Ordonnancement dans des systèmes distribués
ING2-GSI – Projet GSI-S4
Année 2015–2016

EISTI

Introduction

Le but de ce projet est de créer une application *scheduler* permettant de placer des processus selon certaines stratégies d'ordonnancement.

Modalités

Ce projet est à effectuer par groupe de 3, la composition des groupes est libre. Les choix algorithmiques et d'implémentation sont libres. Le code doit être rédigé en C++, avec OpenMP et MPI. On utilisera IPv6 pour toutes les connexions réseau. Tout le travail doit être effectué en utilisant le GitLab de l'école (codes sources, rapport, fichiers de tests, etc.).

Principe d'utilisation

Le *scheduler* devra fonctionner de la manière suivante : des travaux, placés dans une queue, arriveront au *scheduler*, qui les assignera aux ressources (processeurs ou cores) disponibles dans la machine. Le *scheduler* devra prendre en compte les propriétés du travail (temps d'exécution prévu par l'utilisateur, nombre de *threads*, quantité de mémoire et CPU demandée, etc.) pour faire une assignation optimale selon une stratégie d'ordonnancement choisie par l'utilisateur.

Dans un premier temps le *scheduler* fonctionnera de manière séquentielle (assignation des travaux les uns après les autres). Une version parallèle mono-machine (OpenMP) sera ensuite mis en place pour gérer l'assignation simultanée de plusieurs travaux au même temps. Finalement, une version multi-machine (MPI) sera capable d'assigner des travaux vers plusieurs machines.

Améliorations possibles

– Prendre en considération des contraintes imposées au système (par exemple, un temps d'occupation maximal des ressources par chaque travail assigné), de sorte que si un travail excède le temps initialement prévu, il soit tué.

- Mettre en place d'un système de commutation de contexte qui favorise les travaux avec une priorité plus haute.
- Mettre en place d'un système de monitoring avec des capteurs basés sur des compteurs matériels, de sorte que si un travail arrive à un seuil d'utilisation de CPU ou de mémoire, il soit migré vers un autre processeur.

Cahier des charges

Vous devez établir votre cahier des charges pour la troisième séance de projet (la veille à 23h59 au plus tard et devra également être publié sur Arel). Celui-ci devra contenir une reformulation du sujet faisant clairement apparaître votre compréhension du problème (notamment dans la définition des termes inconnus), la stratégie que vous allez adopter ainsi que le planning prévisionnel d'exécution. À la fin de la deuxième séance on fera le point pour discuter et réorienter (s'il y en a besoin) votre stratégie.

Déroulement des séances

Tout au long des séances, vous devrez rendre compte de votre travail à votre encadrant qui se basera sur ce qui aura été posté sur GitLab. Le rendu final sera le dernier rendu publié sur GitLab, au plus tard à 23h59 le **08/05/2016**. Il sera également publié sur Arel.

Notation

Seront notées la vraisemblance de l'implémentation des algorithmes, la cohérence entre la réalisation et le cahier des charges, les techniques et méthodes d'implémentation, la fiabilité des tests de vérification, la gestion du suivi de projet et, bien entendu, la réponse apportée à l'objectif attendu.

Un rapport devra être fourni avec le rendu et être suffisamment pertinent, contenant notamment votre avis sur l'opportunité des méthodes de parallélisation employées. Il ne sera pas noté mais pourra pénaliser lourdement la note donnée à la réalisation s'il se trouvait ne pas être satisfaisant.

Bibliographie

1. A. Tanenbaum. Systèmes d'exploitation, 3ème Ed. Pearson 2008.
2. C. Hughes, T. Hughes. Professional Multicore Programming. *Design and Implementation for C++ Developers*. Wiley 2008.
3. Scheduler d'OpenNebula : `http://docs.opennebula.org/4.12/administration/references/schg.html`