Systèmes d'exploitation centralisés

Deuxième Année Informatique et Mathématiques Appliquées
Durée : 1 heure 45 (Les documents distribués en cours et TD sont autorisés)

6 juin 2012

1 Principes généraux (10 points)

1.1 Notion de binaire objet

Un système d'exploitation exécute des programmes représentés dans un format binaire. Ce format, appelé binaire objet, caractérise toute famille de système.

Questions (1 point par question)

1. Quelle est la conséquence (négative) de l'existence de plusieurs formats de binaire objet vis-à-vis de la portabilité des programmes exécutables ? Considérer le cas de systèmes à format binaire différent s'exécutant sur des processeurs matériels identiques.
2. Donner les principales sections structurant un binaire objet.
3. On distingue en fait deux types de binaire : le format binaire objet ré-éditable et le format binaire objet exécutable. Pour quelle raison et quelles différences existent entre ces deux formats ?
4. Préciser les composants logiciels d'un système d'exploitation qui produisent l'un ou l'autre des formats précédents.
5. Sous quel type de format binaire objet sont représentées les bibliothèques de sous-programmes associés à des langages compilés tels que C ou Ada ?

1.2 Confinement des erreurs et robustesse d'un noyau

Un noyau de système d'exploitation exécutant plusieurs programmes à la fois, des mécanismes sont nécessaires pour garantir que les interférences entre programmes via les ressources matérielles ne rendent pas leur exécution incorrecte et sont bien contrôlées.

Questions (1 point par question)

6. Sur quel concept repose la supervision de l'exécution d'un programme dans un système d'exploitation ?
7. Préciser quelques ressources **matérielles** dont on doit contrôler l'usage par allocation/libération durant l'exécution d'un programme.
8. Pourquoi le noyau d'un système d'exploitation doit être protégé vis-à-vis de l'exécution des programmes applicatifs ? Préciser les différentes formes de protection nécessaires.
9. Un programme peut comporter une instruction inexécutable. Comment traite-t-on ce genre d'événement de façon à assurer le confinement de l'erreur et la poursuite de l'exécution du système et des autres programmes de façon transparente. En particulier, à quel mécanisme matériel fait-on appel ?
10. Préciser deux types de ressources **logiques** offertes par les noyaux de système d'exploitation pour que des programmes puissent échanger et/ou enregistrer des données.

2 Ordonnancement des processus(10 points)

On considère le problème d'ordonnancement des processus dans un noyau de système d'exploitation c'est-à-dire les stratégies d'allocation de temps processeur aux processus prêts à s'exécuter. Il peut exister plusieurs processeurs réels (architecture multi-processeurs ou multi-cœurs).

On envisage plusieurs solutions toutes fondées sur l'affectation d'une **priorité** aux processus et sur le partage du temps processeur **par quantum**. Les processus prêts les plus prioritaires reçoivent des quata sans préemption c'est-à-dire que la création d'un nouveau processus prêt de priorité supérieure à celle des processus actifs n'interrompt pas l'un d'eux dans l'exécution de son quantum.

De nouveaux processus prêts sont sporadiquement créés et leur **profil d'exécution** peut évoluer dynamiquement passant de phases **interactives** (entrées/sorties fréquentes) à des phases **calculatoires**.

2.1 Stratégie à quantum de durée unique

Dans cette stratégie, la durée d'un quantum est unique, quelle que soit la priorité attribuée au processus. Chaque processus est créé avec une priorité prise sur un intervalle $[0..PRIO[$: la priorité 0 est maximale et $PRIO - 1$ est minimale. Une file d'attente des processus prêts est associée à chaque priorité.

Lorsqu'un processeur se libère, un processus prêt de priorité pr reçoit un quantum de temps **si et seulement si** aucun processus prêt de priorité supérieure n'existe **et** s'il est en tête de la file d'attente pr des processus prêts à s'exécuter ($\equiv$ il est en tête de la file la plus prioritaire non vide).

En fin de quantum, l'ordonnanceur remet le processus actif en fin de la file d'attente correspondant à sa priorité (stratégie du tourniquet) et il choisit alors le processus prêt le plus prioritaire pour le prochain quantum. Si un processus actif se bloque avant la fin de son quantum, le processus sera de façon similaire réinséré en fin de file d'attente selon sa priorité mais seulement **après son déblocage**.

Questions (2 points par question)

11. Expliquer pourquoi cette stratégie présente un risque de famine en expliquant un scénario précis conduisant à une situation de famine pour certains processus qui ne deviennent jamais actifs.
12. On propose une variante de cette stratégie : **sur fin de quantum**, le processus actif de priorité pr est réinséré en fin de la file de priorité supérieure $(pr - 1)$ par l'ordonnanceur si $pr > 0$. On augmente donc si possible sa priorité au fur et à mesure de son exécution. Quel profil de processus est favorisé par cette stratégie et pourquoi ? Cette stratégie comporte-t-elle encore un risque de famine ?
13. On propose une autre variante : tout processus actif de priorité pr **qui se bloque** est réinséré en fin de la file de priorité supérieure $(pr - 1)$ par l'ordonnanceur si $pr > 0$. Quel profil de processus est favorisé par cette stratégie ? Cette stratégie comporte-t-elle encore un risque de famine ?

2.2 Stratégie à quantum de durées distinctes

Dans cette stratégie, on conserve la gestion des priorités mais, à chacune des priorités, est associé un quantum de durée distincte. À la priorité maximale 0 correspond un quantum de durée minimale d et à la priorité minimale $PRIO$ correspond un quantum de durée maximale $PRIO \times d$. La stratégie de l'ordonnanceur consiste à :

- réinsérer un processus actif de priorité pr en fin de file de priorité $(pr + 1)$ plus faible **si et seulement si** il consomme son quantum **et** n'est pas encore de priorité minimale$PRIO - 1$;
- réinsérer, lorsqu'il redeviendra prêt, un processus de priorité pr en fin de file de priorité $(pr - 1)$ plus forte **si et seulement si** il se bloque avant la fin de son quantum et n'est pas déjà de priorité 0.

Questions (2 points par question)

14. Expliquez comment cette stratégie essaie d'adapter de manière optimale la durée du quantum selon l'évolution dynamique du profil d'exécution du processus. ?
15. Comporte-t-elle un risque de famine ?