Série d'exercices #11

IFT-1215

April 2, 2014

18.6

L'algorithme d'ordonnancement multi-niveaux à rétroaction (*multi-level feedback queue scheduling*) ressemble à du FIFO au niveau le plus élevé et à du *round-robin* au niveau le plus bas, pourtant il se comporte souvent mieux que chacun des deux en terme des objectifs de performance. Expliquer pourquoi.

18.7

La latence d'un programme est le temps que l'utilisateur doit attendre avant que le programme finisse son exécution. Pour un usage interactif, la latence moyenne est une caractéristique importante d'un algorithme d'ordonnancement.

1. Expliquer pourquoi.
2. Comment un ordonnanceur peut-il réduire la latence?
3. Quel algorithme donne la meilleure latence moyenne?
4. Prouver que cette latence moyenne est effectivement optimale.
5. Pourquoi le noyau Linux n'utilise-t-il pas cet algorithme?

18.?1

Soit une machine LMC à deux *cores* qui partagent la même mémoire. Écrire un morceau de code qui incrémente d'exactement 1 la case mémoire `99`, de sorte que si chacun des deux cores exécutent ce morceau de code N fois, alors la case aura bien une valeur de $2N$ (en présumant qu'on commence à 0) et pas autre chose.

18.29

Qu'est-ce qu'un système *temps-réel*? Discuter l'impact d'un système temps-réel sur la conception du système d'exploitation, tels que les algorithmes utilisés et les fonctionnalités offertes.

18.39

Soit un système à C processeurs qui doit gérer l'exécution de P processus, où $P > C$. Considérer deux choix de l'ordonnanceur: soit il peut permettre à n'importe quel processus de s'exécuter sur le premier processeur libre, soit il peut limiter chaque processus à toujours s'exécuter sur le même processeur. Discuter des avantages et inconvénients de chacune des deux options.