Projet Systèmes d'exploitation (ENSIN6U3)

Leonardo Brenner
leonardo.brenner@univ-amu.fr

2012-2013

Description du projet

Le projet du module de *systèmes d'exploitation* pour l'année 2012/2013 sera focalisé dans l'implémentation des algorithmes d'ordonnancement et de gestion de la mémoire.

L'objectif de ce programme est ordonnancer une suite de processus fournis par l'utilisateur au fur et à mesure que le programme s'exécute.

Le programme doit implémenter l'ordonnancement et la gestion de la mémoire tel qu'ils sont décrit ci-dessous.

Initiation du système Au début de l'exécution du programme, l'utilisateur doit fournir les paramètre d'initialisation du système. C'est paramètre sont :

- la taille de la mémoire vive en octets (mémoire RAM) ;
- la taille de la mémoire virtuelle en octets (mémoire RAM + mémoire sur le disque dur) ;
- la taille cadres de pages en octets ;
- le quantum (en unité de temps) ;

Dans ce projet, la mémoire vive, ainsi que la mémoire virtuelle, seront paginées. La taille de la mémoire virtuelle doit être supérieure à la taille de la mémoire vive, vu que la mémoire virtuelle comprenne la mémoire vive plus la mémoire sur le disque dur. La taille des cadres de pages servira à calculer combien de cadres de pages auront les mémoires.

Ces mémoires seront simulées et uniquement le nombre de cadres de pages sont importants pour le fonctionnement du système. Pour représenter la memoire, utilisez un tableau, où chaque position corresponde à un cadre de page.

Le quantum va détermine le nombre d'unité de temps donné à chaque processus (pour ce projet, chaque unité de temps est de 1 seconde).

Voici un exemple pour l'entrée de paramètres du système :

```
Entrez la taille de la memoire vive :  100000
Entrez la taille de la memoire virtuelle :  200000
Entrez la taille cadres de pages : 1000
Entrez le quantum pour l'ordonnancement :  10

Calcul du nombre de cadres de pages de la memoire vive.
Creation de la memoire vive (100 cadres).
Creation de la memoire virtuelle (200 cadres).
```

Ordonnancement de processus Chaque processus est composé d'un couple (d, m), où d est la durée du processus (en unité de temps) et m est la taille du processus (en octets).

La durée du processus sert à l'ordonnancement des processus lorsque la taille du processus servira à la pagination et à la gestion de la mémoire virtuelle.

À partir des données des processus, fournis par l'utilisateur au fur et à mesure que le programme s'exécute, les processus seront ordonnancer selon l'algorithme suivant :

- les processus seront classés en 5 files de priorités (0 à 4) (les priorités seront calculées dynamiquement) ;
- la file de priorité 0 est la plus prioritaire, suivi de la file de priorité 1, 2, 3 et 4.
- chaque nouveau processus soumis par l'utilisateur a, par défaut, la priorité 0 ;
- dans chaque file, l'algorithme Round Robin sera utilisé ;
- au début de chaque unité de temps, le processus qui va s'exécuter, choisi une page au hasard parmi les pages du processus [1] ;

1. La taille du processus divisée par la taille d'un cadre de page donne le nombre de page d'un processus.

– si la page choisi n'est pas présente dans la mémoire vive, elle doit être chargée à partir de la mémoire virtuelle.

Le calcul des nouvelles priorités de chaque processus se fait à chaque 10 re-ordonnancement. Autrement dit, chaque fois qu'un processus se termine ou que son quantum se termine, on compte un re-ordonnancement. Le calcul est fait en utilisant la formule suivante :

$$nouvelle\_priorite = (int)(n-1)/2$$

où n est le nombre de fois que le processus a eu accès au processeur. Par exemple, si un processus a accédé au processeur 8 fois sur 10 re-ordonnancement, il aura une priorité 3, si un processus a accédé 2 fois sur 10, il aura une priorité 0.

Gestion de la mémoire La mémoire du système est divisée en deux parties : la mémoire vive et la mémoire virtuelle.

Chaque mémoire sera paginée et représentée par un tableau.

Lorsqu'un processus est soumis, il est entièrement place en mémoire virtuelle. Ses pages seront chargées en mémoire vive à la demande.

Une fois qu'il n'y a plus de cadre de page disponible (vide) sur la memoire vive, une page doit être remplacée. Pour cela, on va appliquer l'algorithme de la deuxième chance vu en cours.

Dynamique du programme L'utilisateur pourra, à tout moment, rentrer un nouveau processus. Une fois le nouveau processus dans le système, il sera pris en compte dans l'ordonnancement. Qui aura lieu à la fin du quantum ou de l'exécution d'un processus.

L'utilisateur pourra, à tout moment également, demander d'afficher l'état de la file d'attente de processus ainsi que l'état des mémoires.

Voici un exemple d'interface d'entrée :

```
Bienvenu à SimuDummies - a simulateur d'ordonnancement et gestion de memoire

1 - Entrer un nouveau processus;
2 - Afficher la file d'attente de processus;
3 - Afficher les memoires.

Choisir parmi les 3 options :
```

Si l'utilisateur choisi l'option `1` :

```
Entrez la durée du processus : 11
Entrez la taille du processus : 12345

Processus 1 crée avec succes pour une durée 11 et taille 12345 octets (13 pages).
```

Au moment de la création d'un processus, la mémoire nécessaire (nombre de cadres de pages) au processus sera allouée et les pages du processus seront places dans les places allouées.

Si l'utilisateur choisi l'option `2` :

```
Priorité        File d'attente
0               1, 4, 2, 7, 9
1               3, 5, 8
2
3               6
4               10
```

Le premier processus de la file la plus prioritaire est le processus qui est en train de s'exécuter.

Si l'utilisateur choisi l'option `3` :

```
Memoire vive (10 cadres de pages)
(cadre de page: id du processus, page du processus) :
0: 1,2          1: 1,1          2: 1,0          3: 2,5          4: 2,1
5: 3,0          6: 1,3          7: 2,2          8:              9: 2,3

Memoire virtuelle (20 cadres de pages)
(cadre de page: id du processus, page du processus) :
0: 3,2          1: 3,3          2: 2,4          3: 2,0          4: 4,1
5: 3,4          6: 1,5          7: 5,0          8: 5,1          9: 5,3
10:             11:             12:             13:             14: 5,2
15:             16:             17:             18: 3,1         19:
```

L'utilisation de la mémoire vive et de la mémoire virtuelle change au fur et à mesure que des nouveaux processus arrivent et se terminent.

En cour d'exécution, le programme doit afficher une série d'information :

- l'identifiant du processus que s'exécute ;
- la page que le processus a choisi ;
- le moment du re-ordonnancement (processus pris en compte, calcul des priorités) ;
- le chargement et sauvegarde de pages.

Voici un exemple d'affichage :

```
Processus 1 s'execute sur la page 0
Processus 1 s'execute sur la page 2
Re-ordonnancement : processus 3, 4 pris en compte
Processus 2 s'execute sur la page 3
Processus 3 s'execute sur la page 0
Sauvegarde de la page 2 du processus 1
Chargement de la page 0 du processus 3
Re-ordonnancement : processus 5 pris en compte
Re-ordonnancement : calcul des nouvelles priorités
```

Modalités de travail.

On travaille impérativement par binômes. Inclure le noms du binôme dans le fichier `README` et dans chaque fichier source.

Modalités de soumission.

Préparez un répertoire contenant le code source, le fichier `README`, le fichier `MAKEFILE`, et rien d'autre.

Dates importantes

- Aix-Montperrin - Date limite de soumission : **vendredi 26 avril 2013**. Date de la présentation : **mardi 30 avril 2013 à partir de 8 :00**.
- St-Charles - Date limite de soumission : **lundi 29 avril 2013**. Date de la présentation : **jeudi 02 mai 2013 à partir de 8 :00**.

Une pénalité de 10% sera appliquée par jour de retard.

Modalités d'évaluation

Dans l'évaluation de votre projet importance particulière sera donnée aux critères suivants :

Documentation. Votre projet doit contenir un fichier `README` contenant :

- La description du programme adressé à un utilisateur non-expert.
- Une description concise du programme (adressés aux évaluateurs) décrivant les choix d'implémentation, les structures de données utilisés, les résultats accomplis par rapport à chaque tache décrite ci-dessus.

Qualité du Code C. Le code source doit être bien commenté, les noms des variables, des fonctions, des structures, et des types seront bien choisis. On donnera préférence à des fonction de petite taille. On utilisera de macro-constantes aux lieux des constantes explicites.

Organisation du programme en unités logiques. Chaque unité logique doit être implémentée dans un fichier séparé `nom.c` (avec `nom` est bien choisi) ayant un fichier en tête `nom.h` par défaut. La liste des unités logiques (donc la liste du code source) apparaîtra dans le fichier `README`, la compilation sera géré par le programme `make` à l'aide d'un fichier `Makefile`.

Fonctionnement de votre code. Le code source sera compilé au moment de l'évaluation. Écrire donc votre code en suivant le standard POSIX. Vous devez préparer une présentation d'environ 10 minutes pour présenter votre programme et faire ressortir les points forts de votre implémentation. Le fonctionnement de votre code sera testé par rapport aux résultats déclarés accomplis dans le fichier `README`.