Minimisation des coûts énergétiques dans un centre de calcul

Christelle Jaulin[1], Raja Rebai[1], Lynda Rousseau[1]

[1] EURODECISION

9A rue de la porte de Buc, 78000 Versailles (France)

{christelle.jaulin@eurodecision.com, raja.rebai@eurodecision.com, lynda.rousseau@eurodecision.com}

Mots-clés *: quadratique, local solver, méthode kangourou, Gurobi, machines virtuelles*

1 Introduction

Les centres de calcul sont des salles contenant du matériel informatique très efficace et compétitif. Ce matériel est mis à disposition pour effectuer des calculs scientifiques nécessitant un niveau de fiabilité et de performance élevé. La puissance de calcul proposée évolue très fortement. En 2006, le centre de calcul TERA 10 offre une puissance de calcul de 52.8 téraflops. En 2010, le centre de calcul TERA 100 offre 1,05 pétaflops soit un peu plus de 1015 opérations par seconde sur des nombres flottants. La consommation électrique des centres de calcul suit cette progression. La capacité du bâtiment du TERA 10 est de 3MWatt tandis que celle du TERA 100 est de 10MWatt. Elle offrira 15MWatt en 2011. D'autre part, les centres de calcul produisent beaucoup de chaleur et il est nécessaire de refroidir le matériel informatique et la salle le contenant. Pour cela, les systèmes de refroidissement consomment eux aussi de l'énergie : de 5 à 8 kW par baie pour le TERA 10 et 30 à 40 kW par baie pour le TERA 100. Dans ce contexte, il devient nécessaire de veiller à minimiser au mieux la dépense énergétique pour des raisons tant écologiques qu'économiques. C'est l'objectif du projet COOL-IT dans lequel EURODECISION est impliqué. Il s'agit de minimiser la consommation due à l'exécution des tâches en les regroupant sur un nombre minimum de serveurs pour éteindre ceux qui ne sont pas utilisés. Nous proposons une approche basée sur l'utilisation de machines virtuelles et la résolution d'un programme quadratique. La virtualisation consiste à exécuter plusieurs opérations système sur un seul serveur comme il le ferait sur des serveurs séparés.

2 Formulation du problème

Soit n le nombre de tâches à affecter. Soient d_c^i , d_m^i et d_{io}^i la demande en fréquence CPU, la demande en mémoire et la demande en I/O de la tâche $i \in [1, n]$. Soit r_i un booléen égal à 1 si la tâche i doit être exécutée seule sur un serveur et 0 sinon. Soient c_m^j et c_{io}^j la capacité mémoire et I/O d'un serveur j. Soit p_i un facteur de pénalité relatif à la tâche i si la tâche i n'est pas exécutée. Soit m_i une pénalité si la tâche i doit migrer. Soit l le nombre d'état d'un serveur et $k \in [1, l]$ un état. Soit e_k^j un booléen égal à 1 si le serveur j est dans l'état 1, 0 sinon. Soit $c_c^{j,k}$ la fréquence CPU du serveur j dans l'état k. Soit a_k^j la consommation énergétique du serveur s dans l'état k. Soit b_k^j la consommation énergétique du serveur j dans l'état k quand le serveur est allumé. Soit o_j^i un booléen égal à 1 si la tâche i était exécuter sur le serveur j, 0 sinon. Soit h_j^i égal à 1 si la tâche j doit être exécutée sur le serveur j, 0 sinon. Soit h_j égal à 1 si le serveur j est allumé, 0 sinon

La minimisation de la consommation énergétique des serveurs, du nombre de tâches non assignées, du nombre de migration de tâches peut être formulé par :

$$\min \sum_{j=1}^{m} \sum_{k=1}^{l} e_k^j \left(a_k^j \left(1 - \sum_{i=1}^{n} \frac{d_c^i \, h_j^i}{c_c^{j,k}} \right) + b_k^j \, h_j \right) + \sum_{i=1}^{n} p_i \left(1 - \sum_{j=1}^{m} h_j^i \right) + \sum_{i=1}^{n} \left(m_i \sum_{j=1}^{m} h_j^i \, (1 - o_j^i) \right)$$

Tel que :

$e_k^j \sum_{i=1}^{n} d_c^i \, h_j^i < c_c^{j,k} \quad \forall j,k \in ([1,m],[1,l])$ (1)

$\sum_{i=1}^{n} d_m^i \, h_j^i < c_m^j \quad \forall j \in [1,m]$ (2)

$\sum_{i=1}^{n} d_{io}^i \, h_j^i < c_{io}^j \quad \forall j \in [1,m]$ (3)

$\sum_{j=1}^{m} h_j^i \leq 1 \quad \forall i \in [1,n]$ (4)

$h_j^i + h_j^k \leq 1 \quad \forall i,j,k \in ([1,n],[1,m],[1,n]-\{i\}) \mid r_i = 1$ (5)

$0 \leq h_j^i \leq 1 \quad \forall i,j \in ([1,n],[1,m])$ (6)

$h_j^i \leq h_j \leq 1 \quad \forall i \in [1,n]$ (7)

$\sum_{k=1}^{l} e_k^j = 1 \quad \forall j \in [1,m]$ (8)

(1) sur chaque serveur, les demandes en CPU des tâches en cours d'exécution n'excèdent pas la capacité disponible
(2) sur chaque serveur, les demandes en mémoire des tâches en cours d'exécution n'excèdent pas la capacité disponible
(3) sur chaque serveur, les demandes en I/O des tâches en cours d'exécution n'excèdent pas la capacité disponible
(4) Chaque tâche ne doit être que sur un seul serveur
(5) Les tâches critiques doivent être exécutées seules sur leur serveur
(6) h_j^i est un booléen
(7) h_j est un booléen égal à 1 si le serveur j est allumé, 0 sinon
(8) chaque serveur est dans un seul état
Une solution est donnée par la matrice de booléens $h_j^i \mid i,j \in ([1,n],[1,m])$

$$S = \{h_1^1, \dots, h_j^i, \dots, h_m^n\}$$

3 Conclusions et perspectives

Nous avons résolu ce problème en utilisant le solveur GUROBI pour une approche directe et pour sa linéarisation. Nous avons comparé nos résultats avec ceux obtenus grâce à local solver et à la méthode Kangourou.

4 Références

[1] Hermenier, Fabien. Gestion dynamique des tâches dans les grappes, une approche à base de machines virtuelles. 2010.
[2] VMWare. Principes de base de la virtualisation. http://www.vmware.com/fr/virtualization/virtualization/what-is-virtualization.html.
[3] EcoInfo. Etude de cas "Serveurs de calcul et consommation d'énergie. http://www.eco-info.org/spip.php?article132.