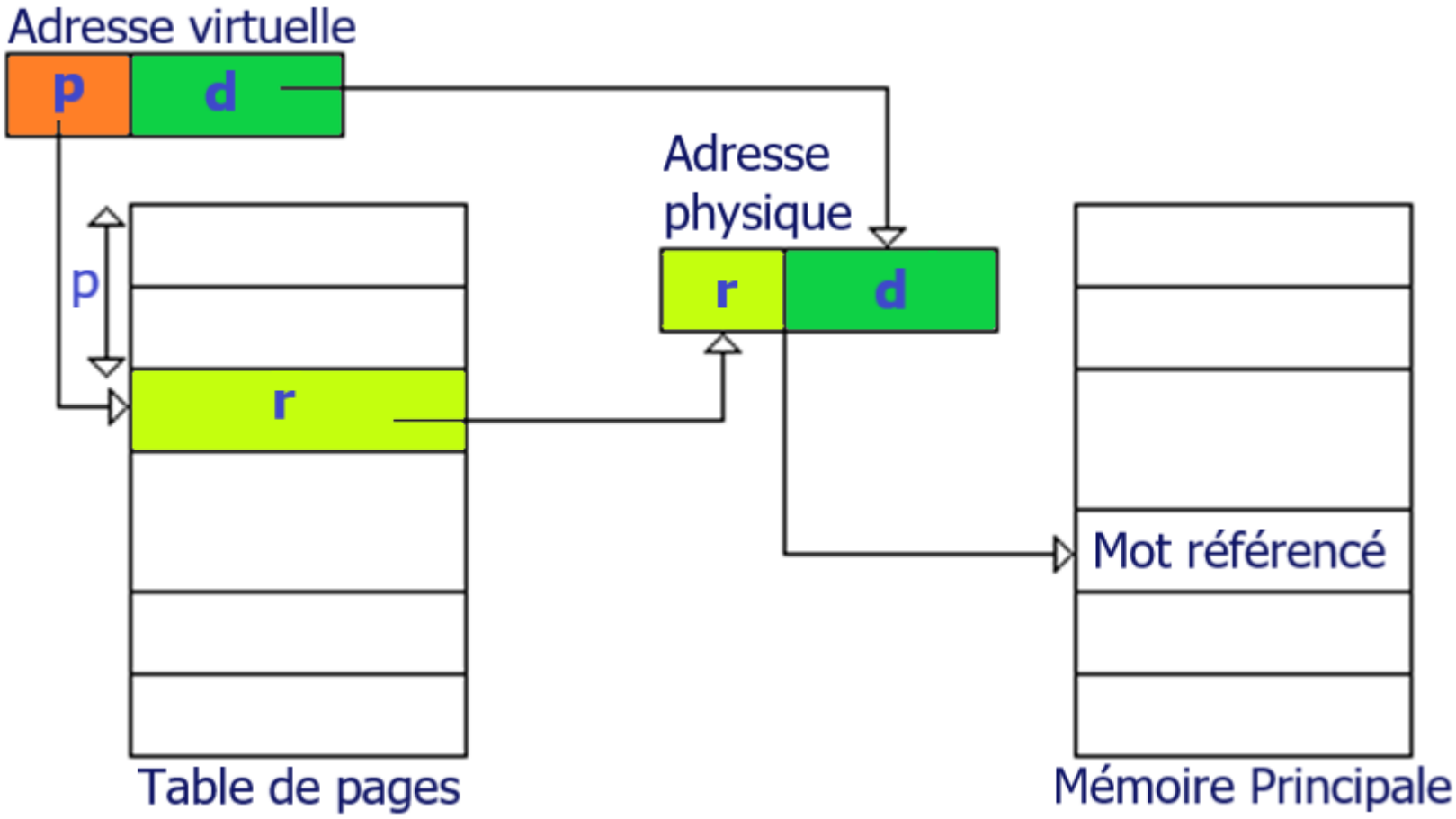
Cours de systèmes d'exploitation centralisés

Gestion de la mémoire virtuelle

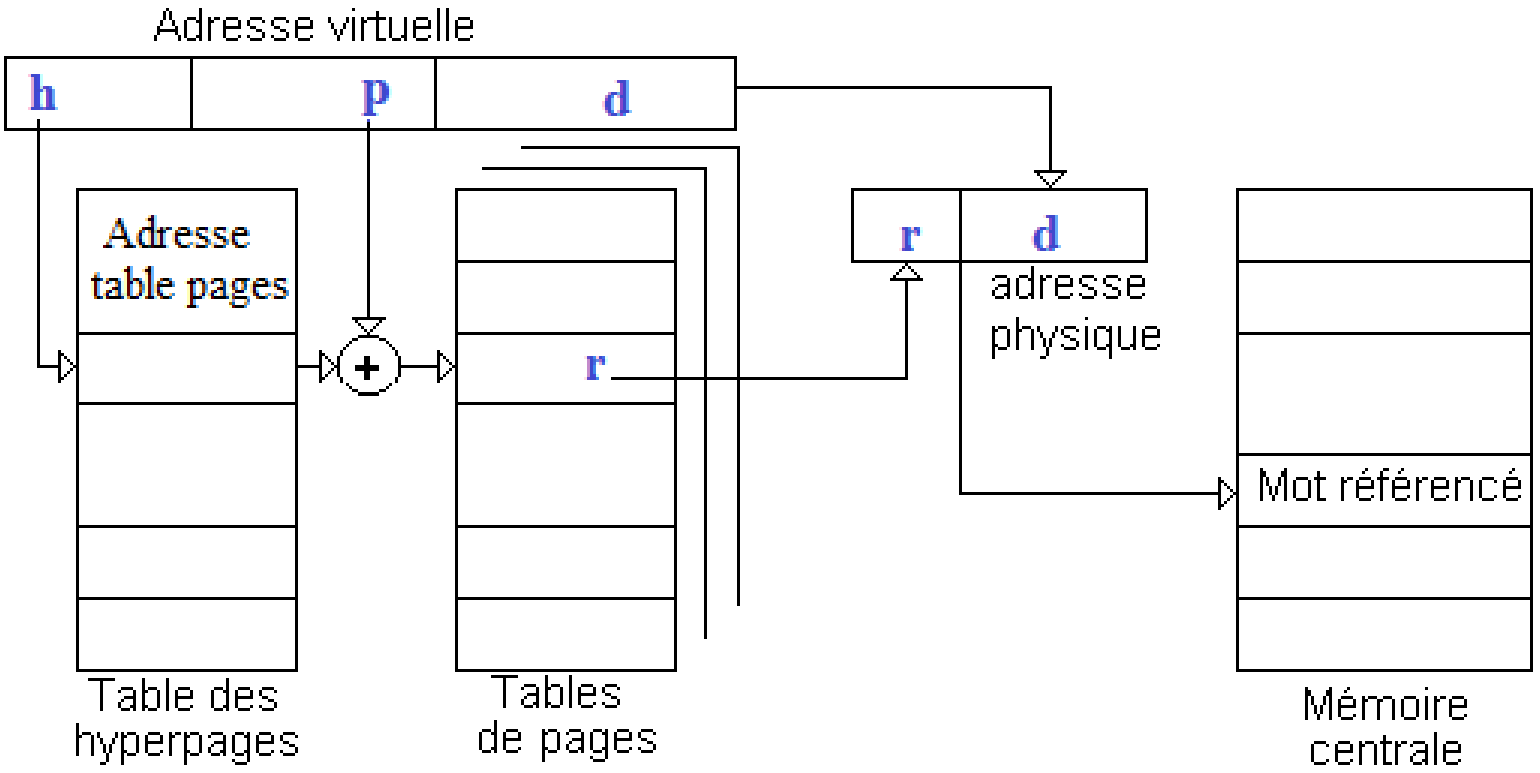
Séance 2

N.TEMGLIT, M.RAHMANI

Rappel (pagination simple)



Rappel: Pagination à deux niveaux



2.4 Table de pages inverse

- Quand l'espace d'adressage virtuel est très grand ➔ table de pages très grande.

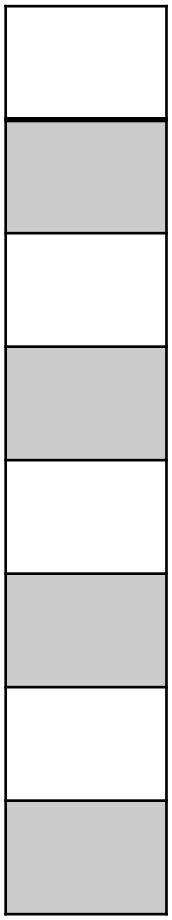
  Exemple: Adresse virtuelle 64 bits

  - ✓ Si page = 2^{12} octets et entrée = 2^2 octets ➔
  - ✓ Nombre d'entrées de la table de pages= $2^{64}/2^{12} = 2^{52}$ entrées
  - ✓ Taille de la table de page d'un processus= $2^{52}*2^2 = 2^{54}$ octets.
  - ✓ La plupart des machines actuelles(Intel) disposent de mémoire physique ne dépassant pas les 64 Giga-octets (2^{36} octets). ➔ Toute la mémoire centrale de la machine ne suffit pas pour charger une seule table de pages !

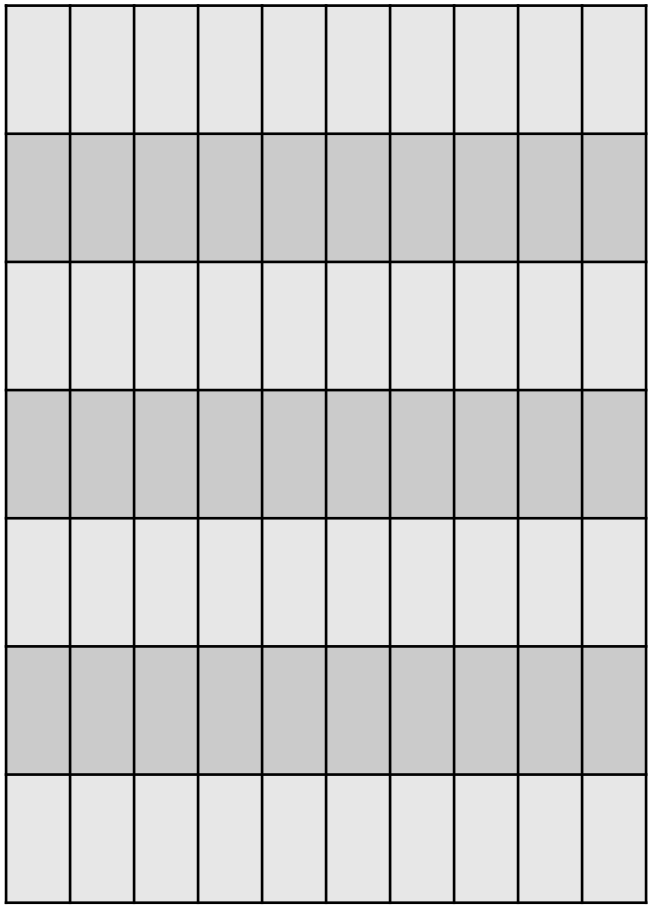
- La traduction des adresses virtuelles à l'aide d'une table de pages inverse est **une deuxième solution** au problème de la taille des tables de pages (pour réduire l'espace occupé par les tables de pages.

Table de page inverse

Table de page inverse
(n entrées)

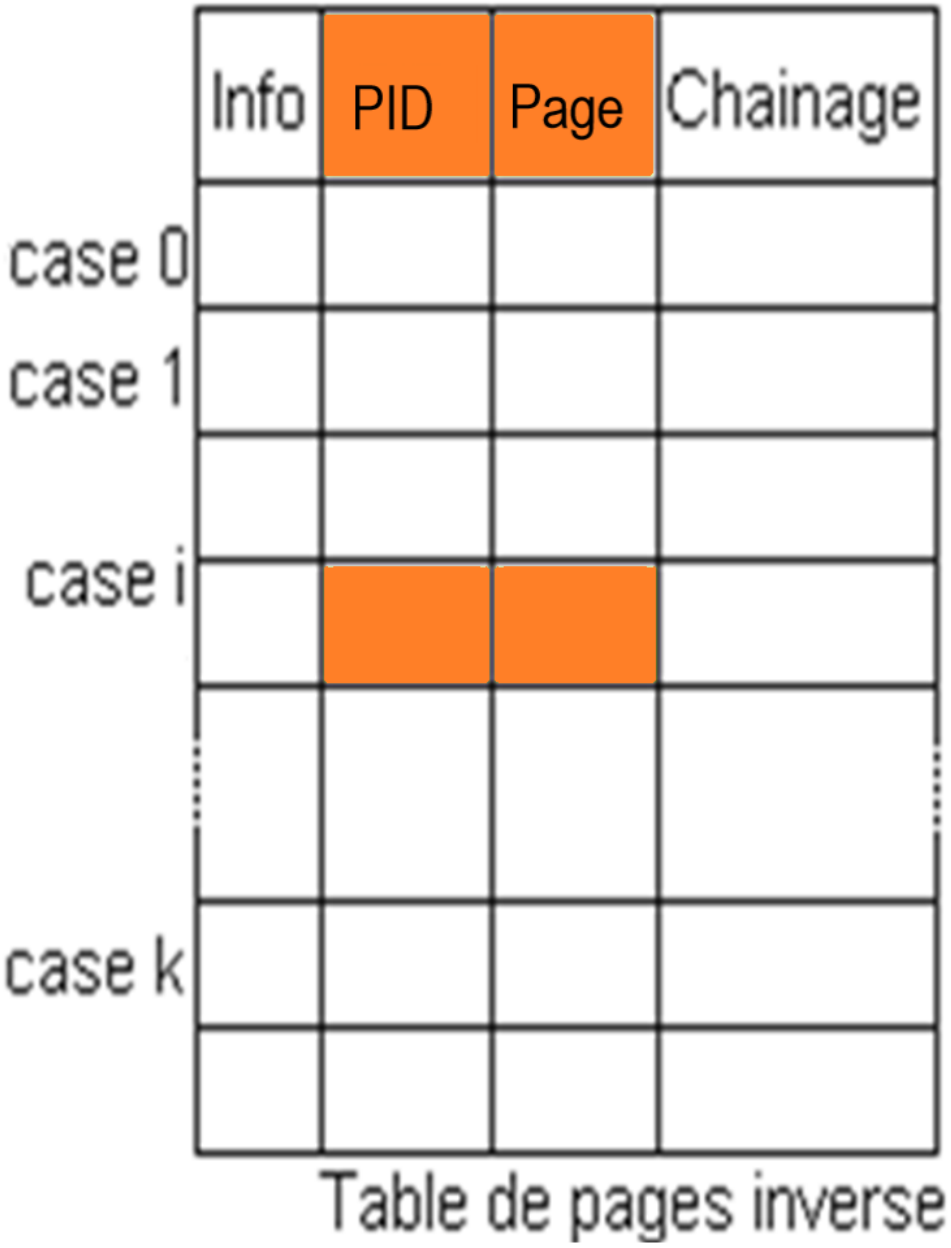


MC (n cases)



- **La table de page inverse décrit l'espace physique ou réel et non pas l'espace virtuel**.

- Le système dispose d'**une seule table des cases** que l'on appelle table de pages inverse qui décrit l'espace d'adressage physique (mémoire principale). ➔
  - Dans cette table, il y a une entrée par page physique ou case.
  - Une entrée de la table de pages inverse contient essentiellement:
    - ✓ L'identificateur **PID** du processus qui utilise la case et
    - ✓ La page **P** qui occupe la case(P appartient au processus PID).

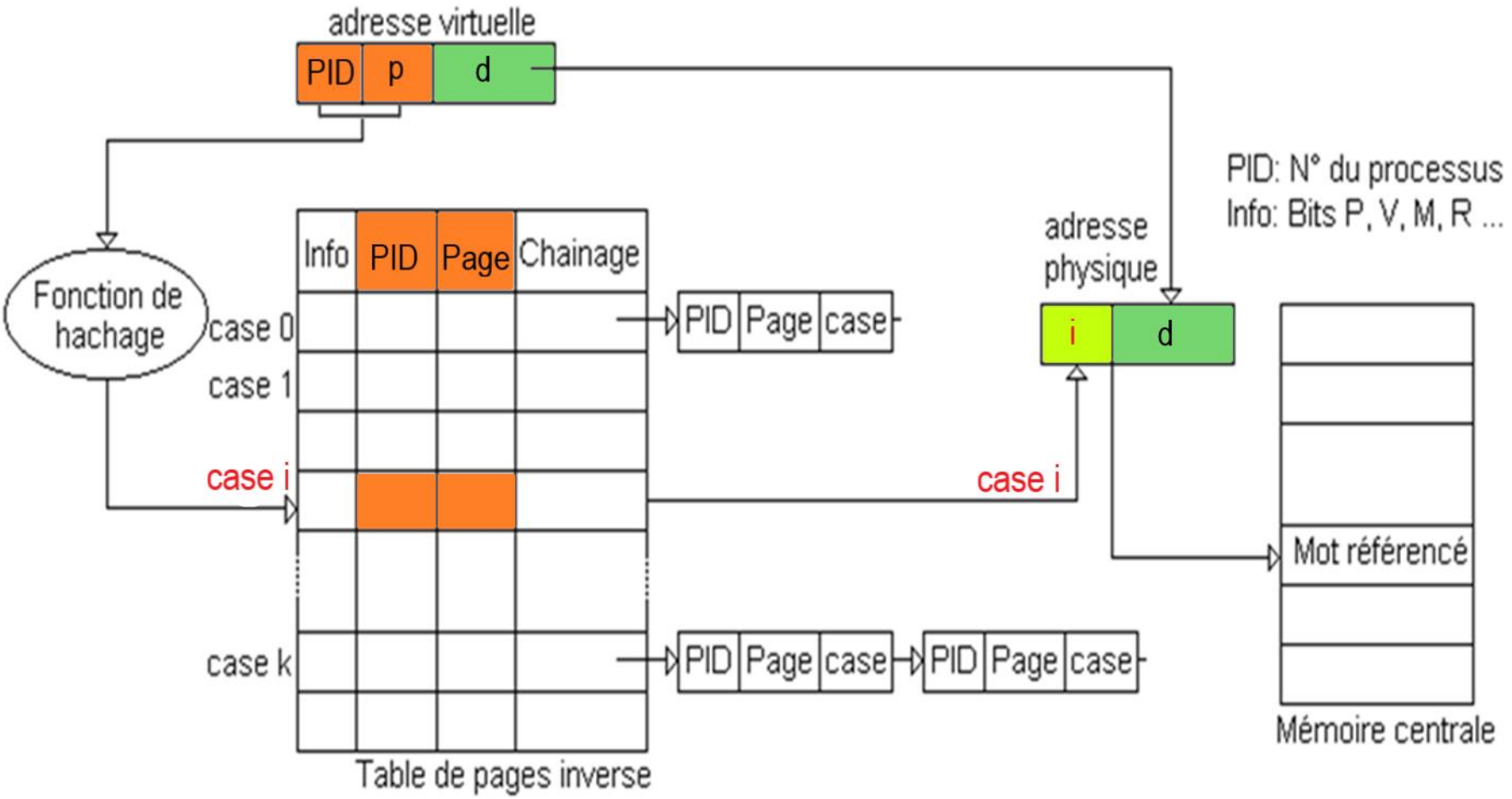

Info
PID
Page
Chainage
case 0
case 1
case i
case k
Table de pages inverse

- Le nombre d'entrées de la table de pages inverse est égal au nombre de cases de la mémoire centrale : Il est indépendant de la taille de l'espace d'adressage virtuel.
- La recherche d'une page P d'un processus PID est réalisée à l'aide d'une fonction de hachage (avec éventuellement un chaînage des entrées qui correspondent au même résultat de la fonction de hachage).
- **Adresse virtuelle**: le MMU dispose du PID du processus, en plus de l'adresse virtuelle.

Page	Déplacement

- Exemples de machines utilisant des tables de pages inverses: Le HP Spectrum, l'IBM System/38 et l'IBM RISC System/6000.

Traduction d'une adresse virtuelle en adresse réelle



Inconvénients :

- Traduction des adresses plus compliquée:

  Utilisation d'une fonction de hachage ➔

  Un ou plusieurs chainages ➔

- Partage de pages plus difficile à implémenter au niveau système.

2.5 Choix de la taille des pages

- La taille des cases d'une machine donnée est fixée par le constructeur.
- Le concepteur de systèmes d'exploitation choisit une taille de pages en fonction de ses objectifs
  - La taille choisie peut être différente de celle qui existe sur le matériel (ou fixé par le constructeur).
- Pour le choix de la taille de page optimale on doit tenir compte des paramètres suivants:

1. fragmentation interne

- La pagination permet de pallier le problème de la fragmentation externe : **la mémoire est allouée par case**.

- L'allocation par page fait apparaître de nouveau la fragmentation interne : La dernière page d'un espace virtuel est, en moyenne, utilisée à moitié, mais occupe une case entière.
- Dans un même espace virtuel on peut séparer le code et les données, on aura donc deux sections:
  Une section de code et une section de données.
- Pour des raisons de protection, chaque section commence sur une frontière de page ➔ on aura, en moyenne, deux demi-pages de fragmentation interne.
- La fragmentation augmente avec le nombre de sections commençant sur une frontière de page.
- **Plus les pages sont petites, plus la fragmentation interne est réduite.**

2. Taille de la table des pages

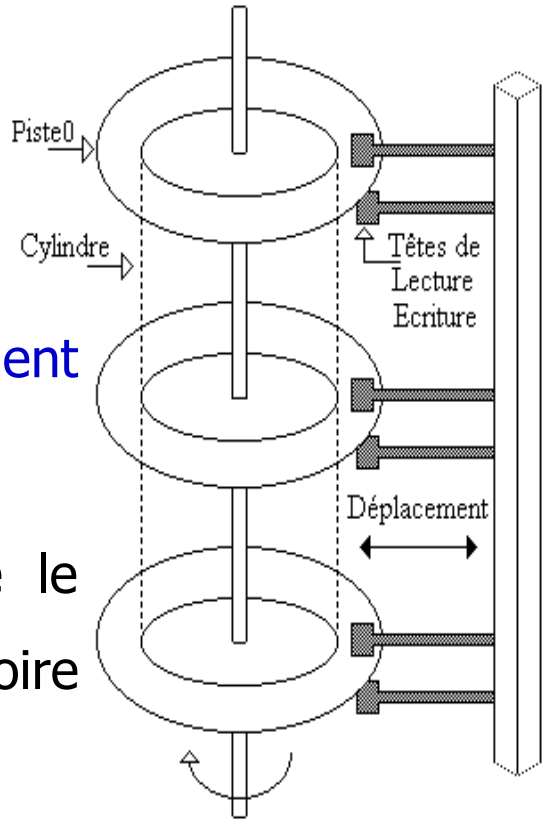
- La taille de la table de page est proportionnelle à la taille de l'espace virtuel. ➔

Pour réduire la taille de la table de pages, on doit utiliser des pages de taille assez grande.

- On peut constater que les paramètres (**1)** et (**2**) sont contradictoires.

3. Temps de lecture/écriture d'une page

- Le temps, de lecture/écriture d'une page à partir d'une mémoire secondaire, fait intervenir le temps de positionnement des têtes de lecture/écriture et le temps de recherche (délai de rotation) qui sont indépendants de la quantité d'information à lire ou à écrire.



Temps d'une entrée/sortie=temps de positionnement + délai de rotation + temps de transfert

- **Remarque**: Dans cette formule, on néglige le temps passé dans la file d'attente de la mémoire secondaire.

Exemple : Disque 10000 tours/minute

- Temps moyen de positionnement : 4,5 ms
- Délai de rotation(1/2 tour) pour un disque ayant une vitesse de 10000 tours/minute = 3 ms
- Taux de transfert : 75 Méga-octets/seconde;
  Transfert d'une page de 4ko= 4k/75Mo = 0,05181ms ➔0,052ms
- Temps moyen d'une entrée/sortie = 4,5 + 3 + 0,052= 7,352 ms.
- Le temps de transfert est proportionnel à la taille de la page, mais il ne constitue qu'une faible partie du temps total de lecture/écriture(**0,052 sur 7,352**).
- **Pour réduire le temps d'entrée/sortie, il est donc préférable d'utiliser de grandes pages (réduit le nombre d'E/S)**.

4. Allocation de la mémoire principale

- Pour une mémoire physique donnée, la taille des pages détermine le nombre de cases à allouer. ➔
- **Utiliser des pages plus petites sur des machines dont la mémoire physique est petite ➔ plus de cases à allouer, optimise le degrés de multiprogrammation.**

Exemples de tailles de pages

Machine	**Taille de pages**
IBM/370	2ko et 4ko
DEC Vax 8800	512 octets
Motorola 68030	256 octets à 32 k octets
Intel 80386	4ko
Intel 80x86(IA32)	4ko, 2Mo et 4Mo
IA32e (Intel 32-64bits)	4ko, 2Mo et 1Go
Sun UltraSPARC	8ko à 4Mo

Résumé

1)	Fragmentation interne	Petite taille
2)	Taille de la table de pages	Grande taille
3)	Temps de lecture/écriture	Grande taille
4)	Allocation des cases : petite Mémoire centrale	Petite taille

2.7 Protection de la mémoire paginée

- La protection d'une page peut être :
  - Page en lecture/écriture,
  - Page en lecture seule,
  - Page en lecture/exécution.
- A chaque page sont associés des bits de protection qui font partie de la table de pages.

Exemples :

- IBM/370 : Verrou=4 bits/page et Clé = 4bits
- CII-10070 : Verrou=2 bits/page et Clé = 2bits

2.8 Partage du code et des données (partage de pages)

- La pagination permet de partager, entre plusieurs processus, des pages de code ou de données.

Exemple : Editeur de texte partagé

- Un système supporte 40 utilisateurs,
- Espace virtuel d'un processus = 70k octets
  - 20 ko : code (éditeur de textes) et
  - 50 ko : espace de données.
- On suppose que l'espace virtuel d'un processus est entièrement chargé en mémoire centrale.
- Pour ces 40 utilisateurs on aura besoin de : (20+50)*40=2800ko.

	Processus1	Processus2	Processus40
Page0	Editeur	Editeur	Editeur
Page1	Editeur	Editeur	Editeur
Page2	Editeur	Editeur	Editeur
Page3	Editeur	Editeur	Editeur
Page4	Editeur	Editeur	Editeur
Page5	Données	Données	Données
…			
Page17	Données	Données	Données

- Si l'éditeur est partageable(réentrant) ➔
  Il peut alors être partagé par les 40 utilisateurs ➔
  - On ne charge qu'une seule copie du code de l'éditeur en mémoire centrale.
  - Chaque processus a ces propres données.

Comment réaliser le partage des pages ?

- Le partage est réalisé en plaçant dans la table de pages de chacun des processus les N° de cases correspondant à notre éditeur de texte.

  Les N° de cases sont les mêmes dans toutes les tables de pages qui partagent ces cases.

  Les N° de pages peuvent être différents.

Table de Pages de **P0**

0	
1	
2	1000
3	1001
4	1005
5	1006
6	1007
…	
n	

Mémoire centrale

Case0				
Case1000	Editeur	Editeur		
Case1004		Editeur	Editeur	Editeur

Table de pages de **P39**

0	1000
1	1001
2	1005
3	1006
4	1007
5	
…	
k	