ASR2-Système : la mémoire

Semestre 2, année 2009-2010

Département d'informatique
IUT Bordeaux 1

Mai 2010

Gestion de la mémoire

Multiprogrammation, objectifs et conséquences
- Contexte
- Partage du temps
- Cohabitation en mémoire

Adresses logiques
- Indépendance code/position physique
- Modification du matériel
- Protection inter-processus
- Résumé

Gestion de l'allocation contiguë
- Problème : la fragmentation
- Exemple de scénario
- Solutions

Swapping
- Idée
- Avantages
- Remarque

Segmentation
- Idée
- Réalisation
- Résumé

Espace mémoire paginé

Mémoire virtuelle paginée

Algorithmes de remplacement de page

Mémoire virtuelle segmentée-paginée

Multiprogrammation, objectifs et conséquences

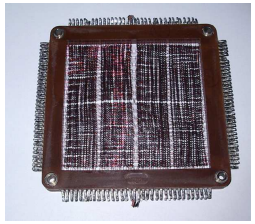
Plan

1. Contexte
2. Multiprogrammation, partage du temps
3. Comment partager l'espace ?

Qu'est-ce qu'un ordinateur ?

(début années 60)

une unité de traitement ...



de la mémoire ...

des périphériques ...

ça coûte très cher

Exemple de configuration

ATLAS (1961) - Université de Manchester

- mots de 48 bits (8 caractères de 6 bits)
- 16K mots de mémoire à tores de ferrite (equiv. 96 K
- 96 Kmots de mémoire à tambour (equiv. 596 Ko)

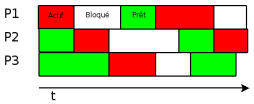
- Addition virgule fixe en 1.6μs, multiplication virgule f 5μs

Atlas

- commercialisé par Ferranti (Electronique, Manchester
  - Ferranti Mark 1 (1951), permier ordinateur commerc
  - Meg (1954) virgule flottante
  - Mercury (1957), mémoire à tores de ferrite
- Prix commercial de l'Atlas : > 1 million de livres sterl

Q : comment rentabiliser ?

R : multi-programmation



(partage du temps)

Multi-programmation

Idées :

- plusieurs programmes en mémoire
- la gestion de périphériques est déléguée
- on profite des "temps morts"
- activation des processus à tour de rôle

⇒ meilleure gestion du temps
(Time is Money)

La cohabitation en mémoire

Mais : cohabitation
dans l'espace mémoire

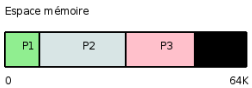
Induit de nouveaux problèmes :

- ne pas gaspiller de place
- protéger les programmes
- sans compliquer la programmation

Partager
la mémoire

Où mettre les programmes ?

Idée ?



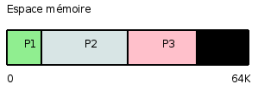
à la suite les uns des autres ?

= allocation
contiguë

L'allocation contiguë

Principes

- Un processus : un espace en mémoire physique
- espace contigu (plage d'adresses)



L'allocation contiguë

Problèmes

1. L'emplacement du code n'est connu qu'au chargemer
2. Les espaces alloués sont
   - de tailles diverses
   - libérés dans un ordre imprévisible

Adresses logiques

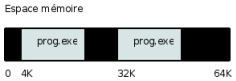
Plan

1. Problème : Indépendance code / position
2. solution : adresses logiques
3. mécanisme de génération d'adresses
4. protection de la mémoire

Indépendance code /position

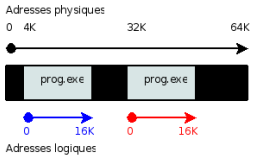
le problème

- Un exécutable peut être chargé à des emplacements d
  (simultanément ou pas)



- Il doit fonctionner sans modification.

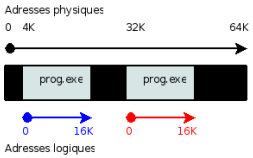
Solution :
adresses logiques



Chaque processus utilise des addresses logi

L'espace mémoire logique d'u
processus commence
toujours à son adresse zéro

Génération
d'adresse
physique ?



adresse physique =
adresse de base + adresse logique

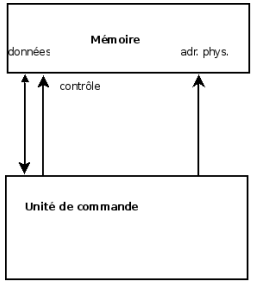
Génération d'adresse

la génération d'adresse physique est faite
- à chaque instruction
- par le matériel.

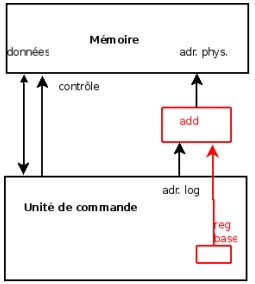
Modification du matériel

Génération d'adresse

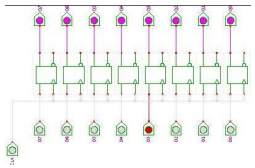
- Un Registre de base contient l'adresse de base du pro
- Additionneur :
  - adresse présente sur le bus d'addresses logiques
  - + registre de base
  - ⇒ bus adresses physiques



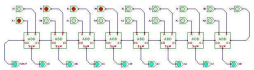
(architecture initiale)



avec registre de base



Registre 8 bits



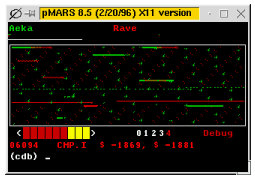
Additionneur 8 bits

(faible coût)

Protection

Objectif

- Éviter qu'un processus n'accède à l'espace mémoire d



Jeu “Core War” (1984)

Protection

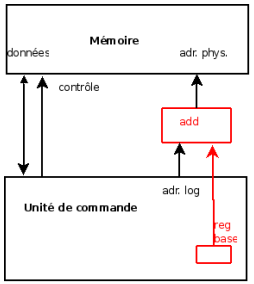
En vrai, on veut éviter.
Une idée ?

idée
une adresse logique valide est comprise entre
- 0
- la taille de l’espace logique (-1)

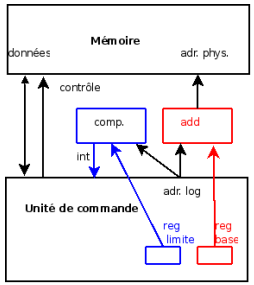
⇒ comparer
adresses logiques
et taille

Modification du matériel (suite)

Vérification d’adresse logique
- Registre limite contenant la taille de l’espace logique processus en cours
- Comparateur : bus adresses logiques + registre limite interruption si dépassement



(avec registre de base)



avec registre limite

comparateur =
soustracteur
+ test de signe

Bilan adresses logiques

1. l'introduction des adresses logiques
   - rend le code indépendant de sa position
   - simplifie le chargement des programmes
   - facilite leur protection
2. la génération d'adresses
   - est réalisée par le matériel
   - est simple (registres, additionneur, comparateur)
   - a un coût faible

Gestion de l'allocation contiguë

Plan

- partage
- phénomène de fragmentation
- y remédier
- Coment l'éviter
- stratégies d'allocation

Gestion de l'allocation contigüe

Initialement

- On dispose d'un espace physique en mémoire (espace

Au cours du temps

- allocations de zones (prises dans l'espace libre)
- libération de zones allouées

La fragmentation

- les allocations et libérations sont faites dans un ordre imprévisible
- tendance à laisser des trous
- les petits trous sont inexploitables

C'est la fragmentation

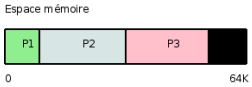
On dit aussi
"gruyérisation"

(à tort)

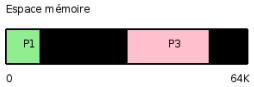
gruyère

emmental

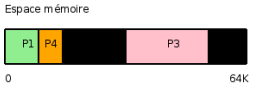
Exemple : un scénario de gr[illegible]fragmentation :



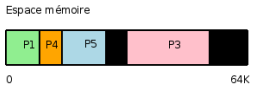
(0) 3 processus sont présents, P2 se termir



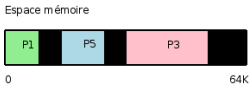
(1) allocation de P4 ...



(3) allocation de P5 ...



(4) libération de P4 ...



(5) l'espace libre est fragmenté

Fragmentation : quels remède ?

Remèdes à la fragmentation

Curatifs

- périodiquement, translater les zones allouées pour rec une grande zone libre
- synonymes : compactage, relocation

Remèdes à la fragmentation (suite)

Préventifs
Limiter l'apparition de la fragmentation

- la stratégie d'allocation first-fit (premier bloc assez g mauvaise
- la stratégie best-fit (plus petit bloc libre qui soit asse est meilleure
- autres stratégies (buddy system)

Résumé

- Fragmentation consécutive à l'allocation de zones cor de tailles diverses
- ce problème se retrouve ailleurs (gestion des disques, delete en programmation, ...)
- les solutions préventives et curatives (best-fit, transla sont plus ou moins satisfaisantes

en pratique, on s'y prend autrement (voir + loin).

Swapping

Plan

1. Idée, motivation
2. Avantages

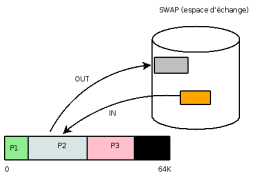
Swapping

Une autre idée

- Transférer sur disque (ou tambour) les processus inac
- les ramener le moment voulu

Motivation

Applications interactives
(l'utilisateur est un périphérique très lent)



Echanges mémoire ↔ disque

Avantages

- on peut charger beaucoup plus de processus en mém virtuelle que la mémoire centrale ne peut en contenir
- permet d'attendre des taux de charge élevés (rentabilisation)

Remarque

Même problème de gestion d'espace sur disque ?

Segmentation

Plan

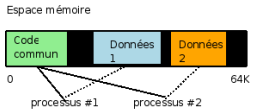
1. Objectif
2. Idée de base
3. Généralisation
4. Tables locales / Globale

Segmentation

Idée de base

- L'espace mémoire d'un processus comporte au moins segments :
  - le segment de code
  - le segment de données.
  - ...
- le segment de code est commun aux différentes instar programme ;
- on préfère charger une seule instance du code en mén
- segment partagé entre les processus

Une idée de réalisation



Deux processus exécutent le même cod

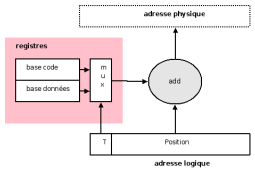
Différencier 2 types d'adresses, selon le premier bit

Bits d'une adresse

T=0 : position dans le segment de code
T=1 : position dans le segment de données

génération d'adresses, comment ?

2 registres de base (code, données)



un multiplexeur sélectionne le registre de b utiliser

On peut généraliser l'idée

Segmentation

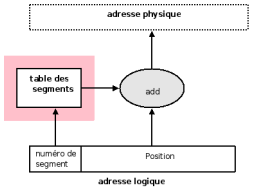
Généralisation

Une adresse logique comporte

- un numéro de segment
- une position (offset) dans le segment

La table des segments

- contient les adresses de basex des segments.
- on y trouve aussi les tailles, les droits d'accès etc.



utilisation d'une table des segments

En fait,
c'est peu plus
compliqué

numéros de segment
- locaux
- globaux

Numéros de segments

Numérotation

- locale : propre à chaque processus
- globale : commune

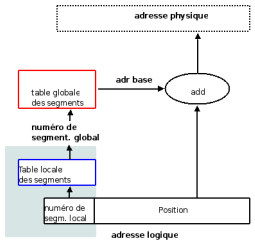
Deux processus peuvent utiliser le même segment réel ave numéros locaux (éventuellement) différents.

Exemple

- P1 : code1(0), bib. trigo(1), données2(2)
- P2 : code2(0), bib. matrices(1). bib. trigo(2), donnée

Tables des segments

- La table locale des segments (TLS) d'un processus fo numéro de segment global à partir du numéro de segi local
- La table globale de segments (TGS) contient la descr chaque segment (longueur, pages, protections ...)



Génération d'adresses, TLS + TGS

Résumé

- adresse logique : numéro de segment, et position
- utilisation d'une table des segments
- permet un gain de place important : on utilise les seg pour les bibliothèques communes.

Espace mémoire paginé

Plan

1. Idée
2. Table des pages
3. Génération d'adresses

Espace mémoire paginé

Contexte

Partage de la mémoire physique par des processus

Problème

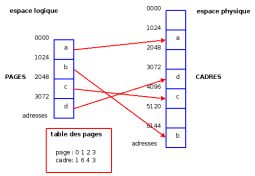
causé par l'allocation et la libération d'espaces de tailles d

Idée

- espace découpé en pages de mêmes tailles
- les pages d'un même espace ne sont pas forcément consécutives
- la table des pages indique la position en mémoire des d'un processus

Plus précisément

- espaces logiques découpés en pages logiques : habitue 1K mots, 2K, 4K, 8K ...
- la mémoire physique est découpée en cadres de pages même taille.
- pour chaque processus, une table des pages établit la correspondance



espace paginé

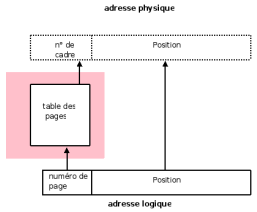
Bits d'une adresse

Numéro de page	Position dans la page

format d'adresse logique

Génération d'adresses

- le numéro de page provient des bits de poids forts
- il est converti en numéro de cadre de page
- combinaison avec les bits de poids faible (position da page)



génération d'adresse

Mémoire virtuelle paginée

Plan

1. Mémoire virtuelle paginée
2. Défaut de page
3. Remplacement de page
4. Algorithmes

Mémoire virtuelle paginée

Principe

Combinaison de la pagination et du swapping.

- la table des pages connait les pages présentes
- interruption en cas de défaut de page (la page référen pas en mémoire)

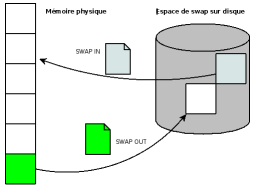
Circuit spécialisé MMU (memory management unit)

Pagination, suite

En cas de défaut de page

Le système d'exploitation reprend la main,

1. il trouve un emplacement libre (au besoin, évacuation disque)
2. y installe la page manquante,
3. met à jour la MMU
4. et relance l'instruction interrompue.



Swapping

Algorithmes de remplacement de page

Pour choisir un emplacement à libérer, en général on choi pages "les moins utiles".
Diverses techniques existent. Voir TD.

Mémoire virtuelle segmentée-paginée

Plan

1. Principe
2. génération d'adresses

Mémoire virtuelle segmentée paginée

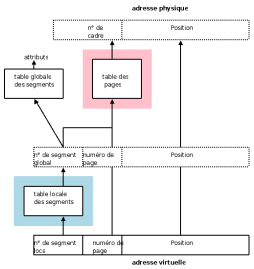
Une adresse virtuelle comporte un numéro de segment, et déplacement.

- chaque processus possède ses numéros de segment.
- chaque segment est composé de pages.

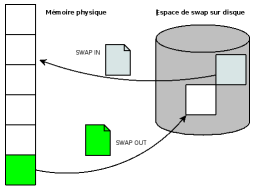
Implémentation

- une table (TLS) traduit le numéro de segment local e
  numéro de segment global.
- le couple (numéro de segment global, numéro de page
  de retrouver le numéro de cadre de page correspondan
- le numéro de cadre de page et la position dans la pag
  forment l'adresse physique.

La MMU utilise une mémoire associative pour mémoriser l
correspondances
(num. de segment global, num. de page → num. de cadre



génération d'adresse



Combinaison avec swapping

IBM 360/67 (1969)

Résumé

- Combinaison de
  - pagination,
  - segmentation,
  - swapping
- permet
  - la cohabitation entre les processus
  - la protection
  - zones partagées
- technique largement adoptée depuis les années 70