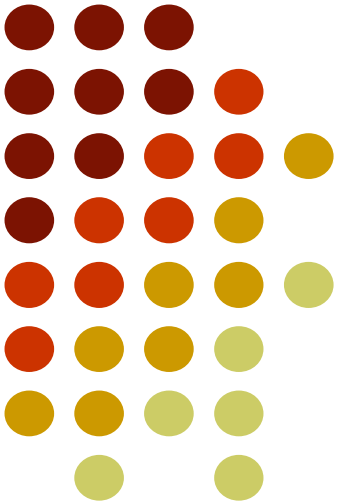
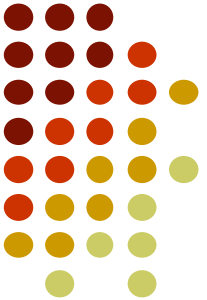
La Gestion de la mémoire

Cours construit à partir de ressources présentes sur le web, notamment Wikipedia



Plan du cours



1. Les types de mémoires
2. Organisation de la mémoire centrale à l'exécution d'un programme
3. Mémoire cache
4. Mémoire virtuelle

Les mémoires de l’ordinateur

On appelle « **mémoire** » tout dispositif capable d'enregistrer, de conserver et de restituer des informations. On distingue ainsi deux grandes catégories de mémoires :

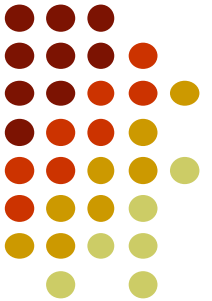
- **la mémoire centrale** (ou mémoire interne) permettant de mémoriser temporairement les données et les programmes lors de l'exécution les applications. La mémoire centrale est réalisée à l'aide de micro-conducteurs, c'est-à-dire des circuits électroniques spécialisés rapides. La mémoire centrale correspond à ce que l'on appelle la mémoire vive.

Les mémoires de l'ordinateur

On appelle « **mémoire** » tout dispositif capable d'enregistrer, de conserver et de restituer des informations. On distingue ainsi deux grandes catégories de mémoires :

- la **mémoire de masse** (appelée également *mémoire physique* ou *mémoire externe*) permettant de stocker des informations à long terme, y compris lors de l'arrêt de l'ordinateur. La mémoire de masse correspond aux dispositifs de stockage
  - magnétiques, tels que le disque dur,
  - optiques, comme les CD-ROM ou les DVD-ROM,
  - sur mémoires flash comme les clés USB, les cartes mémoire des appareils électroniques portables (photo, téléphone, PDA, micro-ordinateur ultra-portables), les disques SSD,
  - mémoires mortes.

Caractéristiques techniques



Les principales caractéristiques d'une mémoire sont les suivantes :

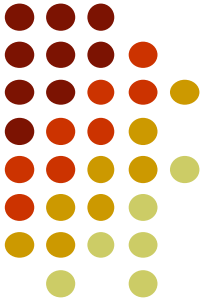
- **Adresse** : c'est la valeur numérique désignant un élément physique de mémoire
- **Capacité** ou taille : c'est le volume global d'informations (en bits) que la mémoire peut stocker
- **Temps d'accès** : c'est le temps qui s'écoule entre le lancement d'une opération d'accès (lecture ou écriture) et son accomplissement.
- **Temps de cycle** : c'est le temps minimal s'écoulant entre deux accès successifs à la mémoire. Il est plus long que le temps d'accès.
- **Débit** : c'est le nombre d'informations lues ou écrites par seconde
- **Volatilité** : elle caractérise la permanence des informations dans une mémoire. Une mémoire volatile perd son contenu quand on coupe le courant.

Ainsi, la mémoire idéale possède une grande capacité avec des temps d'accès et temps de cycle très restreints, un débit élevé et est non volatile.

Types de mémoire

- La **mémoire vive**, généralement appelée **RAM** (*Random Access Memory)* est la mémoire principale du système ; c'est un espace permettant de stocker de manière temporaire les données et les programmes.
- La **mémoire morte**, appelée **ROM** (*Read Only Memory*) est un type de mémoire permettant de conserver les informations qui y sont contenues même lorsque la mémoire n'est plus alimentée électriquement. Ce type de mémoire ne peut être accédée qu'en lecture. Toutefois il est désormais possible d'enregistrer des informations dans certaines mémoires de type *ROM*.
- La **mémoire flash** est un compromis entre les mémoires de type RAM et les mémoires mortes. En effet, la mémoire Flash possède la non-volatilité des mémoires mortes tout en pouvant facilement être accessible en lecture ou en écriture. En revanche les temps d'accès des mémoires flash sont plus importants que ceux de la mémoire vive.

Caractéristiques techniques



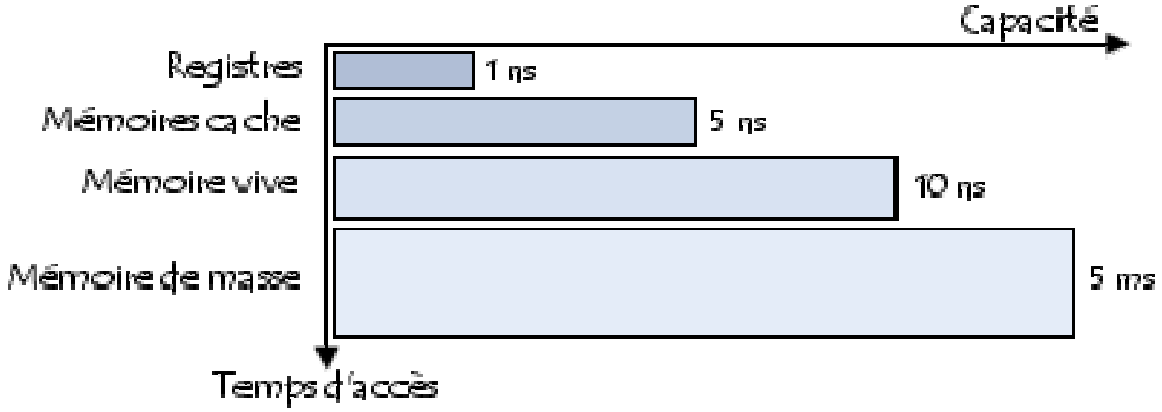
- Les mémoires rapides sont également les plus onéreuses. C'est la raison pour laquelle des mémoires utilisant différentes technologiques sont utilisées dans un ordinateur, interfacées les unes avec les autres et organisées de façon hiérarchique :

  Quand on s'éloigne du processeur vers les mémoires de masse, le temps d'accès et la capacités des mémoires augmentent, et le coût par bit diminue.

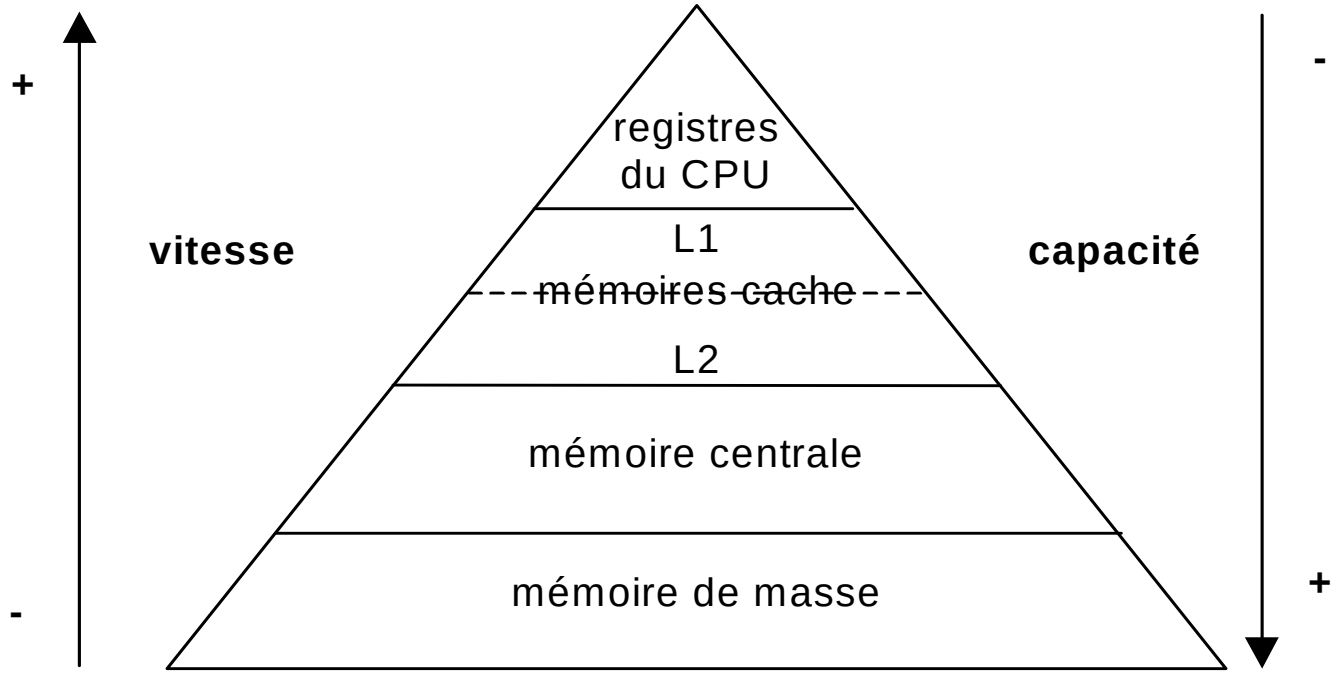
- Il faut bien distinguer le rôle des différentes mémoires qui existent dans la machine.

Généralités et définitions

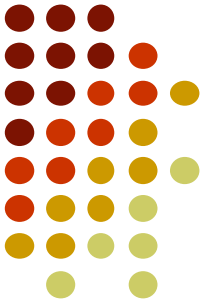
- Au sein de l'unité centrale, on trouve les **registres** caractérisés par une grande vitesse et servant principalement au stockage des opérandes et des résultats intermédiaires
- la **mémoire cache** ou l'**antémémoire** : mémoire rapide de faible capacité (par rapport à la mémoire centrale) utilisée comme mémoire intermédiaire entre le processeur et la mémoire centrale
- la **mémoire centrale** : pouvant atteindre de grandes capacités mais d'un temps d'accès plus grand que les deux premières, elle contient les données et les instructions des programmes.
- la **mémoire de masse** ou **mémoire auxiliaire** : mémoire périphérique de grande capacité et de coût relativement faible utilisée pour le stockage permanent des informations (sur support magnétique, magnéto-optique, ou optiques).



Hiérarchies de mémoire

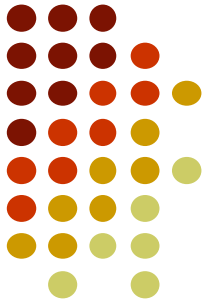


Les types d'accès aux mémoires



- Accès **séquentiel** : c'est l'accès le plus lent ; pour accéder à une information particulière on est obligé de parcourir toutes celles qui la précèdent (exemple les bandes magnétiques)
- Accès **direct** : les informations ont une adresse propre, et sont donc directement accessibles (par exemple la mémoire centrale, les registres)
- Accès **semi-séquentiel** : c'est une combinaison des accès direct et séquentiel (dans un disque magnétique, l'accès au cylindre est direct et l'accès au secteur est séquentiel)

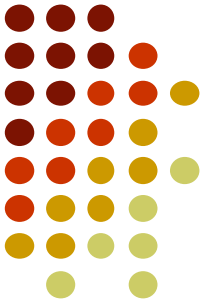
Mémoires secondaires



- Une mémoire secondaire n'est pas directement adressable par le processeur, ce qui signifie que son contenu doit être préalablement transféré en mémoire centrale pour pouvoir être manipulé.
  - **Les bandes magnétiques** ont été les premiers dispositifs de mémoire secondaire*. En faisant varier le courant dans une tête d'enregistrement, l'ordinateur écrit l'information sur la bande sous forme de petits points magnétiques. L'accès est **séquentiel**.
  - **Les disques magnétiques :** la capacité, la taille et les performances du disque dur ont considérablement évolué depuis l'apparition au début des années 80 du premier ordinateur IBM XT à disque dur (un disque de 5,25 pouces, de 10 cm d'épaisseur et de 10 Mo de capacité). Aujourd'hui des disques durs de 1 To, sont courants sur les ordinateurs personnels. Le disque dur est l'un des rares éléments de l'ordinateur qui soit à la fois mécanique et électronique, avec les disques SSD la mécanique disparaît.

(*) en fait non : les premiers étaient les cartes et les rubans perforés…

Disque dur

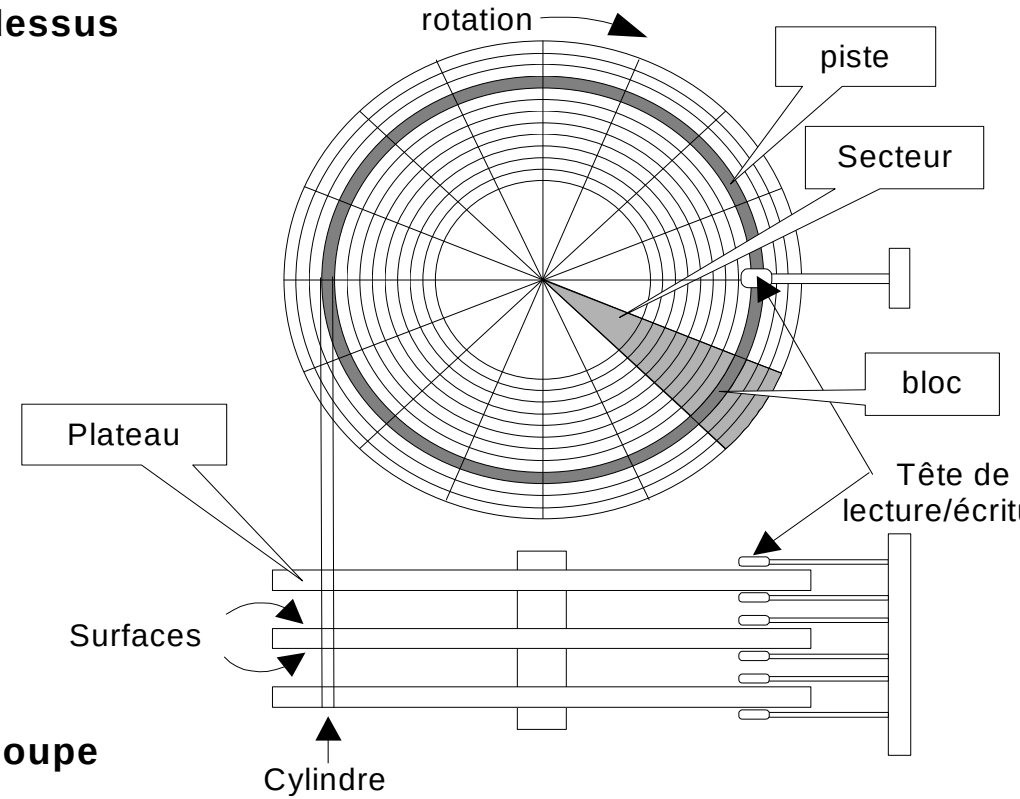


Un disque dur est composé :

- d'un ou plusieurs **plateaux** circulaires, dont l'une ou les deux surfaces sont recouvertes d'un fin matériau magnétique
- d’une **tête de lecture-écriture** par surface
- les surfaces sont généralement divisées en anneaux concentriques, les **pistes.**
- les pistes sont divisées en **secteurs** (couramment entre 32 et 256 mots)
- les pistes de même numéro de l’ensemble des plateaux forment un **cylindre.**

Disque dur

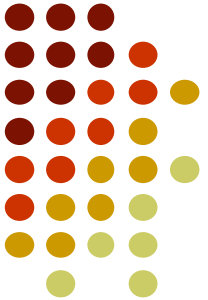
Vue de dessus



Vue en coupe

Le nombre de plateaux (ou têtes), de cylindres et de secteurs définissent la **géométrie** du disque

Disque dur

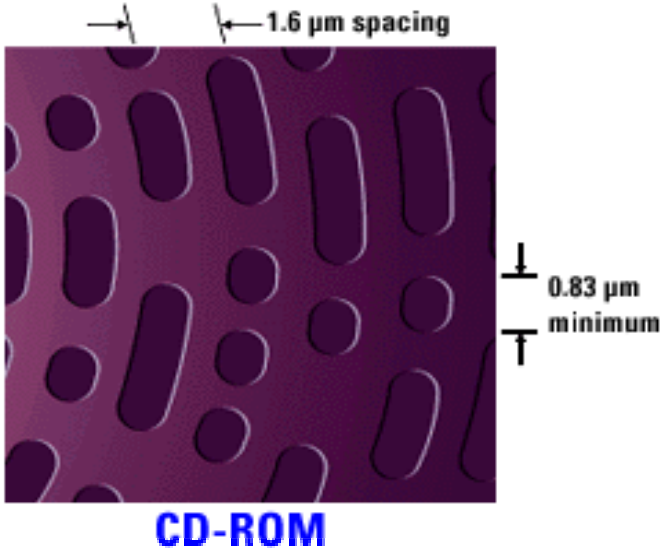


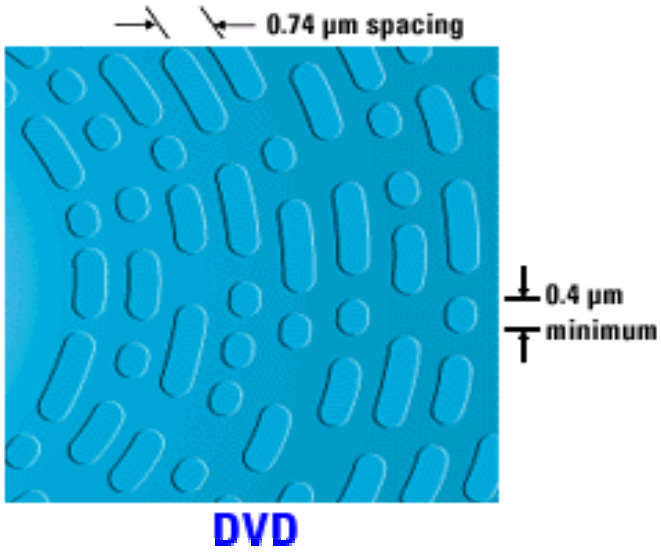
- Un **contrôleur** chargé de gérer le transfert de l'information entre la mémoire principale et le disque est associé à chaque unité de disque.
- Un **boîtier** scellé protège les éléments internes du disque dur contre la poussière qui risquerait de s'introduire entre les têtes de lecture/écriture et les plateaux et d'endommager irrémédiablement le disque en rayant son revêtement magnétique très fin (en général, la distance entre la tête et la surface n'est que de 2 millionième de pouce).

Les disques optiques

- En raison de leur énorme capacité de stockage, les disques optiques ont fait l'objet de nombreuses recherches entraînant leur évolution rapide. La première génération fut inventée par le groupe hollandais Philips, des développements ultérieurs furent établis en collaboration avec Sony.
- Ces disques, les **CDROM** [*Compact Disk Read Only Memory*], sont basés sur une technologie semblable à celle des compacts disques audionumériques : un détecteur reçoit et mesure l'énergie d'un rayon laser de faible puissance réfléchie sur la surface d'aluminium. Les "trous" dans le support et les "zones sans trous" entraînent des différences de réflectivité.
- Un CDROM contient environ 700 Mo.

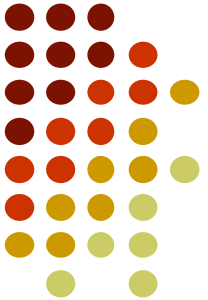
Les disques optiques





- Les disques inscriptibles constituent la génération suivante de disques optiques : d'abord les disques **CD-R** [*CD Recordable*] ou **CD-WORM** [*Write Once Read Many*] qui autorisent une seule opération d'écriture et de multiples opérations de lecture, puis les **CD-RW** [*CD-ReWritable*] autorisant la réécriture multiple (environ 1000 fois).
- Enfin le **DVD** [*Digital Versatile Disk*] s'impose comme le standard actuel. Son utilisation est multiple ainsi que ses appellations : DVD-Vidéo, DVD-ROM, DVD-Audio, DVD-R (enregistrable une fois) et DVD-RAM ou DVD-RW (réinscriptible à volonté). Extérieurement, il a la même taille qu'un CD (12 cm) mais peut contenir jusqu'à 17 GB.

Mémoires

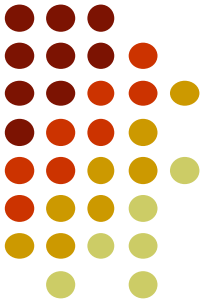


- En résumé, il convient de ne pas confondre la taille de la mémoire centrale qui définit les possibilités logicielles de la machine et la taille des disques qui indique sa capacité à stocker des informations permanentes.

2 - Organisation de la mémoire à l'exécution d'un programme

- Un programme en cours d'exécution en mémoire centrale est composé de 4 parties ou « segments »
  - Code : contient les instructions du programme
  - Données : contient les informations (variables) manipulées par le programme
  - Pile : permet l'évaluation du programme à l'exécution. Y sont représentées les données crées dynamiquement à l'appel d'une procédure
  - Tas : utilisé pour représenter les autres données dynamiques dont la durée de vie n'est pas liée à l'exécution des procédures

Schéma d'exécution d'un programme



Données
(statiques)

Code
(instructions)

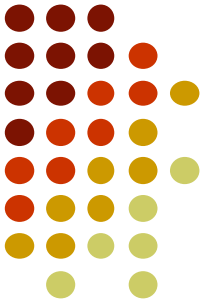
Pile :
contextes des
procédures en
cours
d'exécution

Tas :
Autres données
dynamiques
- chaînes
- listes
- arbres

Accès aux mots d'un segment

- L'adresse d'un mot est obtenue à partir
  - de l'adresse du début du segment (la « base »)
  - de la position du mot dans le segment (le « déplacement »)
- Pour accéder à un mot le système effectue un calcul pour obtenir l'adresse effective en mémoire de chaque mot :
  AE = base + déplacement
- Cette technique permet de disposer de code « relogeable », c'est-à-dire déplaçable en mémoire
- Pour déplacer un segment, il suffit de modifier sa base pour pouvoir l'exploiter après son déplacement

Gestion de la pile

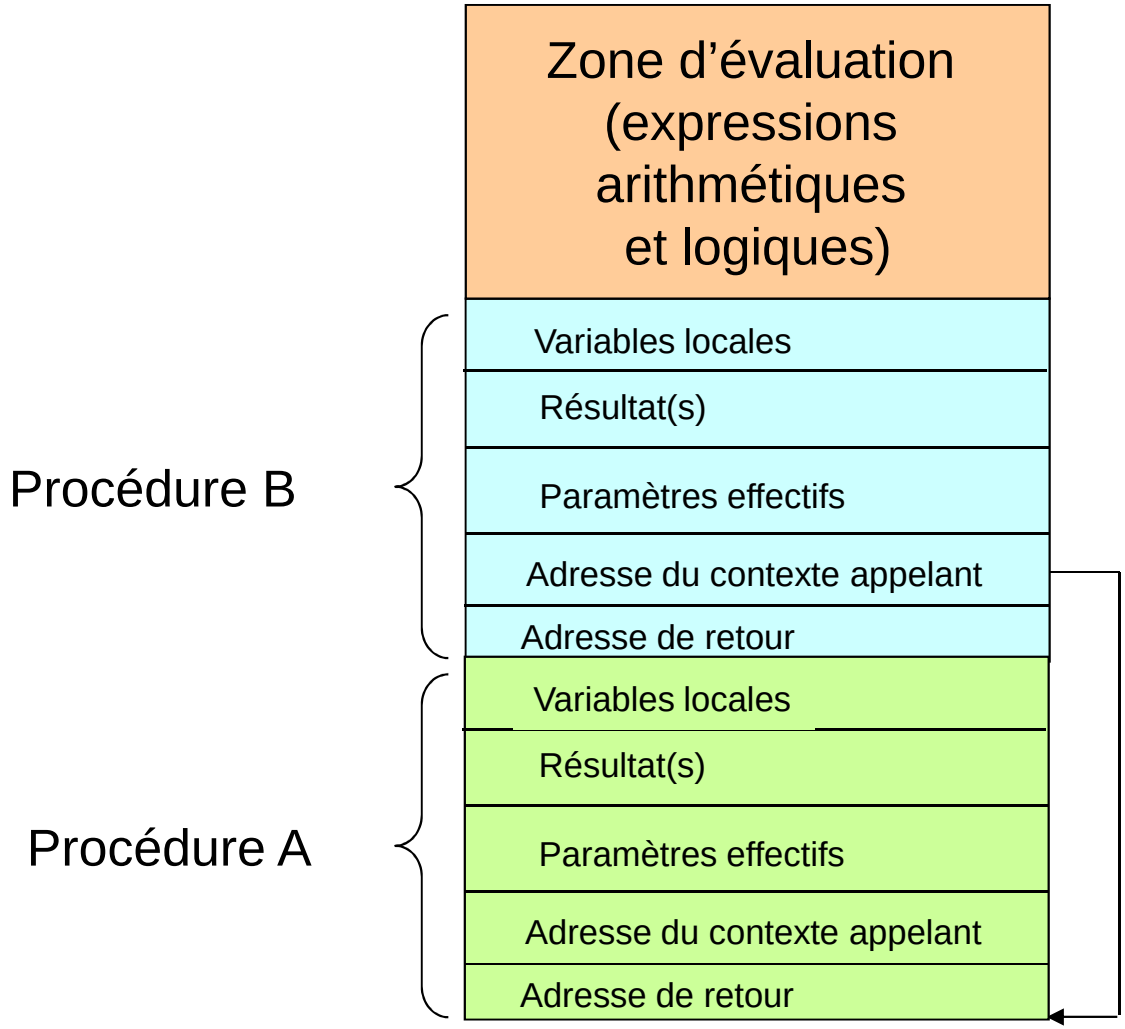


- La pile est utilisée pour empiler les contextes des procédures en cours d'exécution.
- Un contexte est structuré de la manière suivante:

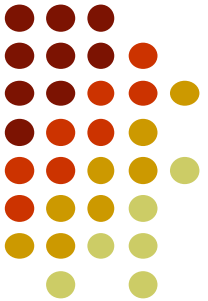
Variables locales
Résultat(s)
Paramètres effectifs
Adresse du contexte appelant
Adresse de retour à l'appelant

Gestion de la pile

- La durée de vie des informations du contexte correspond à l'exécution de la procédure qui lui correspond
- Intérêt de l'adressage basé : toutes les informations sont désignées par rapport à l'adresse de début de contexte

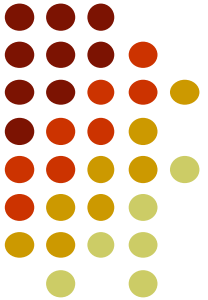


Gestion du tas



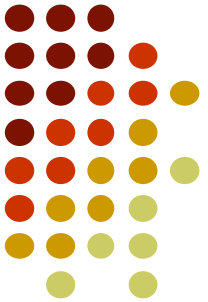
- L'organisation du tas est beaucoup plus anarchique :
  - Le tas est une réserve d'emplacements mémoire mise à la disposition du programme pour répondre aux demandes d'allocation dynamique (new) ou de libération de la mémoire (delete)
  - Le tas est composé de segments de mémoire qui sont soit occupés, soit libres.
  - Les segments libres sont chaînés entre eux

Gestion du tas



- A chaque demande de mémoire, le gestionnaire du tas choisit le segment libre dans lequel il va prendre la mémoire à donner au programme (différentes stratégies de choix du segment existent)
- Dans certaines situations plus aucun segment n'est suffisamment grand pour répondre à la demande
- Il faut alors mettre en œuvre un algorithme « ramasse-miettes » (garbage collector) pour restructurer le tas. Ceci va nécessairement engendrer le déplacement de données en mémoire, ce qui va nécessiter de mettre à jour les pointeurs sur ces objets

3 - Mémoire cache

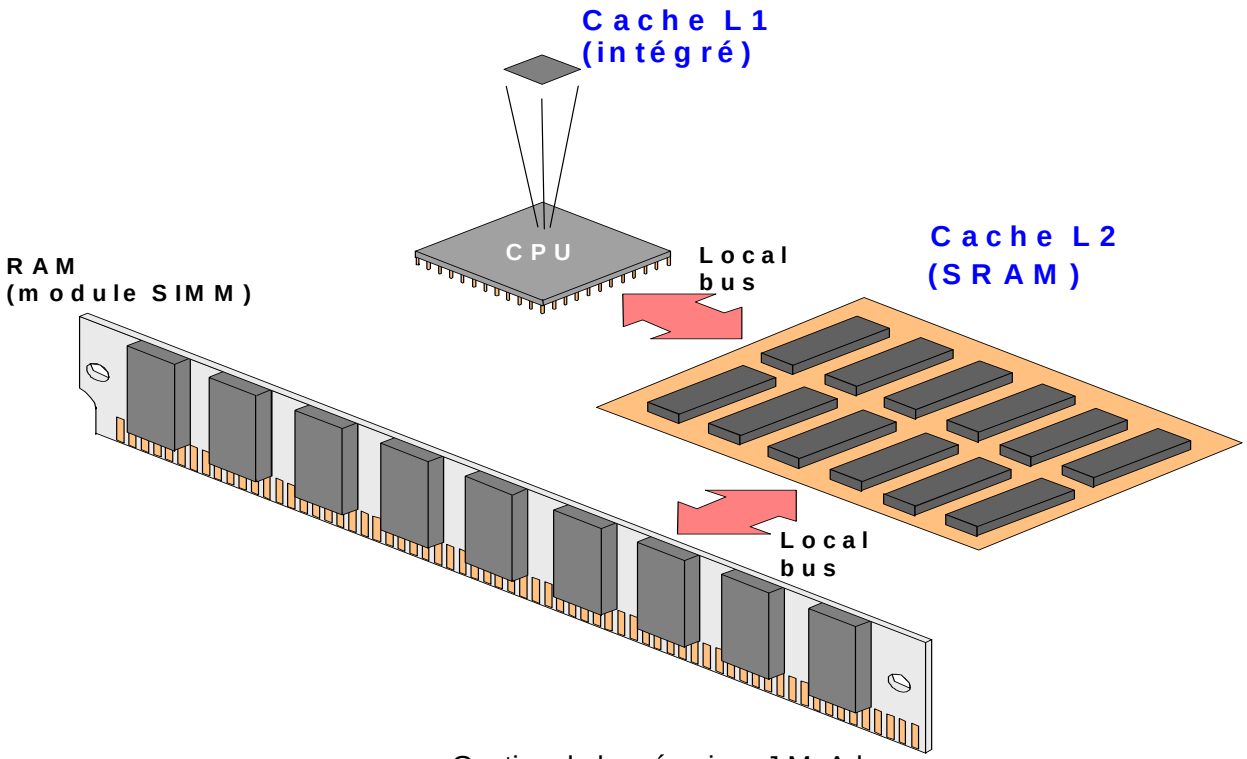


Notion de cache

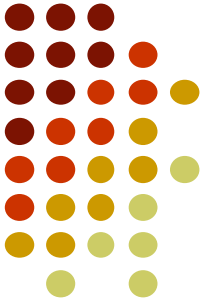
- Un cache est une *mémoire intermédiaire* dans laquelle se trouvent stockées toutes les informations qu'un élément demandeur est le plus susceptible de demander.
- Un cache sert donc à accélérer la communication entre un élément fournisseur (disque dur par exemple) plus lent que l'élément demandeur (processeur par exemple). Comme ces informations sont immédiatement disponibles, le temps de traitement se trouve diminué d'autant, ce qui mécaniquement accroît notablement les performances de l'ordinateur.

Mémoire cache

- Il existe souvent plusieurs niveaux de mémoire cache : une interne au processeur, une autre intégrée sur la carte mère, mais on peut en avoir aussi sur le disque dur.

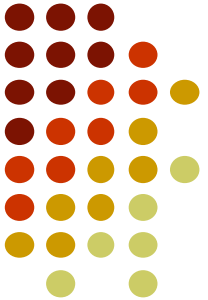


Mémoire cache



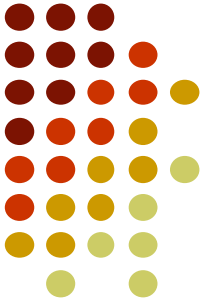
- **Mémoire cache** est la traduction littérale de l'expression anglaise **cache memory**, qui vient elle même de mémoire cachée, principe inventé à Grenoble dans les années 1960, l'académie française propose antémémoire.
- La différence entre **mémoire cache** et mémoire tampon (buffer) réside dans le fait que la mémoire cache duplique l'information, tandis que le tampon exprime l'idée d'une salle d'attente, sans impliquer nécessairement une duplication.
- Le ***cache buffer*** (tampon de cache) du disque dur ou *disk cache* (cache de disque) est à la fois un tampon où transite l'information et une mémoire cache qui recopie sous forme électronique les données stockées dans le disque sous forme magnétique.

Fonctionnement du cache



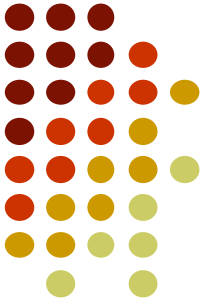
- Le cache contient une **copie des données originelles** lorsqu'elles sont coûteuses (en terme de temps d'accès) à récupérer ou à calculer par rapport au temps d'accès au cache. Une fois les données stockées dans le cache, l'utilisation future de ces données peut être réalisée en accédant à la copie en cache plutôt qu'en récupérant ou recalculant les données, ce qui abaisse le temps d'accès moyen.
- Le processus fonctionne ainsi :
  - L'élément demandeur (microprocesseur) demande une information
  - Le cache vérifie s'il possède cette information. S'il la possède, il la retransmet à l'élément demandeur; on parle alors de ***succès de cache***. S'il ne la possède pas il la demande à l'élément fournisseur (mémoire principale); on parle alors de ***défaut de cache***

Fonctionnement du cache



- L'élément fournisseur traite la demande et renvoie la réponse au cache ;
- Le cache la stocke pour utilisation ultérieure et la retransmet à l'élément demandeur.
- Si les mémoires cache permettent d'accroître les performances, c'est en partie grâce à deux principes qui ont été découverts suite à des études sur le comportement des programmes informatiques :
  - Le principe de localité spatiale : qui indique que l'accès à une instruction située à une adresse X va probablement être suivi d'un accès à une zone tout proche de X.
  - Le principe de localité temporelle : qui indique que l'accès à une zone mémoire à un instant donné a de fortes chances de se reproduire dans la suite du programme.

Les caches

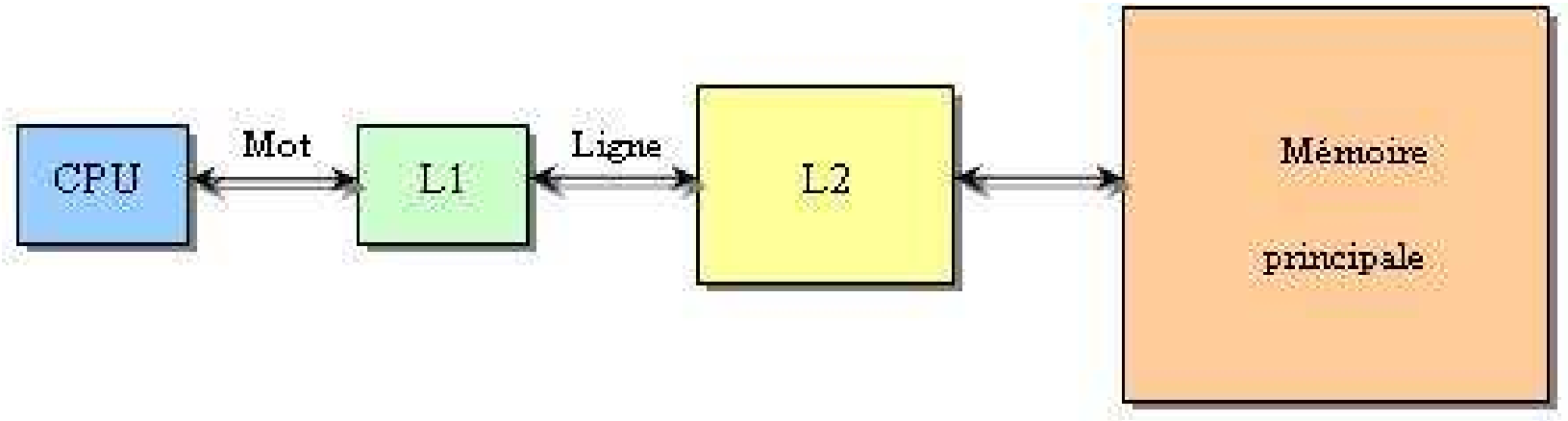


On trouve une zone de cache :

- Cache de premier niveau (L1) dans les processeurs (cache de données souvent séparé du cache d'instructions)
- Cache de second niveau (L2) dans certains processeurs (peut se situer hors de la puce) ;
- Cache de troisième niveau (L3) rarement (présent sur les Intel Core i7)
- Dans les disques durs
- Dans les serveurs proxy

Mémoire cache des microprocesseurs

- Elle est souvent subdivisée en niveaux qui peuvent aller jusqu'à trois. Elle est très rapide, et donc très chère. Il s'agit souvent de SRAM (Static Random Access memory).

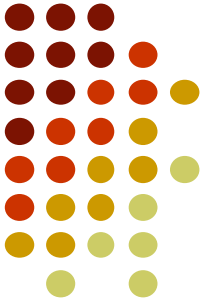


Différents niveaux de mémoire d'un microprocesseur

Mémoire cache des microprocesseurs

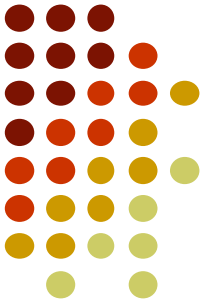
- En programmation, la taille de la mémoire cache revêt un attrait tout particulier, car pour profiter de l'accélération fournie par cette mémoire très rapide, *il faut que les parties de programme tiennent le plus possible dans cette mémoire cache*. Comme elle varie suivant les processeurs, ce rôle d'optimisation est souvent *dédié au compilateur*. De ce fait, plus la taille de la mémoire cache est grande, plus la taille des programmes accélérés peut être élevée.
- C'est aussi un *élément souvent utilisé par les constructeurs pour faire varier les performances d'un produit* sans changer d'autres matériels. Par exemple, pour les microprocesseurs, on trouve des séries bridées (avec une taille de mémoire cache volontairement réduite), tels que les Duron chez AMD ou Celeron chez Intel, et des séries haut de gamme avec une grande mémoire cache comme les processeurs *Opteron* chez AMD, ou *Pentium EE* ou Core i7 chez Intel.

Définitions



- Une *ligne* est le plus petit élément de données qui peut être transféré entre la mémoire cache et la mémoire de niveau supérieur.
- Un *mot* est le plus petit élément de données qui peut être transféré entre le processeur et la mémoire cache.

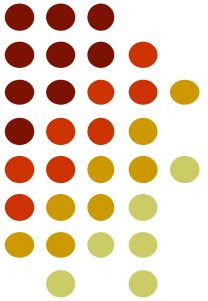
Défauts de cache



Il existe trois types de défauts de cache en système monoprocesseur et quatre dans les environnements multiprocesseurs :

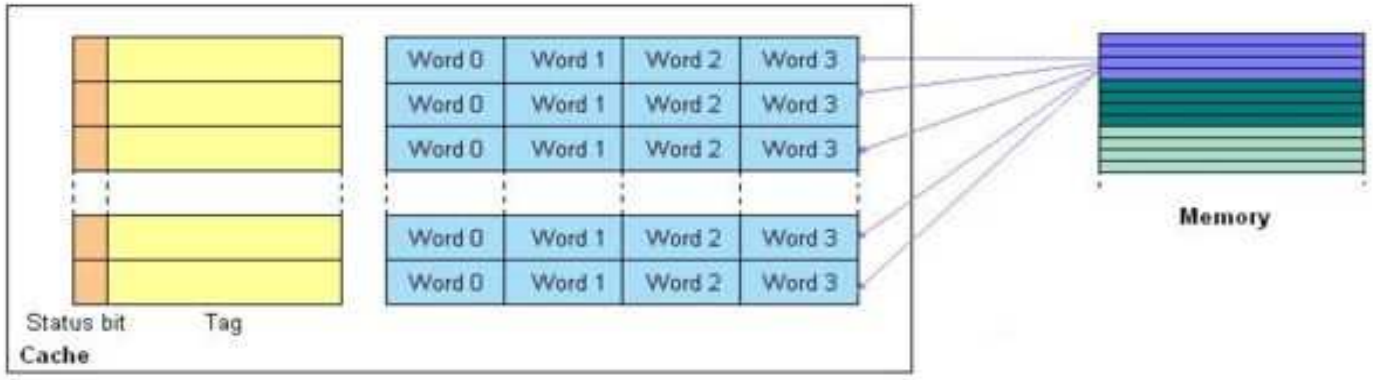
- les **défauts de cache obligatoires** : ils correspondent à la première demande du processeur pour une donnée/instruction spécifique et ne peuvent être évités,
- les **défauts de cache capacitifs** : l'ensemble des données nécessaires au programme excèdent la taille du cache, qui ne peut donc pas contenir toutes les données nécessaires,
- les **défauts de cache conflictuels** : deux adresses distinctes de la mémoire de niveau supérieur sont enregistrés au même endroit dans le cache et s'évincent mutuellement, créant ainsi des défauts de cache,
- les **défauts de cache de cohérence** : ils sont dus à l'invalidation de lignes de la mémoire cache afin de conserver la cohérence entre les différents caches des processeurs d'un système multi-processeurs.

Le mapping

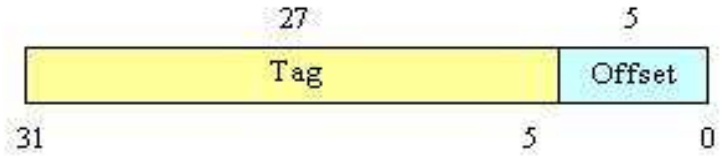


- La mémoire cache ne pouvant contenir toute la mémoire principale, il faut définir une méthode indiquant à quelle adresse de la mémoire cache doit être écrite une ligne de la mémoire principale. Cette méthode s'appelle le mapping.
- Il existe trois types de mapping :
  - La mémoire cache complètement associative (*fully associative cache*),
  - La mémoire cache directe (*direct mapped cache*),
  - La mémoire cache N-associative (*N-way set associative cache*).

Mémoire cache complètement associative

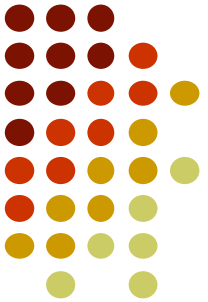


- Chaque ligne de la mémoire de niveau supérieur peut être écrite à n'importe quelle adresse de la mémoire cache.
- Cette méthode requiert beaucoup de logique car elle donne accès à de nombreuses possibilités. Ceci explique pourquoi l'associativité complète n'est utilisée que dans les mémoires cache de petite taille. Cela donne le format suivant de l'adresse :



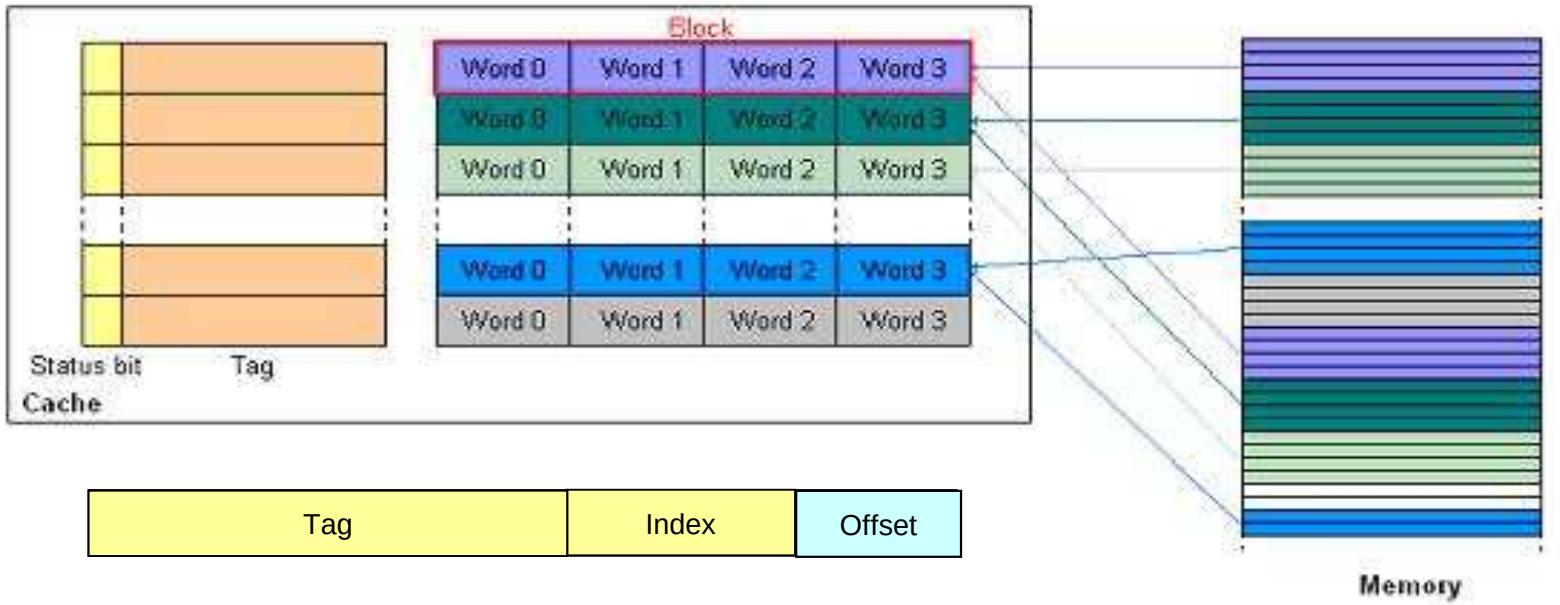
Tag = n° de la ligne mémoire enregistrée Offset = n° du mot dans la ligne

Mémoire cache directe



- Chaque ligne de la mémoire principale ne peut être enregistrée qu'à une seule adresse de la mémoire cache.
- Ceci crée de nombreux défauts de cache conflictuels si le programme accède à des données qui sont mappées sur les mêmes adresses de la mémoire cache.
- La sélection de la ligne où la donnée sera enregistrée est habituellement obtenue par: *Ligne* = *Adresse* mod *Nombre de lignes*.

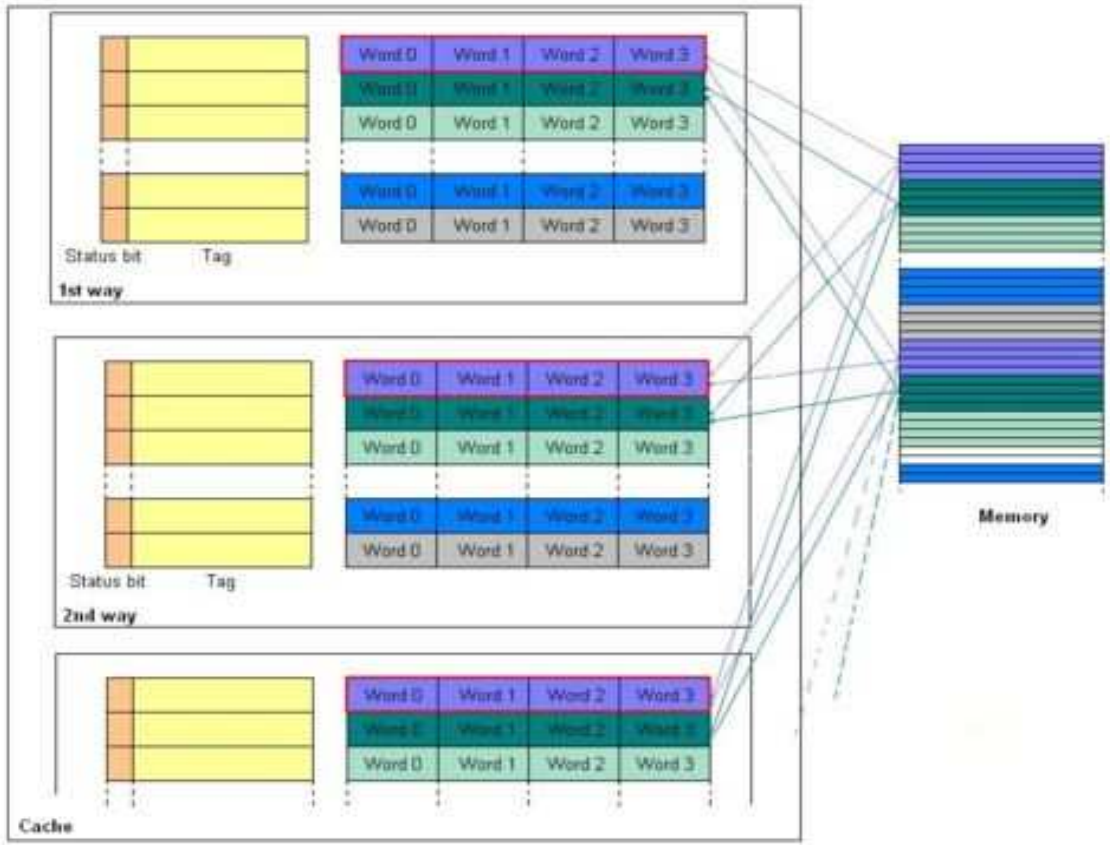
Mémoire cache directe



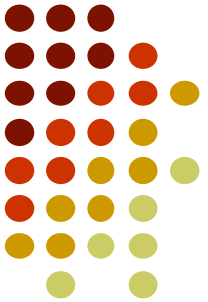
Une ligne de cache est partagée par de nombreuses adresses de la mémoire de niveau supérieur. Il nous faut donc un moyen de savoir quelle donnée est actuellement dans le cache. Cette information est donnée par le *tag*, qui est une information stockée dans le cache. L'index correspond à la ligne où est enregistrée la donnée. En outre, le contrôleur de la mémoire cache doit savoir si une ligne contient une donnée ou non. Un bit additionnel (appelé bit de validité) indique si la ligne est libre ou non.

Mémoire cache N-associative

- Il s'agit d'un compromis entre le mapping direct et complètement associatif essayant d'allier la simplicité de l'un et l'efficacité de l'autre. La mémoire cache est divisée en *ensembles* (*sets*) de N lignes de cache.
- Une ligne de la mémoire de niveau supérieur est affectée à un ensemble, elle peut par conséquent être écrite dans n'importe laquelle des voies.
- Ceci permet d'éviter de nombreux défauts de cache conflictuels.
- À l'intérieur d'un ensemble, le mapping est complètement associatif. En général, la sélection de l'ensemble est effectuée par: *Ensemble = Adresse mémoire* mod (*Nombre d'ensembles*).



Caches unifiés ou caches séparés



- Pour fonctionner, un processeur a besoin de données et d'instructions. Il existe donc deux solutions pour l'implémentation des mémoires cache:
  - le cache unifié : données et instructions sont enregistrées dans la même mémoire cache,
  - les caches séparés de données et d'instructions.
- Séparer données et instructions permet notamment d'augmenter la fréquence de fonctionnement du processeur, qui peut ainsi accéder simultanément à une donnée et une instruction.

Politique d'écriture dans la mémoire de niveau supérieur

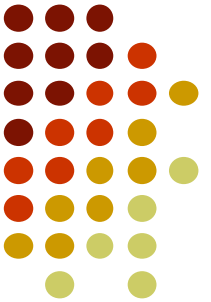
- Quand une donnée/instruction se situe dans le cache, le système en possède deux copies: une dans la mémoire de niveau supérieur et une dans la mémoire cache. Deux différentes politiques s'affrontent:
  - *write through*: la donnée/instruction est écrite à la fois dans le cache et dans la mémoire de niveau supérieur. La valeur de la mémoire principale est constamment cohérente entre le cache et la mémoire de niveau supérieur, simplifiant ainsi de nombreux protocoles de cohérence,
  - *write back*: l'information n'est écrite dans la mémoire de niveau supérieur que lorsque la ligne disparaît du cache (invalidée par d'autres processeurs, évincée pour écrire une autre ligne...). Cette technique est la plus répandue car elle permet d'éviter de nombreuses écritures mémoires inutiles.

    Cependant, afin de ne pas écrire des informations qui n'ont pas été modifiées chaque ligne de la mémoire cache est pourvue d'un bit indiquant si elle a été modifiée.

Algorithmes de remplacement des lignes de cache

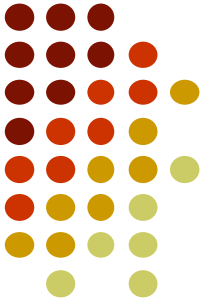
- Les caches associatifs de N voies et complètement associatifs impliquent le mapping de différentes lignes de la mémoire de niveau supérieur sur le même ensemble.
- Ainsi, il faut désigner la ligne qui sera effacée au profit de la ligne nouvellement écrite. Le but de l'algorithme de remplacement des lignes de cache est de choisir cette ligne de manière optimale.

Algorithmes de remplacement des lignes de cache



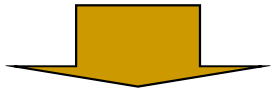
- Les algorithmes de remplacement des lignes de cache les plus répandus sont :
  - aléatoire pour sa simplicité de création de l'algorithme,
  - FIFO(First In First Out) pour sa simplicité de conception,
  - LRU (Least Recently Used) qui mémorise la liste des derniers éléments accédés
  - FINUFO - First In Not Used, First Out (algorithme de l'horloge ou Clock) = approximation du LRU

4 - Mémoire Virtuelle



Définition

- Mémoire **virtuelle** = **support** de l'ensemble des informations **potentiellement** accessibles.

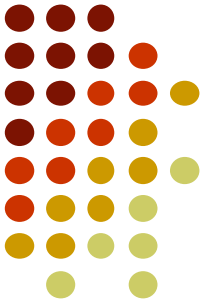


Ensemble des emplacements dont l'adresse **peut être** engendrée par le processeur.

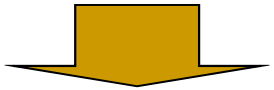
≠

- Mémoire **physique** = Ensemble des emplacements RAM physiquement présents dans l'ordinateur.

Pourquoi une mémoire virtuelle ?

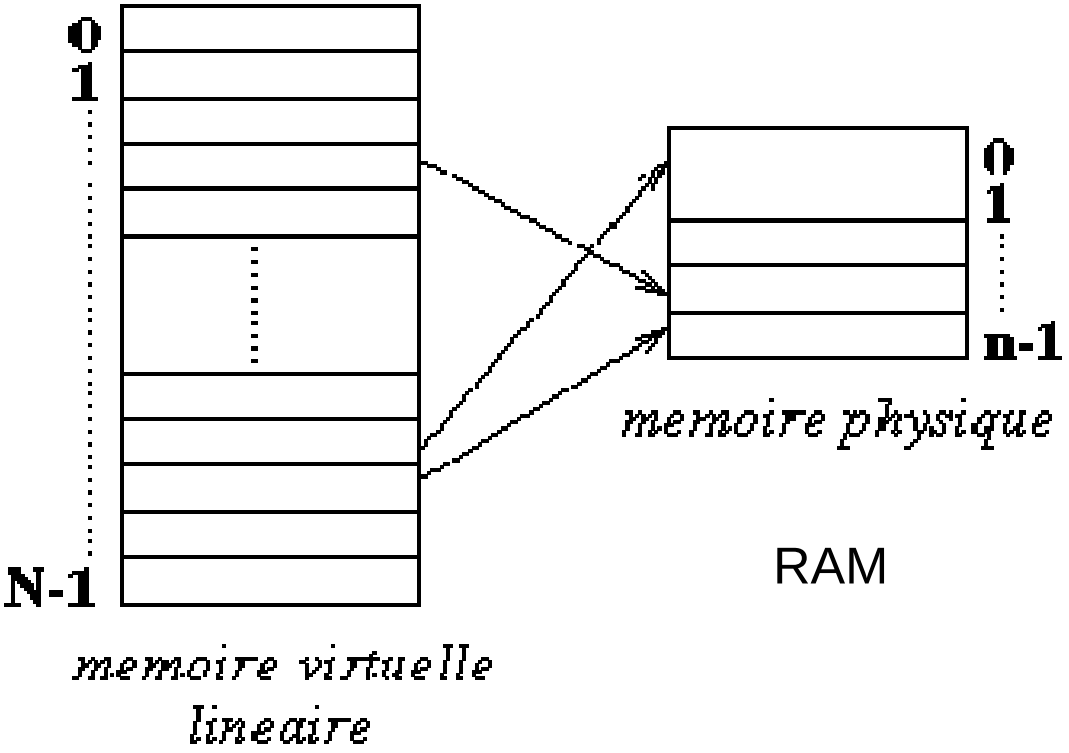


- Mémoire ***physique*** coûteuse.
- Mémoire ***secondaire*** (disques, mémoire étendue, ...) peu coûteuse.
- Programmes gourmands en mémoire et qui ne "tiennent pas" toujours en RAM.



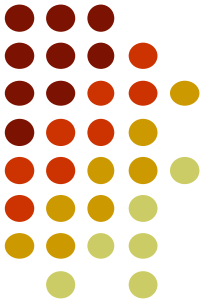
- Idée : utiliser la mémoire secondaire "comme" mémoire RAM.

Mémoire virtuelle



Mémoire secondaire

Idée générale



- Il s'agit de conserver en mémoire une « partie » des programmes en cours d'exécution.
- Si un programme ***A*** veut s'exécuter alors qu'il n'y a plus de place en mémoire, un « morceau » d'un autre programme est « déplacé » en mémoire secondaire et remplacé par un « morceau » de *A*.
- Donc, un programme est découpé en morceaux que l'on nomme **pages**, de taille fixe.
- La mémoire physique est elle aussi découpée en pages, de même taille, ainsi que la partie de la mémoire secondaire servant de mémoire virtuelle.

Les problèmes de l'allocation mémoire

- Comment faire la correspondance entre adresses **virtuelles** et adresses **physiques** ?
- Comment gérer de la mémoire physique ?

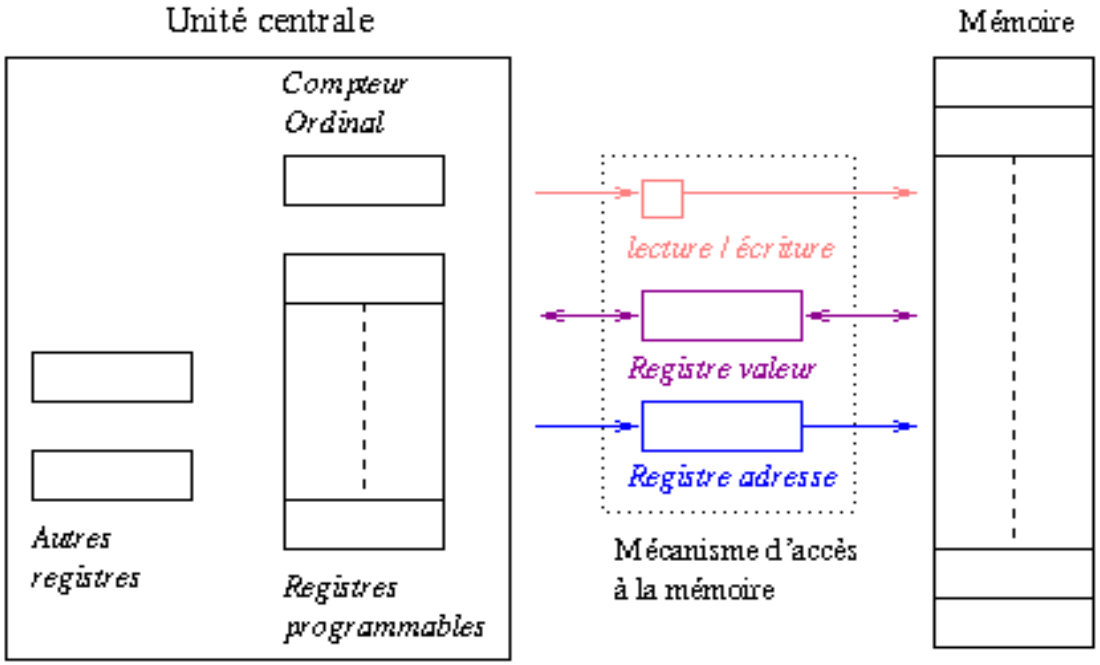
Dans un environnement multitâches, comment assurer :

- le partage de l'information ?
- La protection mutuelle ?

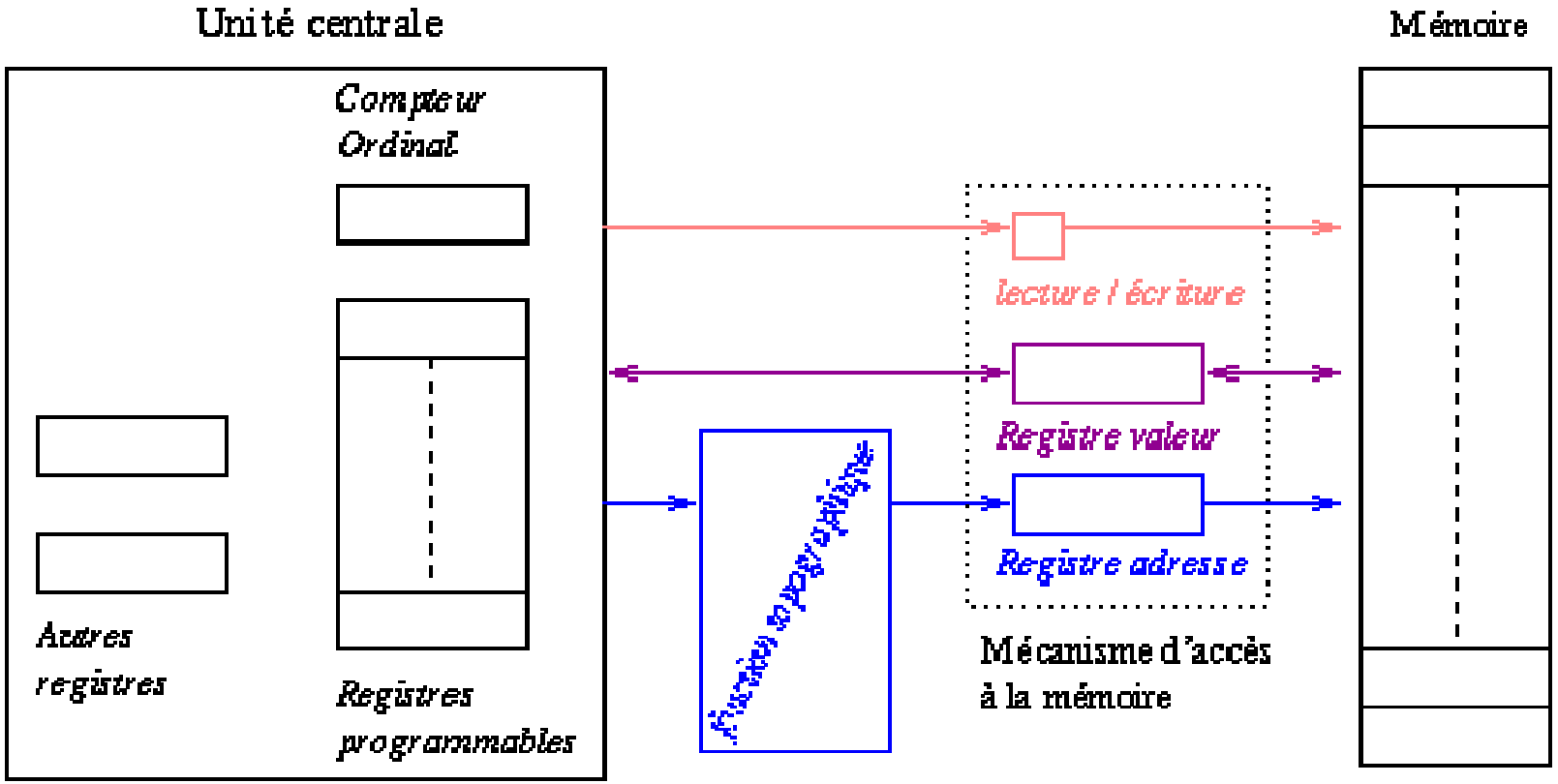
Correspondance adresses virtuelles - adresses physiques

On utilise une fonction topographique qui associe à une adresse virtuelle, une adresse réelle.

Rappel : l'architecture classique d'accès à la mémoire :



Architecture avec réimplantation dynamique



Fonction topographique (x : *adresse_virtuelle*) → *adresse_réelle*

Table des pages virtuelles

- La mémoire virtuelle, avec sa table des pages, est **une** implémentation possible de la fonction topographique.

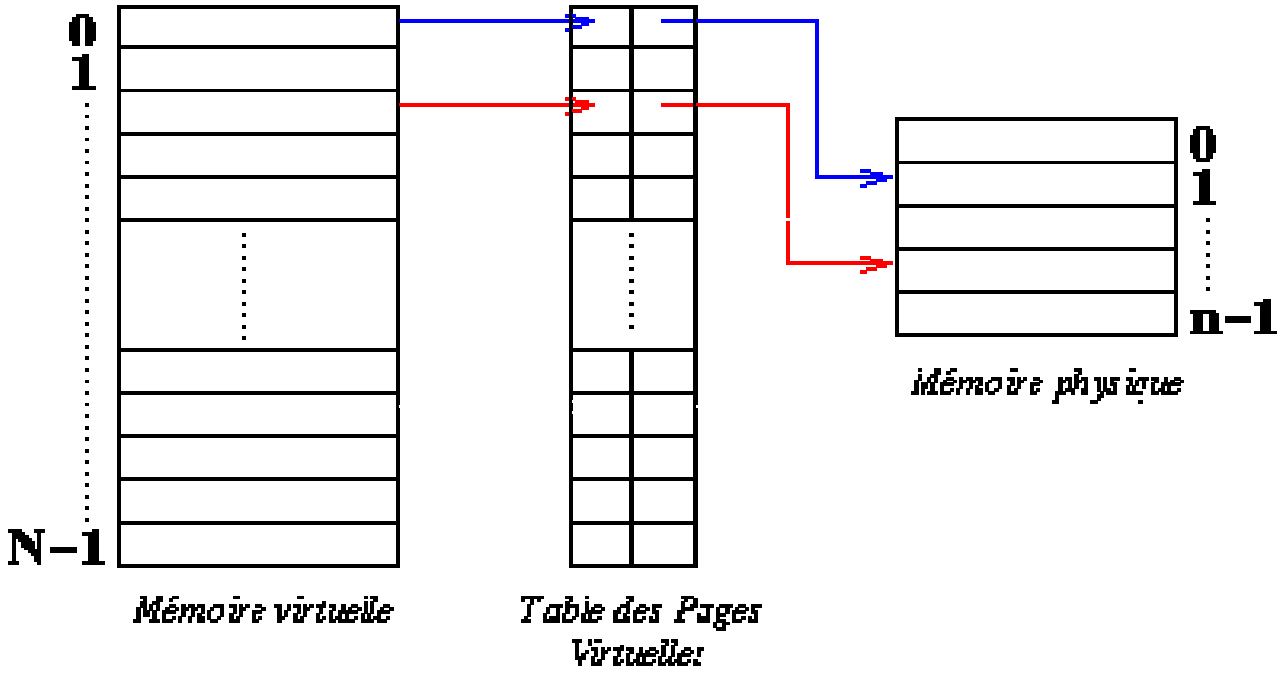
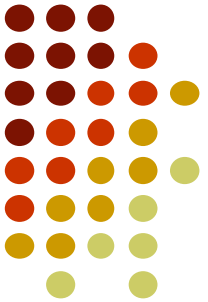


Table des pages virtuelles

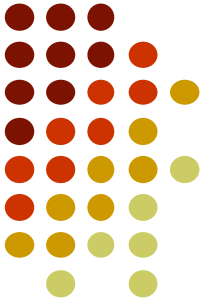


Présent	Modifié	Protection	Num page physique
... oui ...	... non ...	... rx ...	... 18 ...

- Présent : page virtuelle présente en mémoire physique ?
- Modifié : page modifiée ?
- Protection : droits d'accès.
- Num page physique : page physique correspondante.

- La table des pages virtuelles est une implémentation particulière d'une **fonction de pagination** (il en existe d'autres).

La mémoire virtuelle linéaire

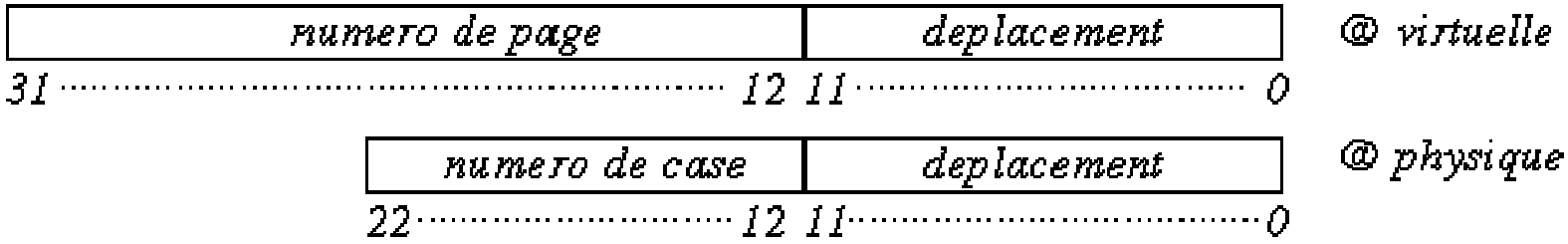


- L'adresse du 1er octet de la page *n*

= l'adresse du dernier octet de la page (*n-1*) + 1.

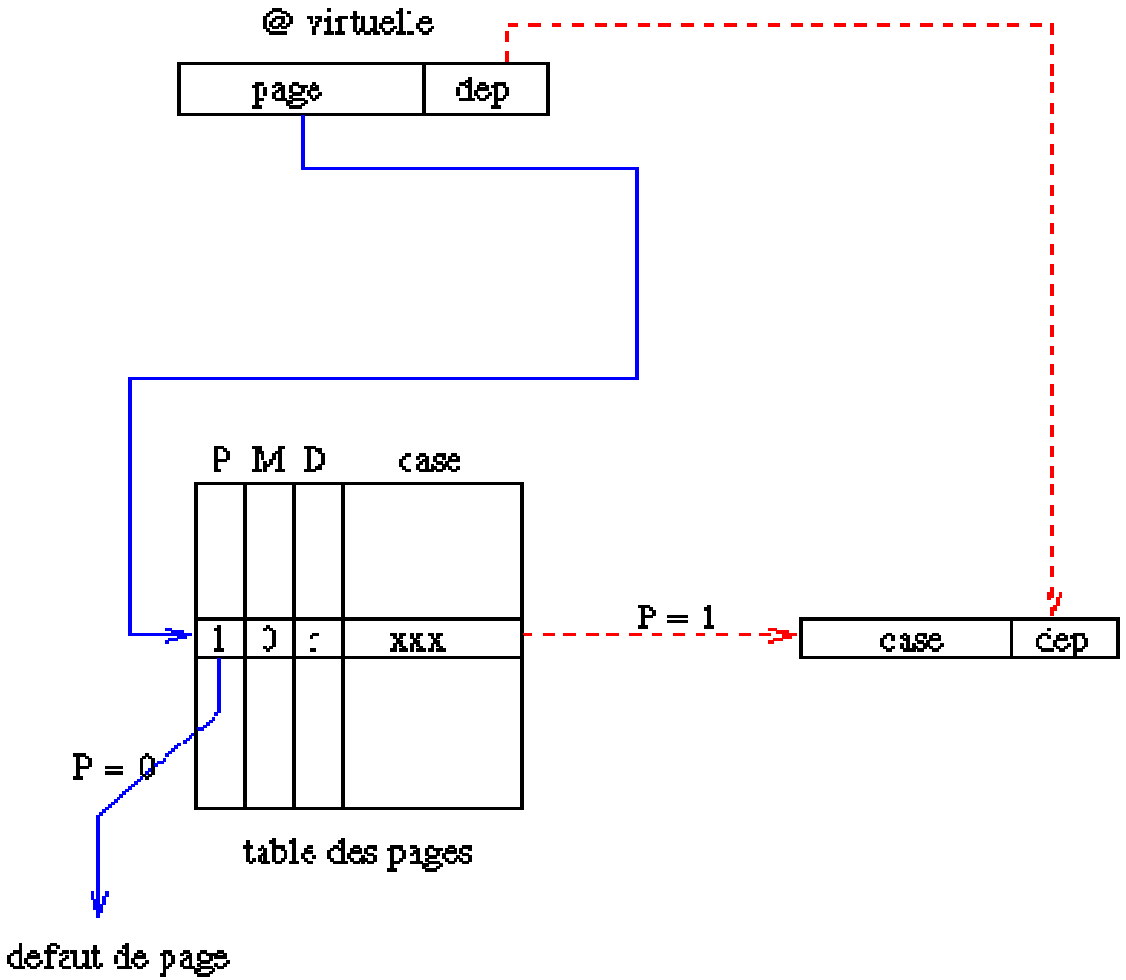
- Avantage : organisation identique à celle d'une mémoire physique.

Passage de l'Adresse Virtuelle à l'Adresse Physique



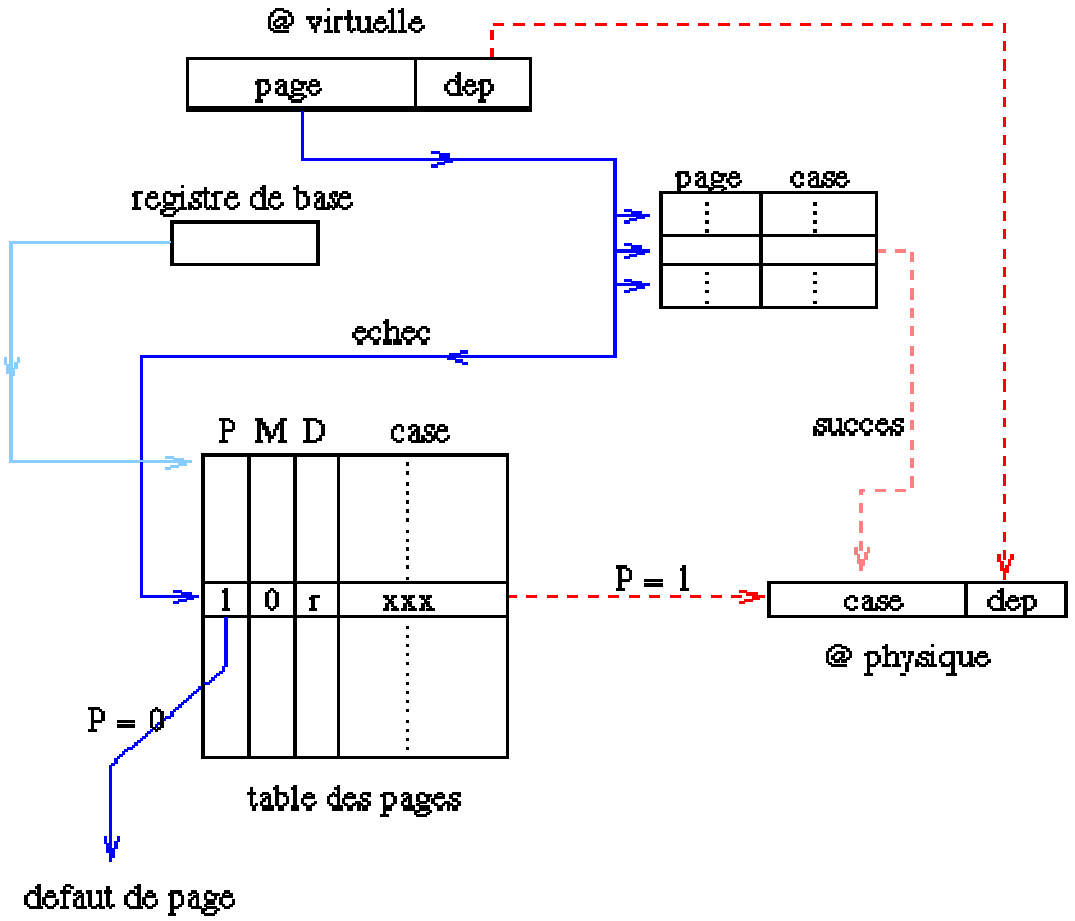
- Le calcul de l'adresse réelle à partir de l'adresse virtuelle se réalise ainsi :
  - le numéro de page virtuelle donne l'entrée de la TPV dans laquelle se trouve le numéro de page physique ;
  - le déplacement est le même (les pages physiques et virtuelles ont la même taille) ;
  - si la page virtuelle n'est pas présente en mémoire physique, alors il se produit un *défaut de page*.

Passage de l'Adresse Virtuelle à l'Adresse Physique

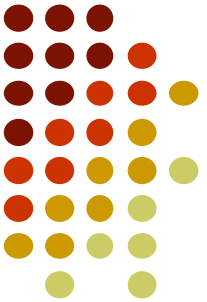


Passage de l'Adresse Virtuelle à l'Adresse Physique

- Pour accélérer le processus, on utilise des *mémoires associatives* qui recensent les dernières pages utilisées

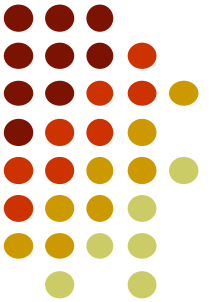


Le défaut de page



- L'adresse virtuelle référence une page qui n'est pas présente en mémoire physique. Le mécanisme d'adressage génère un **défaut de page**.
- Si la mémoire physique est pleine, il faut virer de la mémoire physique une page (**remplacement**) :
  - choisir une page "victime",
  - si elle a été modifiée, la réécrire sur disque,
  - modifier les indicateurs de présence en TPV
- Puis, dans tous les cas :
  - charger la page référencée en mémoire physique (**placement**) ;
  - modifier les indicateurs de présence en TPV.

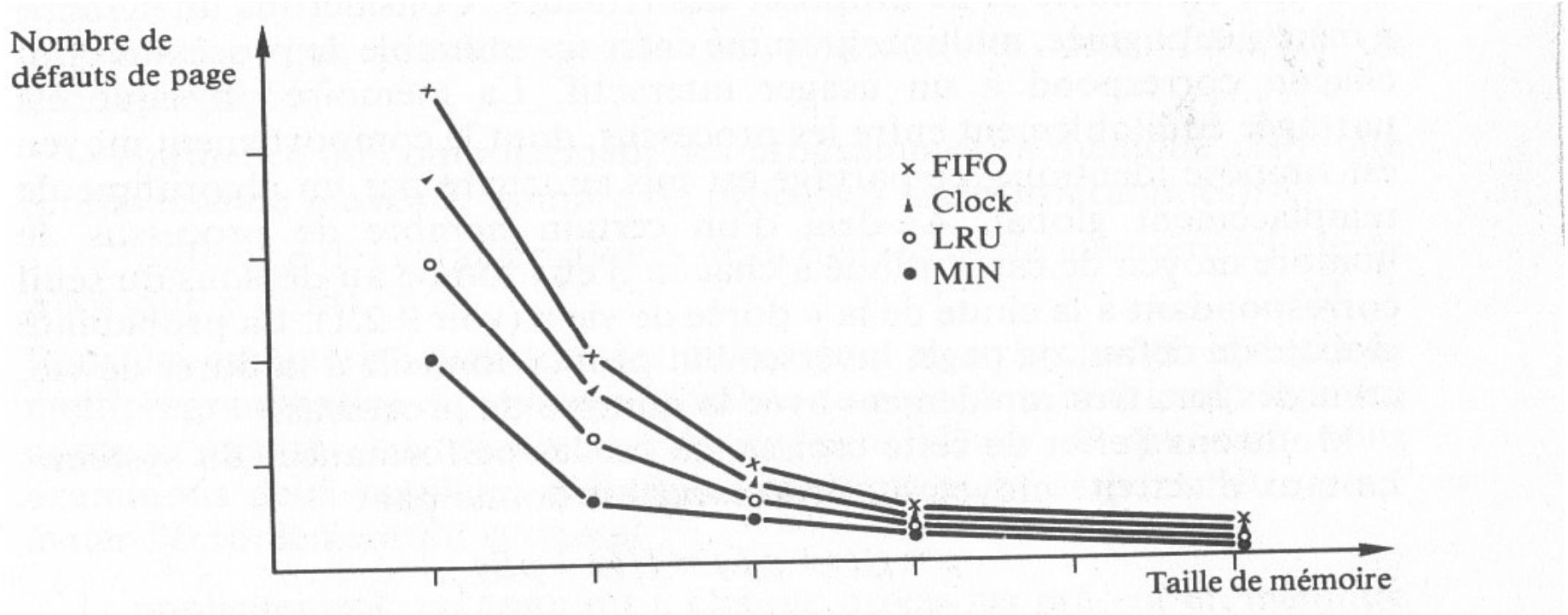
Le choix d'une victime remplacement



De nombreux algorithmes :

- **FIFO** - First In First Out : ordre chronologique de chargement
- **LRU** - Least Recently Used : ordre chronologique d'utilisation (ne convient pas : contraintes de temps trop fortes)
- **FINUFO** - First In Not Used, First Out (algorithme de l'horloge ou Clock) : un bit indique si la page a été accédée
- **LFU** - Least Frequently Used
- **Random** : au hasard
- **MIN** : algorithme optimal (correspond à une évaluation a posteriori).

Performance des algorithmes de remplacement

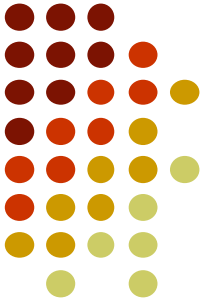


L'influence de la taille de mémoire est très largement supérieure à celle de l'algorithme de remplacement; autrement dit, on améliore davantage les performances d'un programme en augmentant le nombre de cases allouées plutôt qu'en raffinant l'algorithme de remplacement; et cela d'autant plus que les performances initiales sont mauvaises.

Avantages de la pagination

- Meilleure utilisation de la mémoire physique (programmes implantés par fragments, dans des pages *non-consécutives*).
- Possibilité de ne charger des pages que lorsqu'elles sont référencées (chargement à la demande).
- Indépendance de l'espace virtuel et de la mémoire physique (mémoire virtuelle généralement plus grande).
- Possibilité de ne vider sur disque que des pages modifiées.
- Possibilité de recouvrement dynamique (couplage).

Inconvénients de la pagination



- Fragmentation interne (toutes les pages ne sont pas remplies).
- Impossibilité de lier deux (ou plusieurs) procédures liées aux mêmes adresses dans l'espace virtuel.

Et voilà !