## Légende

SE : Système d’exploitation. E/S : Entré/Sortie. EX : Exception système. M : Mémoire. MC : Mémoire Centrale. Mcache : Mémoire cache. Av : Adresse virtuelles. AP : Adresse physique. MMU : Memory-Management Unit. CPU : Central Processing Unit. Rg : Registres. PC : Program Counter. Pg : Programme. Pu : Processus. PCB : Pu control block. ORD : Ordonnanceur. PID : Pu Id. IPC : Intermediary Pu Communications. MPC : Multiprocessors. TD : Thread. IB : Interblocage.

*AVs* : Avantages – *DAVs* : Désavantages – Mots surlignés : Données manquantes.

Gestion Mémoire

## Concept de base

Un Pg doit être chargé de la M vers un Pu. Le CPU a un accès direct à la MC, et aux Rg, mais c’est tout.

L’accès à la MC peut prendre plusieurs cycles, provoquant un stall – La Mcache se trouve entre la MC et aussi entre les Rg.

À considerer: Support matériel, performance, fragmentation, relocalisation, *swapping*, partage de données, protection.

Chaque Pu reçoit une partie de M pour opérer. Cette allocation est caractérisée par deux valeurs : l’adresse de début (base) et la limite de M allouée (limit) – Le Pu opère donc entre les adresses base (inclue) et base+limit. – Lorsque le CPU veut accéder à la M d’un Pu, l’adresse doit se trouver dans l’intervalle alloué à celui-ci, sinon on lance une EX.

## Adresses

Adresses logiques ou virtuelles (AV) : Utilisées par les Pu. Ces adresses ne sont pas les vraies adresses. – Adresses physiques (AP) Utilisées par le CPU.

Les AP sont traduites en AV et vice-versa par le Memory-Management Unit (MMU). Le MMU est utilisé par le CPU pour convertir les AV en AP de façon dynamique.

## Allocation

Problèmes reliés à l’allocation de M dynamique :

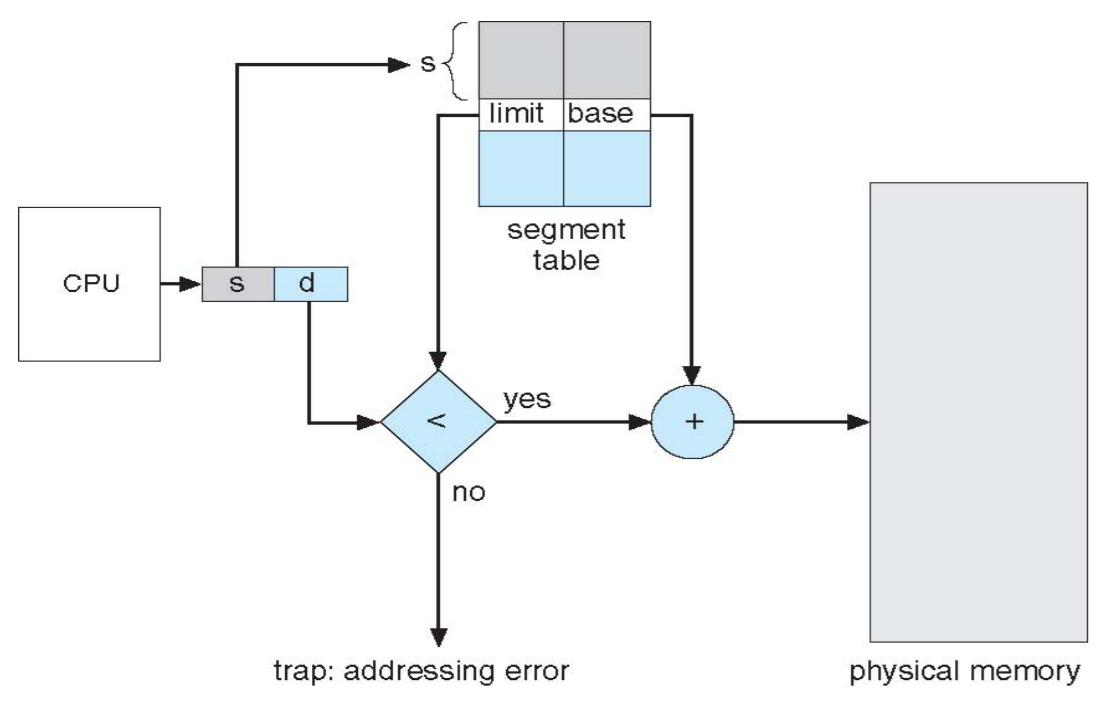
Choix d’allocation  
Utiliser le premier trou assez grand? (First-fit) – Utiliser le plus petit trou assez grand? (Best-fit) Utiliser le plus gros trou? (worst-fit).

Fragmentation  
Externe : Il y a assez de M libre, mais elle est fragmentée en trop de petits morceaux inutilisables. – Interne : M gaspiée par le SE (utilise moins que ce qu’il a alloué).

## Segmentation

Chaque espace logique (stack, méthodes objets, etc.) d’un Pg est segmenté en utilisant la technique base+limit.

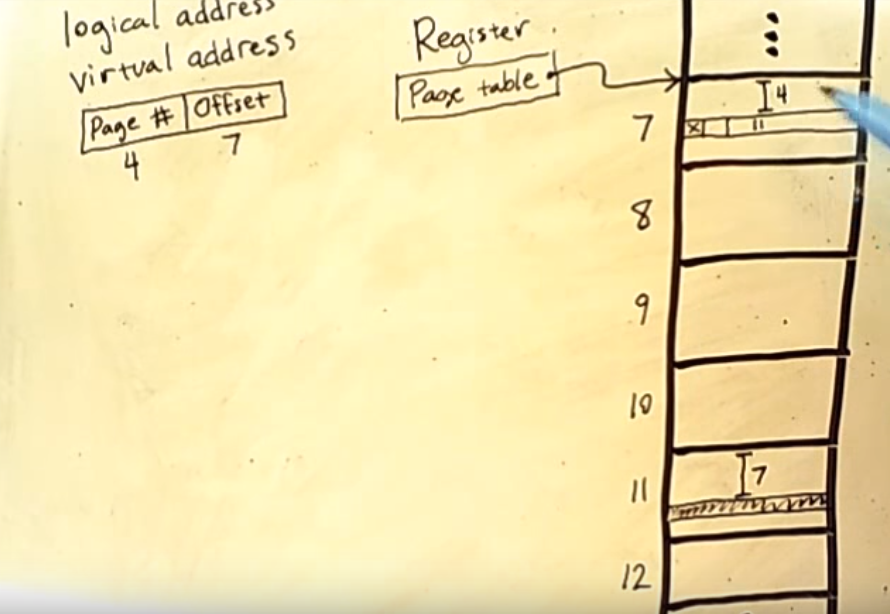
Chaque segment à une adresse qui contient : l’id du segment et le offset. Ainsi, l’adresse physique correspond à base[segId] + offset.



Problèmes? Jamais assez de segments – Adresses trop grandes, fragmentation externe.

## Pagination

[Youtube](https://www.youtube.com/watch?v=KqHNaOrxttM). Le système de pagination est plutôt simple. Pourquoi utiliser un système comme celui-ci? Car la gestion de la M d’un point de vu utilisateur est différente de celle de la machine. L’utilisateur a besoin que la M soit gérée de la même façon pour chaque Pu. Cependant la machine alloue la M de façon non-ordonnée. Nous avons donc besoin d’un pont entre les deux façons de faire : la pagination.



La table des pages est située dans la M. Elle contient les adresses physiques de chaque page. Chaque page contient les adresses physiques d’une certaine donnée.

Une adresse virtuelle contient deux parties : l’index de la page et le offset. Donc, nous avons alors que :

v: Virtual Address of some data; pt: Page table; ppa: Page physical address; dpa: Data physical address; mem: M.

ppa = pt [v.page\_index]; dpa = mem[ppa][v.offset].

Performance? Pour améliorer la vitesse d’accès aux données, une partie de la M cache est réservée pour la sauvegarde de l’adresse de certaines tables.

Protection? Pour protéger la M, un bit de validité est associé à chaque frame (portion de la M).

## Tables de pages à deux niveaux

[Youtube](https://youtu.be/KqHNaOrxttM?t=659).