Critère d'équité en ordonnancement de grille

E. Medernach, Eric Sanlaville

Critère d'équité en ordonnancement de grille

E. Medernach[1,2] et E. Sanlaville[2]

[1] Laboratoire de Physique Corpusculaire, Campus des Cézeaux, 63177 AUBIERE Cedex
medernach@clermont.in2p3.fr
[2] LIMOS, Campus des Cézeaux, 63177 AUBIERE Cedex
Eric.Sanlaville@math.univ-bpclermont.fr

1 Contexte : la grille EGEE et la gestion des sites

L'intérêt d'une grille de calcul est de permettre le partage de ressources de calcul et/ou de stockage. Dans un tel environnement concurrentiel apparaît le problème de distribuer ces ressources à la fois de façon efficace et équitable. Le contexte de notre travail est la grille EGEE qui se compose de 200 sites sur 50 pays pour un total de plus de 30000 CPUs. Un site d'EGEE se trouve au LPC-IN2P3 de Clermont Ferrand. L'infrastructure de grille EGEE est destinée aux applications de recherche scientifique : les applications pilotes d'EGEE sont la physique des particules et les applications biomédicales. Plus d'une centaine de scientifiques à travers le monde ont soumis environ 2 millions de jobs durant les dix premiers mois de l'année 2005. Ces jobs varient, de scripts très courts à des calculs s'étendant sur plusieurs jours.

Chaque utilisateur appartient à un groupe et est identifié et connu par chaque site. L'objectif de la grille est de partager la puissance de calcul entre les utilisateurs ; le fait que les utilisateurs utilisent plusieurs sites permet de niveler la charge entre les sites. Les intergiciels de grille actuels ont recours à une architecture centralisée au dessus des moyens existants. Malheureusement à grande échelle ce choix crée un goulet d'étranglement et des problèmes de tolérance aux pannes. Au niveau global, il semble que la gestion entièrement centralisée des jobs entraîne de nombreux dysfonctionnements. En effet, l'absence de coordination entre sites ainsi qu'une latence importante des informations conduisent à de mauvais choix de placement des tâches.

Au niveau local (gestion d'un site), il est fondamental pour une bonne gestion d'offrir à chaque utilisateur en concurrence un accès *équitable* aux ressources. Par exemple dans les problèmes de files d'attentes des études montrent qu'il est plus important d'être servi équitablement que de diminuer l'attente [3].

Nous nous plaçons au cours de cette présentation au niveau local sur un site. Nous étudions comment ordonnancer les requêtes des utilisateurs de telle sorte que la part reçue par chacun ainsi que la part donnée aux groupes soit la plus équitable possible, dans un sens que nous cherchons à définir.

2 Formalisation du problème étudié

La soumission des tâches par utilisateur est soit irrégulière soit périodique pour certaines tâches spécifiques. Le modèle d'ordonnancement sous-jacent est celui de tâches indépendantes exécutées sur des machines identiques en parallèle. L'ordonnancement se fait en ligne, le site reçoit les tâches au cours du temps sans connaître précisément leurs durées d'exécution. Celles-ci devraient être modélisées par des variables aléatoires différentes suivant le groupe de l'utilisateur. L'utilisateur peut spécifier une durée maximale pour ses tâches mais en pratique cette mesure se révèle imprécise.

Definition 1. *Le problème mono-site et mono-groupe est donc défini par :*
- *m machines identiques*
- *n utilisateurs*
- *$p_{j,k}$: durée du j^e job de l'utilisateur k, $r_{j,k}$: date d'arrivée. Les jobs sont supposés mono-processeurs et indépendants les uns des autres.*
- *L'objectif est de maximiser l'équité.*

Selon des études préliminaires [2], le temps entre deux arrivées sur les clusters locaux du LPC peut être modélisé par une distribution hyper-exponentielle par utilisateur. Les durées des tâches varient beaucoup (de quelques secondes à plusieurs jours) et suivent une loi log-linéaire par morceaux.

3 La notion d'équité

Lorsqu'on minimise le makespan on satisfait le système car l'ordonnancement obtenu est alors le plus "compact" possible. Mais on ne satisfait pas forcement les utilisateurs car certains peuvent être servis moins bien que d'autres. Ainsi minimiser une fonction économique globale qui ne prend pas en compte la présence de multiples utilisateurs peut aboutir à une situation injuste pour certains d'entre eux. Par exemple une politique favorisant les longs jobs comme LEPT (Longest Estimated Processing Time) donne de bons résultats pour le makespan mais pénalise les petits jobs, donc un utilisateur chez qui ce type de jobs prédomine.

Critère individuel Chaque utilisateur utilise la puissance de calcul du site afin d'accélérer ses temps de traitement. Il évalue donc la performance que lui propose un site par la vitesse de calcul accordée. L'objectif du site est de servir chaque utilisateur de façon équitable en offrant une puissance de calcul bien répartie. Nous proposons de mesurer le niveau de service reçu individuel par la vitesse d'exécution moyenne obtenue de la part du site.

Definition 2 (Vitesse d'execution). *La vitesse d'exécution obtenue par l'utilisateur k est le ratio entre la quantité de calcul fournie au site par l'utilisateur k et la durée de traitement de cette quantité de calcul.*

La vitesse d'exécution peut aussi être appelée débit de calcul. Clairement, la différence face aux critères classiques est la prise en compte de l'utilisateur considéré comme un sous-ensemble de jobs indépendants.

Mesures pour l'équité Le problème de l'équité est un problème bien connu dans les réseaux telecoms. Les algorithmes de contrôle de congestion cherchent à allouer à chaque utilisateur un flux sur un graphe le plus équitablement possible en évitant un possible engorgement du réseau. Par exemple la norme ATM propose pour l'équité de maximiser le minimum des débits [1]. Cependant le modèle du réseau est continu, et les flux calculés sont constants.

Un ordonnancement donné est caractérisé par son vecteur des vitesses d'exécution. Nous cherchons donc une mesure sur les vecteurs de $\mathbb{R}_+^n$. Cette mesure peut être définie comme une fonction de $\mathbb{R}_+^n$ dans $\mathbb{R}$. Nous proposerons d'abord une définition de l'équité via un certain nombre de caractéristiques désirables.

Definition 3 (Fonction équitable). *Une fonction f symétrique est dite équitable si elle vérifie :* $\forall i,j \leq n,\ \min(x_i,x_j) < \min(x'_i,x'_j) \Rightarrow f(..,x_i,..,x_j,..) \leq f(..,x'_i,..,x'_j,..)$ *Les points de suspension indiquent des valeurs identiques entre elles.*

L'exigence de symétrie pour la fonction d'équité entraîne l'impartialité, cela exprime qu'aucun utilisateur n'est privilégié car f est invariante pour toute permutation des individus. Les indices peuvent être égaux ce qui signifie que si on augmente l'allocation d'un seul utilisateur on augmente la fonction d'équité. Le critère d'équité est équivalent au fait que diminuer l'allocation à un utilisateur pour augmenter celle d'un autre déjà plus favorisé diminue l'équité. En effet si on passe de x' à x soit la situation est dégradée pour chacun des utilisateurs soit la situation de celui qui a le moins empire alors que celle de l'autre s'améliore. On montre alors que toute fonction f continue qui vérifie ces conditions est telle qu'il existe une fonction continue croissante g telle que $f(x_1,x_2,\ldots) = g(\min(x_1,x_2,\ldots))$. En d'autres termes, une fonction vérifiant ces trois caractéristiques ne dépend que du minimum du vecteur d'entrée. Ceci conduit à un ordre partiel qui ne dépend que de la plus faible vitesse d'exécution. Cette conclusion n'est pas très satisfaisante.

Si on cherche un ordre dans $\mathbb{R}_+^n$ au lieu d'une fonction scalaire on peut réecrire la définition de l'équité, on trouve alors que le seul ordre qui vérifie cette définition est l'ordre lexicographique sur les vecteurs triés. Remarquons que cet ordre est un sur-ordre de l'ordre de *majoration* [4] utilisé pour l'étude de protocoles réseaux.

Références

1. D. Bertsekas and R. Gallager : Data Networks. Prentice-Hall, Englewood Cliffs, New Jersey, 1992.
2. Emmanuel Medernach : Workload Analysis of a Cluster in a Grid Environment. JSSPP 2005 : 36-61
3. Larson, R.C. Beyond the physics of queueing (1998)
4. Rishi Bhargava, Ashish Goel, and Adam Meyerson : Using approximate majorization to characterize protocol fairness. In SIGMETRICS/Performance, pages 330–331, 2001.