Systèmes d'exploitation - Ordonnancement/Scheduling

Othmane AJDOR

2018-2019

Table des matières

1 Fondamentaux

1.1 Qui, où, quoi ?

Dnas un ordinateur, on réalise de nombreuses opérations en parallèle où le chef d'orchestre est l'OS. C'est lui qui va décider de l'unité de calcul/stockage, de la date d'execution et qui va receuillir le résultat associé à l'operation.
On retrouvera dans l'OS plusieurs chefs d'orchestres (ordonnanceurs), notamment celui des processus/threads.

On se confronté à plusieurs problèmes :
— Connaitre le futur est difficile, on doit donc prendre des décisions à la volée avec une information partielle.
— Structure de données centrale : liste des tâches prêtes.
— Algorithme de listes **[Graham, 66, 69, 73]** => lorsqu'une ressource est disponible, y mettre immediatement une tâche prête en prenant la plus prioritaire de la liste. Les taches dans la liste ne sont pas forcement toutes prêtes, on va donc preferer prendre une moins prioritaire et prête qu'une plus prioritaire mais pas prête.

1.2 Complexité et analyses

Les problèmes d'ordre sont NP complets sauf les triviaux (1 processeur, Cmax), (2 processurs, Cmax) <- NP Complet.

1.2.1 Algorithme de liste

1.2.2 Approximation duale [Schmoys, Hochbaum]

1.2.3 Algorithme de liste et tâches [parallèle | avec precedence]

1.2.4 Taches paralleles indépendantes

2 Implantations

2.1 Disques

2.1.1 Algorithme d'ordonnancement : ascenseur

— la tete fait des va et vient
— les acces qui sont sous la tete

2.2 SSD

Agglomeration des acces pour faire des acces plus gros
=> systemes de fichiers qui rend les acces "locaux" (ext4, NTFS, HFS+ fonctionnent en ko).

Les SSD fonctionnent en SuperBlock (Mo), on fait un reset à 0 du bloc et on inscrit bit à bit tous les bits qu'il faut passer de 0 à 1. Ce qui veut dire que sur un SSD, on fait un reset du SuperBlock pour ecrire quelques ko (no bueno).

2.3 Reseaux

Les reseaux envoient des paquets, lequel il faut envoyer en premier et quand ?
Il existe sur Linux un script appelé **wonder shaper** qui permet de limiter la bande passante d'un ou de plusieurs network adapters en utilisant iproute.

2.4 Processus/Thread

2.4.1 Linux 1.0

— liste unique de tâches prêtes
— elle est triée par priorité (fo de priorité goodness)

C'est plus simple à utiliser et à régler.
Son défaut, le mutex et la manipulation/insertion de la liste, il faut la garder triée pour que ça soit en O(n).

Les ordonnanceurs en O(1) utilisent des listes par priorité.
Quand la liste "crédits" est vide :
— on re-crédite les expirés (en fo de leur priorité)
— on inverse les deux ensembles de listes
C'est simple à utiliser mais difficle à régler ou calibrer la fonction de crédits qui dit quand est ce qu'on passe.

Les ordonnanceur en O(log(n)) (CFS) utilisent une "liste" (Red-black tree, arbre binaire bien balancé) de priorité, avec une fo de priorité.

2.5 Multi-Coeurs/Processeurs

Contention du **lock** protegeant la "liste" => plusieurs listes (1 par coeur/processeur) => pas de lock.
Le défaut c'est que maintenant on dispose d'une liste par proc/coeur, ce qui veut dire que certains coeurs peuvent rester sans activité, il faut donc equilibrer les listes.

Vol de travail => un voleur (inoccupé) va choisir (random) une victime et lui prendre des tâches. Une victime c'est bien, deux c'est mieux (c'est le prof qui l'a dit), ça permet d'equilibrer rapidement.

Daemon pour équilibrer. De temps en temps, vole les riches pour donner aux pauvres.
Ca coute un peu de temps en temps

3 Passage à l'echelle