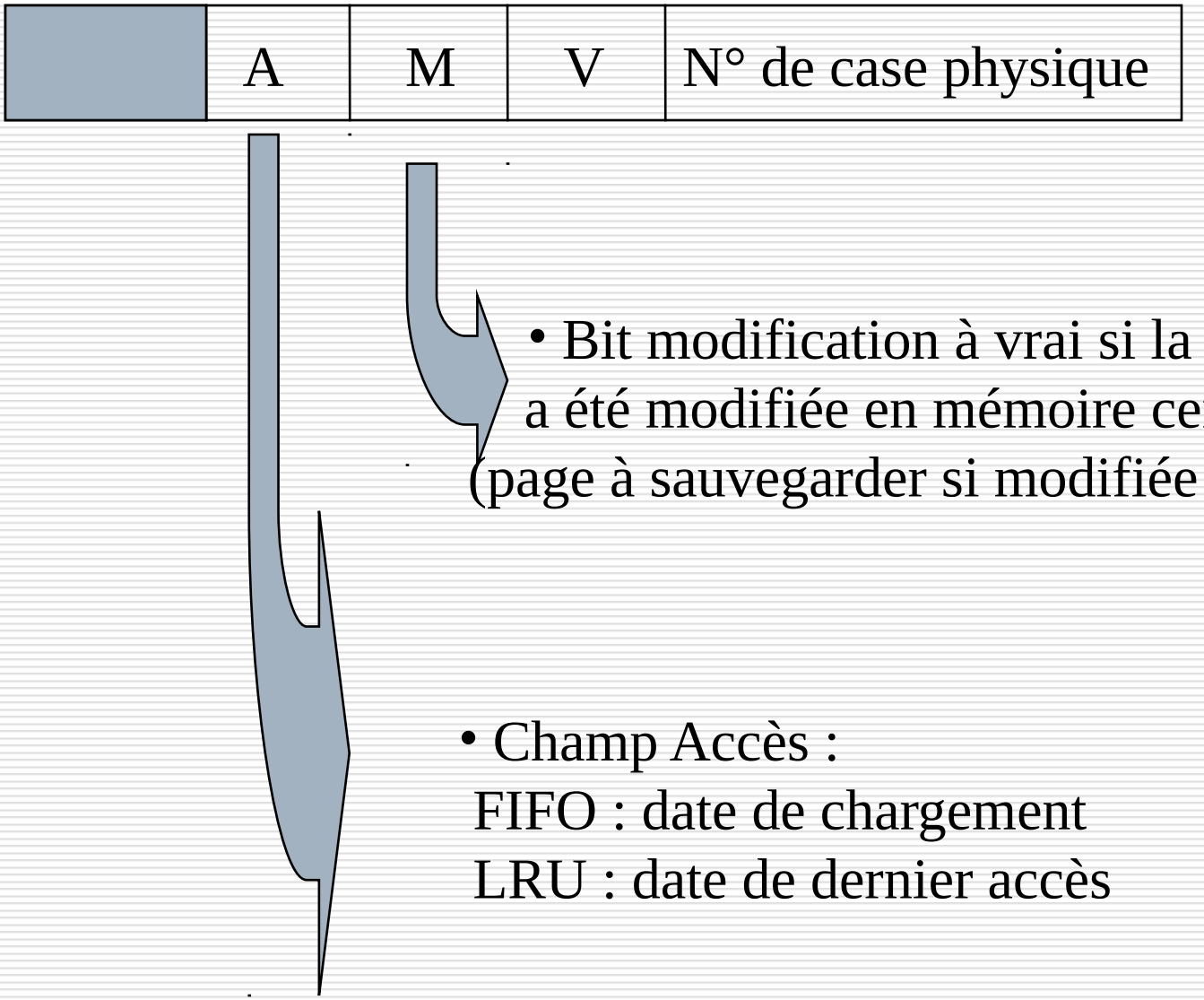
LRU : la page la moins récemment accédée est la page remplacée

Chaine de référence



- Pertinent, mais couteux

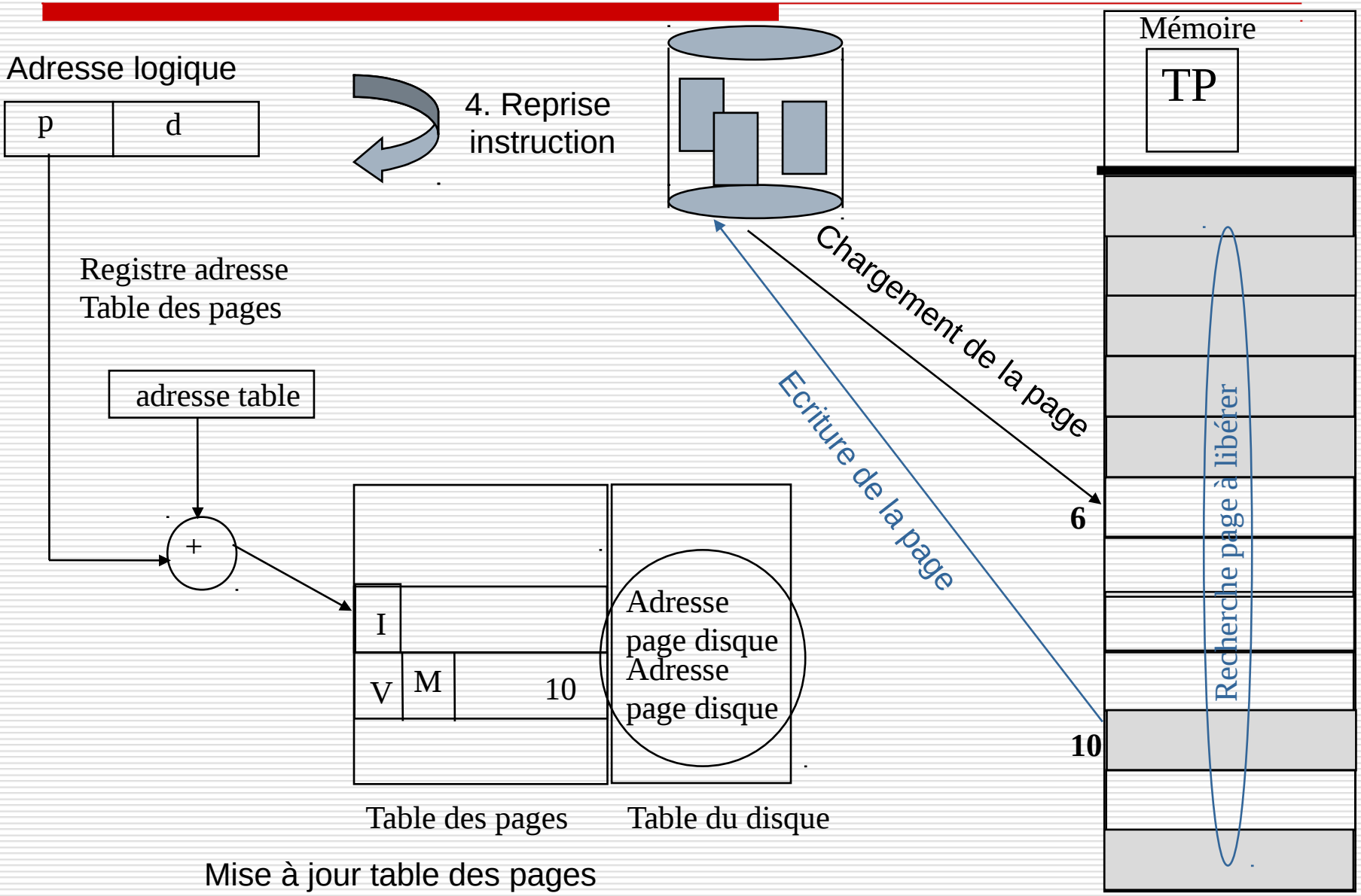
Algorithme de remplacement de page



- Bit modification à vrai si la page a été modifiée en mémoire centrale (page à sauvegarder si modifiée en MC)

- Champ Accès :
  FIFO : date de chargement
  LRU : date de dernier accès

Libération de page



Mise à jour table des pages

Exemple : linux

La mémoire vue par Linux

- L'espace d'adressage d'un processus est composé de régions
  - une région de code
  - une région des variables initialisées
  - une région des variables non initialisées
  - une région pour les codes et données des bibliothèques
  - une région pour la pile

- Une région est une zone contiguë de l'espace d'adressage traitée comme un objet pouvant être partagé et protégé. Elle est caractérisée par
  - ses adresses de début et de fin
  - les droits d'accès qui lui sont associés
  - l'objet qui lui est associé

- Une région est divisée en pages (4 Ko)

```
bbj>cat /proc/self/maps
08048000-0804a000 r-xp 00000000 03:02 7914
0804a000-0804b000 rw-p 00001000 03:02 7914
0804b000-08053000 rwxp 00000000 00:00 0
40000000-40005000 rwxp 00000000 03:02 18336
40005000-40006000 rw-p 00004000 03:02 18336
40006000-40007000 rw-p 00000000 00:00 0
40007000-40009000 r--p 00000000 03:02 18255
40009000-40082000 r-xp 00000000 03:02 18060
40082000-40087000 rw-p 00078000 03:02 18060
40087000-400b9000 rw-p 00000000 00:00 0
bfffe000-c0000000 rwxp fffff000 00:00 0
```

La table des pages

- Une entrée de table des pages contient :

Present	Accessed	Dirty	*Read/write*		Case

- Chaque case est décrit par un descripteur :

Adresses case libre précédente et suivante

Nombre de processus se partageant la page

État de la page (verrouillée, accédée, …)

Champ dirty (page modifiée)

Champ age

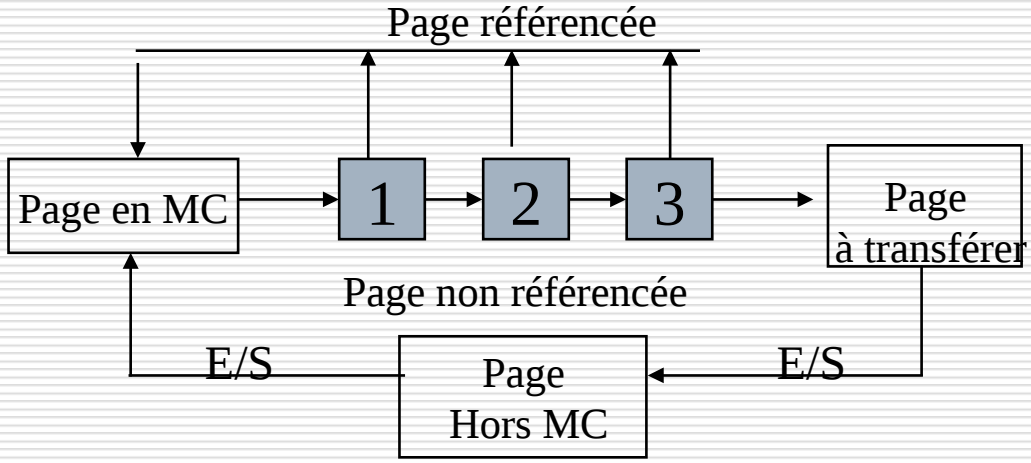
La table des pages

Les champs Accessed et Age sont utilisés par le processus "Dérobeur de Pages" pour choisir des victimes

- le dérobeur de pages est réveillé toutes les 10 secondes
- le dérobeur de pages libère des pages si le nombre de cases libres est tombé en dessous d'un seuil minimal
- Une page est victime si elle a atteint un age donné (paramètre système) sans être référencée.

La table des pages

- Les champs Accessed et Age sont utilisés par le processus "Dérobeur de Pages" pour choisir des victimes.
- A chaque référence par le processus, l'age de la page devient égal à 0 et le bit Accessed est mis à vrai
- A chacun de ses passages, le dérobeur de pages :
  - met à faux le bit Accessed si il est à vrai
  - incrémente l'age de la page
- Une page est victime si
  - Le bit Accessed est faux
- l'age limite est atteint (ici par exemple 3)



En mémoire	Accessed	Age
Accès P	v	0
Accès D	F	1
Accès D	F	2
Accès P	v	0
Accès D	F	1
Accès P	v	0
Accès D	F	1
Accès D	F	2
Accès D Transférée	F	3

Exercices

On considère trois processus PA, PB et PC qui disposent d'un espace d'adressage paginé, respectivement composé de 4, 2 et 5 pages.
La mémoire centrale est composée de 15 cases numérotées de 1 à 15. Chaque case a une capacité de 512 octets. Lors d'un défaut de pages, la page manquante est chargée **dans la case libre de plus petit numéro.**
A l'instant t, l'allocation des espaces d'adressage est la suivante :
Pour le processus PA, seules les pages P1, P2 et P3 sont chargées en mémoire centrale respectivement dans les cases 5, 2 et 1 ;
Pour le processus PB, seule la page P1 est chargée en mémoire centrale dans la case 10 ;
Pour le processus PC, seules les pages P1, P2 et P5 sont chargées en mémoire centrale respectivement dans les cases 4, 8 et 11.

Question 1
Représentez sur un schéma les structures de données (tables des pages et mémoire centrale) correspondant à l'allocation décrite.

Question 2
Le processus PA accède à l'adresse linéaire 804 dans son espace d'adressage Donnez l'adresse paginée puis l'adresse physique correspondante.
Le processus PB accède à l'adresse linéaire 804 dans son espace d'adressage Donnez l'adresse paginée puis l'adresse physique correspondante.
Le processus PC accède à l'adresse linéaire 2544 dans son espace d'adressage Donnez l'adresse paginée puis l'adresse physique correspondante.

Num de case	MC	Adresse de la case
1	PA,3	0
2	PA,2	512
3		1024
4	PC,1	1536
5	PA,1	2048
6		2560
7		3072
8	PC,2	3584
9		4096
10	PB,1	4608
11	PC,5	5120
12		5636
13		6144
14		6656
15		7168

PA		
1	V	5 : o_user + 2048
2	V	2 : o_user + 512
3	V	1 : o_user +0
4	I	

PB		
1	V	10 : o_user + 4608
2	I	

PC		
1	V	4 : o_user + 1536
2	V	8 : o_user + 3584
3	V	
4	I	
5	V	11 : o_user +5120

PA accède à l'adresse linéaire 804 qui correspond à l'adresse paginée (2,292).
L'adresse physique correspondante est
o_user + 512 + 292

Numéro de case	MC	Adresse de début de case
1	PA,3	0
2	PA,2	512
3	PB,2	1024
4	PC,1	1536
5	PA,1	2048
6		2560
7		3072
8	PC,2	3584
9		4096
10	PB,1	4608
11	PC,5	5120
12		5636
13		6144
14		6656
15		7168

PB		
1	V	10 : o_user + 4608
2	V	3 : o_user + 1024

PB accède à l'adresse linéaire 804 qui correspond à l'adresse paginée (2,292).

La page 2 de PB n'est pas chargée, il y a défaut de page. La page 2 de PB est chargée en MC dans la case libre de plus petit numéro : ici la case 3.

La table des pages de PB est mise à jour.

L'adresse physique correspondant est : o_user + 1024 + 292

Numéro de case	MC	Adresse de début de case
1	PA,3	0
2	PA,2	512
3	PB,2	1024
4	PC,1	1536
5	PA,1	2048
6		2560
7		3072
8	PC,2	3584
9		4096
10	PB,1	4608
11	PC,5	5120
12		5636
13		6144
14		6656
15		7168

PC		
1	V	4 : o_user + 1536
2	V	8 : o_user + 3584
3	V	
4	I	
5	V	11 : o_user +5120

PC accède à l'adresse linéaire 2544 qui correspond à l'adresse paginée (5,496).

L'adresse physique correspondant est : o_user + 5120 + 496