Laboratoire de l'Informatique du Parallélisme
(UMR CNRS - ENS Lyon - UCB Lyon 1 - INRIA 5668)

équipe ROMA

Proposition de stage – Master 1/2

Ordonnancement de requêtes pour les bases de données répliquées

Loris Marchal, Sonia Ben Moktar & Étienne Rivière.

Mots-clés : Systèmes distribués, ordonnancement, algorithmique.

Duration : 4 à 6 moins, entre janvier et juillet 2019.

Advisors :

Loris Marchal chargé de recherche CNRS, laboratoire, ENS Lyon, France
(http://perso.ens-lyon.fr/loris.marchal/, loris.marchal@ens-lyon.fr)

Sonia Ben Mokhtar : directeur de recherche CNRS, laboratoire LIRIS, Univ. Lyon 1, France
(https://sites.google.com/site/soniabm/, sonia.benmokhtar@insa-lyon.fr)

Étienne Rivière : professeur à l'université catholique de Louvain, Belgique
(https://uclouvain.be/repertoires/etienne.riviere, etienne.riviere@uclouvain.be)

Environnement de recherche

Le stage se déroulera dans l'**équipe ROMA du "Laboratoire de l'Informatique du Parallélisme" de ENS Lyon**, sous la direction de Loris Marchal, dont la recherche s'articule autour de la conception d'ordonnancement pour les plates-formes de calcul haute-performance (HPC). Le stage sera aussi co-dirigé par Sonia Ben Mokhtar (au laboratoire LIRIS, Univ. Lyon 1) et Étienne Rivière (à UC Louvain, Belgique), tout deux spécialistes des systèmes distribués et en particulier des systèmes de stockage distribués. S'il le souhaite, le stagiaire pourra également effectuer un **séjour de recherche d'un mois à UC Louvain** à la fin du stage (des financements spécifiques pourront être recherchés).

Contexte du stage

De nombreux systèmes distribués ont été proposés pour améliorer le stockage de l'information et sa récupération, comme les bases de données utilisant le système clé-valeur telles que Apache Cassandra ou MongoDV. Même sur des systèmes bien dimensionnés, il est toujours difficile de s'assurer que toutes les requêtes peuvent être servies rapidement à cause du déséquilibre de la charge des serveurs. En particulier, les 5% des requêtes les plus lentes peuvent avoir une latence beaucoup plus élevée que la moyenne (problème de *tail latency*). La réplication des données est couramment utilisée pour surmonter ce problème mais requiert un algorithme efficace pour la sélection de réplica. Lorsque les données sont de tailles hétérogènes, les requêtes ordonnancées derrière des requêtes pour de grandes données (et nécessitant un grand temps de traitement) peuvent subir un délai supplémentaire (problème de *head of line blocking*).

Héron [1] est un algorithme de sélection de réplica proposé récemment pour éviter ce problème. Il identifie les requêtes pour de grandes données à l'aide de filtres de Bloom afin d'éviter d'ordonnancer d'autres requêtes derrières elles. Expérimentalement, Héron surpasse les algorithmes existants en réduisant à la fois la latence moyenne et la latence des 5% des requêtes les plus lentes.

Cependant, nous ne savons pas si un tel algorithme de sélection de réplica peut être encore optimisé. Plus précisément, nous voudrions savoir si Héron est loin d'un algorithme de sélection optimal. Un moyen naturel de répondre à cette question est d'abstraire le problème et de modéliser le système distribué en choisissant des caractéristiques clés et en simplifiant d'autres aspects afin d'obtenir un problème d'ordonnancement plus simple à résoudre. Des algorithmes optimaux ou

garantis ont d'ailleurs été proposés pour résoudre des problèmes d'ordonnancement similaires : lorsque les requêtes sont de taille homogène et les données totalement répliquées, on sait que la politique « premier arrivé, premier servi » est optimale pour minimiser à la fois le temps de réponse moyen et maximal [3]. La contrainte de localité particulière induite par la politique de réplication peut être modélisée précisément en utilisant les « disponibilités restreintes », comme proposé dans [2].

Objectif du stage

Le but de ce stage est à la fois de proposer des bornes inférieures précises sur la latence maximale pour le problème de sélection de réplica en utilisant des techniques d'ordonnancement et d'identifier les caractéristiques clés du problème qui permettent de concevoir des strategies efficaces pour la sélection des réplicas.

Nous avons identifiés deux spécificités notables du problème, dont l'utilisation pour concevoir ou analyser des algorithmes d'ordonnancement serait particulièrement innovante :

— Grâce au suivi constant de la plate-forme et à son redimensionnement dynamique, la charge moyenne de chaque serveur est maintenue entre 50% et 80%. Ceci exclut les cas extrêmes de surcharge et de sous-utilisation de la plate-forme.

— La popularité des données peut être apprise pendant l'exécution. Les données les plus populaires sont généralement la cible d'un bien plus grand volume de requêtes, ce qui justifie un ordonnancement spécifique.

Le stage consistera en deux parties, dont l'importance respective pourra être adapté en fonction des compétences de l'étudiant et de ses souhaits :

1. Des études théoriques pour modéliser le problème, explore la bibliographie en ordonnancement, concevoir des algorithmes d'ordonnancement et de sélection de réplicas, montrer leur optimalité ou des garanties sur leur performance.
2. Des études expérimentales utilisant la caractérisation statistique de jeu de données provenant de systèmes de stockage existants, ainsi que des simulations des algorithmes d'ordonnancement et de réplication, pour comprendre les caractéristiques clés du problème.

Compétences requises

L'étudiant devra avoir un bon niveau en algorithmique et en programmation. Des compétences préliminaires en ordonnancement et/ou en systèmes distribués sont les bienvenues.

Bibliographie

[1] Vikas Jaiman, Sonia Ben Mokhtar, Vivien Quéma, Lydia Y. Chen, and Etienne Rivière. Héron : Taming tail latencies in key value stores under heterogeneous workloads. In *Proceedings of the International Symposium on Reliable Distributed Systems (SRDS)*, 2018.

[2] Arnaud Legrand, Alan Su, and Frédéric Vivien. Minimizing the stretch when scheduling flows of divisible requests. *J. Scheduling*, 11(5) :381–404, 2008.

[3] Barbara Simons. Multiprocessor scheduling of unit-time jobs with arbitrary release times and deadlines. *SIAM Journal on Computing*, 12(2) :294–299, 1983.