Examen de Système et administration

Partie Système

Vendredi 11 mai 2018, 9h–12h

Les documents sont interdits. La calculatrice est autorisée.

1 Questions de cours [8 points]

Question 1.1 Un lave-linge récent est-il pourvu d'un système d'exploitation ? Expliquer pourquoi.

Un lave-linge propose de nombreuses fonctions, mais elles ne peuvent jamais s'exécuter en même temps. Par conséquent, un lave-linge n'est pas pourvu de système d'exploitation.

Question 1.2 Qu'est-ce qu'un interblocage ? Sous quelles conditions un interblocage apparaît-il ?

Un interblocage est une situation où certains processus ne peuvent pas poursuivre leur exécution faute de ressources.
Un interblocage se produit si les 4 conditions suivantes se produisent :

- exclusion mutuelle (toute ressource ne peut être allouée qu'à un seul processus au plus)
- détention et attente (tout processus impliqué dans l'interblocage détient des ressources nécessaires à d'autres processus interbloqués, et attend des ressouces détenues par des processus interbloqués)
- non réquisition (aucun aurbitre ne peut réquisitionner les ressources détenues par un processus)
- attente circulaire (il existe un circuit de type P_0 attend les ressources de P_1, qui attend les ressources de P_2,..., qui attend les ressources de P_0)

Question 1.3 Définir la fragmentation interne, la fragmentation externe, et donner un exemple pour chacun de ces deux termes.

La fragmentation interne se manifeste lorsque qu'on alloue trop de place à un fichier. Cette situation est possible car l'espace d'un disque est divisé en blocs (un bloc représente typiquement entre 512 octets et 4048 octets). Par exemple, si on veut stocker un fichier de 513 octets, alors deux blocs sont nécessaires, et le second mobilise 512 octets alors qu'un seul octet y est stocké.
La fragmentation externe se manifeste lorsque l'espace libre d'un disque dur est suffisant pour recevoir un nouveau fichier, mais qu'il est impossible d'allouer un espace contigu pour enregistrer le fichier. Par exemple, si un disque dur a 10 blocs libres sous la forme de deux ensembles contigus de 5 blocs, et que l'on veut enregistrer un ficher de 6 blocs, on est obligé de fractionner le ficher et de le répartir dans les deux ensembles de 5 blocs libres.

Question 1.4 A quoi servent les conditions de Bernstein ? Donner un exemple de leur utilisation.

Les conditions de Bernstein servent à vérifier si l'on peut diviser une fonction séquentielle en plusieurs parties indépendantes afin d'exécuter ces parties en parallèle. Soit la fonction écrite en C++ effectuant les opérations suivantes :

```
void f(int a, int b, int &c, int &d)
{
c = a + b;
d = 2*a;
}
```

On peut la décomposer en deux parties, `f1` et `f2` pouvant être exécutées en parallèle :

```
void f1(int a, int b, int &c)
{
c = a + b;
}
```

et

```
void f2(int a, int b, int &d)
{
d = 2*a;
}
```

En effet, les trois conditions de Bernstein sont satisfaites : la première stipule que `f1` n'a pas d'entrées parmi les sorties de `f2`, la seconde stipule que `f2` n'a pas d'entrées parmi les sorties de `f1`, la dernière dispose que `f1` et `f2` n'ont pas de sorties en commun.

Question 1.5 Discuter les avantages et les inconvénients de l'allocation chaînée.

L'allocation chaînée présente l'avantage de nécessiter un overhead minimal : pour tout fichier, il suffit de conserver l'adresse du premier bloc de données. Un autre avantage est que la taille des fichiers stockables n'est limitée que par la capacité du disque dur, et que le fichier peut être stocké de manière fragmentée sans inconvévient.
Les désavantages sont que l'on a un mélange entre données utiles et données de gestion (pointeur vers le bloc suivant), ce qui fait qu'une erreur sur le disque peut compromettre l'accès à un fichier. L'accès aléatoire à un fichier peut être difficile, car il faut parcourir tous les pointeurs pour parvenir à la position désirée, ce qui peut être long étant donné qu'un disque dur est lent.

Question 1.6 On copie 100 Mo d'un CD ROM vers un disque dur en 1 minute. Indiquer s'il est possible de déduire les caractéristiques suivantes :

- Le débit du lecteur de CD ROM
- Le débit du disque dur
- La fréquence du microprocesseur

On justifiera si nécessaire les hypothèses simplificatrices que l'on fera.

On peut en déduire le débit du lecteur de CD ROM, car c'est le dispositif le plus lent (par rapport au disque dur et au processeur). Ce débit est de 100 Mo par minute. Les hypothèses simplificatrices dans ce raisonnement sont que l'on a considéré que le débit du disque dur et la vitesse du processeur étaient illimités. En réalité, elles ont un impact sur ce résultat, mais on peut le considérer comme négligeable.

2 Les deux boulangers jouent à pile ou face [2 points]

On rappelle le cas des deux boulangers, du pain et du couteau : le premier boulanger réserve le couteau (sa première ressource), puis il réserve le pain (sa deuxième ressource) et fait son travail : produire une tranche de pain, puis il libère les deux ressources. Le second boulanger agit de la même façon, mais il commence par réserver le pain (sa première ressource), et le couteau ensuite (sa deuxième ressource). Dans cet exercice, on modifie le comportement des boulangers comme suit. Après avoir obtenu sa première ressource, tout boulanger attend une seconde, et tire à pile ou face. S'il obtient pile, il rend la ressource, attend une seconde, et tente à nouveau de réserver sa première ressource. S'il obtient face, il demande sa seconde ressource (et l'attend aussi longtemps que nécessaire), puis il coupe du pain pendant une seconde, libère le pain et le couteau, et recommence le cycle (réservation de la première ressource, etc).

Question 2.1 A l'instant $t = 0$, toutes les ressources sont disponibles, et les deux boulangers commencent leur fonctionnement comme indiqué ci-dessus. Dans quel état se trouve le système à l'instant $t = 0.5$ seconde en matière

d'interblocage ? On justifiera la réponse.

A l'instant $t = 0.5$ seconde, le premier boulanger a réservé le couteau, et le second boulanger a réservé le pain. Ils sont tous les deux en train d'attendre une demi-seconde, avant de tirer à pile ou face. Le système n'est pas interbloqué, car un boulanger peut encore libérer la ressource qu'il détient. Cependant, si les deux boulangers tirent face, alors on observera un interblocage. Le système est dans un état non sûr, car il ne dépend pas de nous que l'interblocage puisse être évité. On a, à cet instant, une chance sur quatre que l'interblocage se produise (si les deux boulangers tirent face).

3 Interblocages [4 points]

On considère un système gérant cinq types de ressources A, B, C, D, et E, partagées entre quatre processus.

Processus	Allocation *A B C D E*	Maximum *A B C D E*
p_0	1 0 2 1 1	1 1 2 1 2
p_1	2 0 1 1 0	2 2 2 1 0
p_2	1 1 0 1 0	2 1 3 1 0
p_3	1 1 1 1 0	1 1 2 2 1

Seules une ressource C, une ressource D et une ressource E sont actuellement disponibles.

Question 3.1 L'état initial décrit ci-dessus est-il sûr ?

On construit la matrice *Besoin* afin de pouvoir appliquer la procédure `TestSur()` de l'algorithme du banquier.

Processus	Besoin *A B C D E*
p_0	0 1 0 0 1
p_1	0 2 1 0 0
p_2	1 0 3 0 0
p_3	0 0 1 1 1

Avec $Disponible = [0\ 0\ 1\ 1\ 1]$, on peut terminer le processus p_3, ce qui conduit à $DispoTest = [1\ 1\ 2\ 2\ 1]$. Ces ressources sont suffisantes pour terminer p_0, ce qui conduit à $DispoTest = [2\ 1\ 4\ 3\ 2]$. On peut alors terminer p_2, ce qui conduit à $DispoTest = [3\ 2\ 4\ 4\ 2]$, enfin terminer p_1 est possible, ce qui conduit à $DispoTest = [5\ 2\ 5\ 5\ 2]$. Le système est dans un état sûr, et la séquence des processus est (p_3, p_0, p_2, p_1).

Question 3.2 Le processus p_0 demande une ressource A. Comment répondre à cette demande ?

On applique l'algorithme du banquier. Cette demande n'est pas raisonnable (on dépasse les besoins maximaux déclarés pour p_0), il convient donc de la rejeter.

Question 3.3 Le processus p_1 demande une ressource C. Comment répondre à cette demande ?

Cette demande est raisonnable, et peut être satisfaite avec les ressources disponibles, on l'accepte temporairement. Or il se trouve (on le voit en lançant `TestSur`) qu'on ne peut pas terminer de processus, donc l'état obtenu n'est pas sûr. Il convient donc de rejeter cette demande.

4 Pagination à la demande [6 points]

On considère un système comportant 3 cadres. Les pages P_4, P_8 et P_9 se trouvent dans les cadres 1, 2 et 3 respectivement, et ont été référencées dans l'ordre suivant : P_8 d'abord, P_4 ensuite, et enfin P_9.

Question 4.1 Donner la matrice de l'algorithme LRU ainsi que l'indicateur entier associé à chaque page, et l'occupation des cadres.

Pour construire la matrice de l'algorithme LRU (Least Recently Used), on part d'une matrice 3×3 (car il y a trois cadres) remplie de zéros. Comme P_8 est la première page chargée, et qu'elle se trouve dans le cadre 2, on a la matrice de gauche ci-dessous. On ajoute ensuite P_4 dans le cadre 1, et on obtient la matrice de droite :

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{matrix} - \\ P_8 \\ - \end{matrix} \qquad \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{matrix} P_4 \\ P_8 \\ - \end{matrix}$$

Pour terminer, on ajoute la page P_9 dans le cadre 3, puis on calcule l'entier associé à chaque cadre :

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \begin{matrix} P_4 & 2 \\ P_8 & 0 \\ P_9 & 6 \end{matrix}$$

Question 4.2 Citez d'autres algorithmes qui pourraient être utilisés à la place de LRU.

On pourrait utiliser FIFO à la place de LRU, même si cet algorithme est trop simpliste, puisqu'il désigne la page la plus ancienne comme victime, ce qui n'est pas toujours un bon choix (une page peut être ancienne et très souvent utilisée. L'algorithme de la seconde chance peut également être utilisé, il est semblable à FIFO, mais utilise un bit de référence qui est mis à 1 chaque fois que la page est référencée. L'algorithme « optimal » ne peut pas être mis en œuvre, car il nécessite de savoir avec précision quelles seront les prochaines pages demandées, ce qui est impossible à savoir a priori.

Question 4.3 Un processus demande à lire la page P_6. Expliquer ce qui se passe, et effectuer l'algorithme LRU.

La page P_6 n'est pas présente en mémoire, la demande du processus provoque un défaut de page. On choisit alors une page victime (qui sera déplacée sur le disque dur, afin de libérer un cadre pour P_6. Le choix de la page victime se porte sur P_8 car elle est associé au plus petit entier. Le cadre 2 est ainsi libéré est alloué à P_6 :

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{matrix} P_4 & 0 \\ P_6 & 5 \\ P_9 & 4 \end{matrix}$$

Question 4.4 Un processus demande à lire la page P_4. Expliquer ce qui se passe, et effectuer l'algorithme LRU.

La page P_4 est présente en mémoire, la demande du processus ne provoque donc pas de défaut de page. On met cependant à jour la matrice et l'indicateur entier des pages pour tenir compte du référencement de P_4 :

$$\begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{matrix} P_4 & 3 \\ P_6 & 1 \\ P_9 & 0 \end{matrix}$$