POLYTECH'RICM2 : Architecture de systèmes

Durée 3h – Tous documents autorisés – Le barême est indicatif

23 avril 2004

Le sujet comprend 2 parties indépendantes. Ces parties doivent être rendues sur des copies distinctes.

Partie I (10 pts)

I.1) Inversion de priorité (5 pts)

Dans un noyau de processus gérés par priorité, il existe un phénomène dit **d'inversion de priorité** :

1) Expliquer ce phénomène via un diagramme temporel.

2) Décrivez les 2 méthodes permettant d'éviter ce phénomène.

3) Comparez le fonctionnement de ces 2 méthodes via 2 diagrammes temporels.Quels sont leurs avantages/inconvénients respectifs.

I.2) Implementation de synchronisation (5 pts)

On se propose d'étudier l'implantation d'une synchronisation de type évènementiel **Event** sur un noyau de gestion de threads sur monoprocesseur tel que FRED étudié en TD/TP. L'interface du type évènement est definie par les 3 opérateurs suivants `WaitEvent(Event*)`, `Set( Event *)` et `Reset( Event *)`. Une occurrence d'évènement est produite par `Set( Event *)`. Les processus peuvent attendre une occurrence d'un évènement par `WaitEvent(Event*)`. Un évènement de synchronisation est caractérisé par 2 propriétés : la **portée** et la **durée de vie** :

Portée lors d'une occurence d'évènement de portée **unique**, l'occurence n'est visible que par un seul processus au plus. Une portée **globale** implique qu'une occurence d'évènement est visible par tous les processus au plus.

Durée Un évènement peut être **fugace** ou **durable**. La durée de vie d'une occurence d'évènement fugace est nulle. Seuls les processus qui l'attendaient au moment où elle est signalée (`Set( Event *)`) la voient s'ils en ont le droit(cf portée). Une occurrence d'évènement durable existe entre le moment où elle est signalée (`Set( Event *)`) et celui où elle disparait. Elle disparait automatiquement quand elle a été vue par celui ou ceux pour qui elle est visible (cf portée). Elle disparait explicitement par `Reset( Event *)`.

Décrire en pseudo C l'implantation de ces 3 opérateurs pour les quatres types d'évènements suivants :

4) fugace - unique Une occurrence est visible par un processus au plus l'attendant.

5) fugace - global Une occurrence est visible par tous les processus au plus l'attendant.

6) durable- unique Une occurrence est visible par un processus au plus.

7) durable - global Une occurrence est visible par tous les processus au plus.

Rappel : FRED repose sur les opérateurs suivants :

Exclusion mutuelle Elle est assurée par un simple mécanisme de contrôle des interruptions : `MaskOn()` et `MaskOff()`

Temps partagé Les threads sont gérés en temps partagé **sans priorité**. Le thread courant est repéré par le pointeur global `Thread* current`. Les threads prêts sont dans la file `readyQ`.

Suspension-réveil Le thread courant peut abandonner le processeur en se suspendant dans une file par `Suspend( Queue *)`. Un thread peut insérer le premier thread d'une file dans la file des prêts par

`Wakeup(Queue*)`.Il est possible d'insérer/retirer un descripteur de thread `Thread` suspendu indifférement en tête ou en queue d'une `Queue` ou tester si la file est vide.

Partie II (10pts)

II.1) Algorithmes de pagination (3 pts)

On dispose d'un système doté d'une pagination à la demande, suivant les algorithmes A1 et A2. Au cours de son exécution, un programme accède successivement aux pages 1, 5, 2, 5, 1, 4, 1, 5, 3. Le système dispose d'un espace physique de 3 pages. On constate que l'on a successivemment en mémoire les pages suivantes:

```
    Algorithme A1                  Algorithme A2
1  1  1  1  1  2 1 1 1         1  1  1  1  1  1 1 1 1
   5  2  2  2  4 2 4 3            5  2  2  2  4 4 4 3
      5  5  5  5 4 5 5               5  5  5  5 5 5 5
```

1) Expliquer brièvement (20 lignes) les algorithmes FIFO, LRU et LFU.
2) A votre avis, à quoi correspondent les algorithmes A1 et A2?
3) Déterminer dans chacun des cas le nombre de défauts de pages.

II.2) Bibliothèque d'Entrées-Sorties (7 pts)

Pour cet exercice, il est maintenant envisagé d'intégrer des mécanismes permettant exclusivement des accès directs dans des fichiers. Dès la phase d'ouverture, le programmeur doit spécifier la taille des enregistrements de son fichier. Les opérations de lecture et écriture s'effectuent ensuite à partir d'un numéro d'enregistrement.

L'optimisation des accès aux fichiers dans une bibliothèque passe par une gestion d'un tampon structuré comme un ensemble (pool) d'enregistrements qui ne sont pas forcément contigus. Un nombre fixe d'enregistrements (`MAX_ENREG`) est alors préalloué dès l'ouverture du fichier. Ce pool d'enregistrement se remplit au fur et à mesure de l'exécution du programme et des opérations de lecture/écriture. Le pool des enregistrements présents en mémoire est géré avec une stratégie de type LRU. Dans le cas où le pool d'enregistrements est plein, c'est l'enregistrement le moins récemment utilisé qui est remplacé.

```
FICHIER *OuvrirFichierDirect (char *name, char mode, int enreg_size) ;
/* Ouverture du fichier, création, initialisation du pool d'enregistrements */
int LireFichierDirect (FICHIER *f, void *buff, int num) ;
/* Lecture de l'enregistrement de numéro num, chargement dans le pool */
int EcrireFichierDirect (FICHIER *f, void *buff, int num) ;
/* Ecriture de l'enregistrement de numéro num, chargement dans le pool */
int FermerFichierDirect (FICHIER *f)
/* Fermeture du fichier, sauvegarde des enregistrements modifiés */
```

Vous répondrez aux différentes questions en utilisant un langage de programmation de type pseudo C.

1) Donner la nouvelle description de la structure `FICHIER`.

2) Décriver l'implantation de l'opération d'ouverture de fichier `OuvrirFichierDirect`. Vous préciserez plus particulièrement comment vous initialisez le pool d'enregistrements.

3) Décriver l'implantation de l'opération de fermetue de fichier `FermerFichierDirect`. Vous préciserez plus particulièrement comment sont sauvegardés les enregistrements modifiés.

4) Vous utiliserez un algorithme de remplacement des enregistrements de type LRU. Quelles sont les principales opérations sur le pool d'enregistrement qui devront être implémentées et utilisées par les opérations de lecture et d'ecriture?

5) Décriver les opérations de lecture et d'écriture en utilisant les opérations proposées à la question précédente.