DHT : Rapport de projet

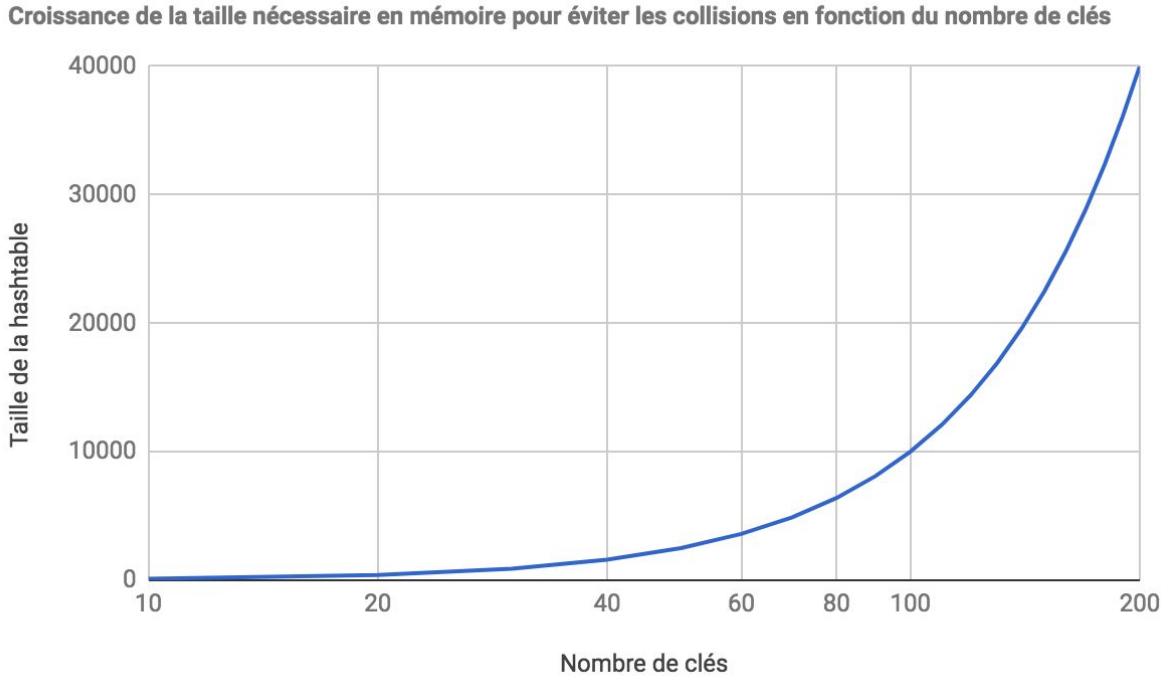
Abstract

Dans ce rapport, nous nous attacherons à montrer les potentialités et les limites de notre projet de C du semestre de printem ps 2018 : les distributed hashtables.

Nous nous attacherons aux performances en terme de vitesse, et fiabilité de l'information, et de taille en mémoire du stockage.

L'intérêt de la hashtable, distribuée ou non, c'est que les opérations put et get sont en O(1). Ceci n'est possible que si la fonction de hash utilisée n'a pas de collisions, sinon le temps d'accès est plus long.

Pour avoir une probabilité de collision faible, le lemme de l'anniversaire (birthday lemma) nous dit que si on appelle m la taille de la hashtable et K le nombre d'élément à placer alors on doit avoir $m>K^2$.



On voit clairement qu'il y a un équilibre performance/mémoire à trouver.

Analyse de performance

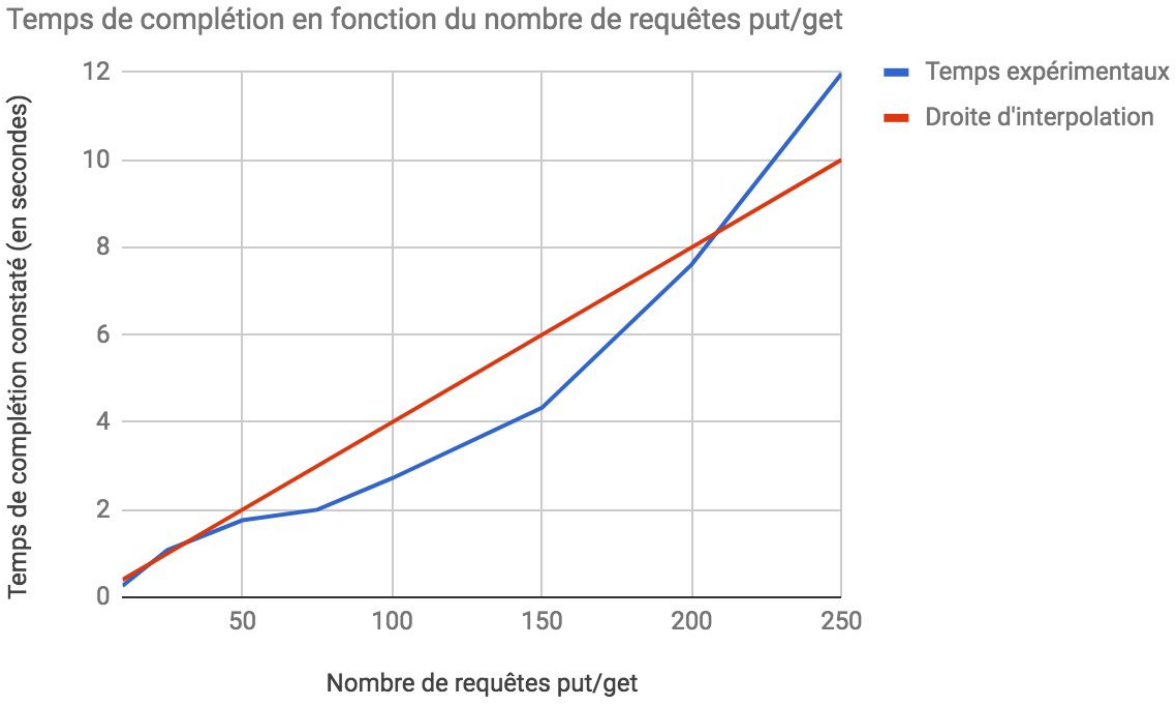
Les expériences réalisées sur le cluster de serveurs sont difficiles à interpréter pour plusieurs raisons :

- Le statuts des serveurs n'était pas stable
- Le nombre de serveurs disponibles à un instant donné pouvait grandement varier
- Certains serveur ne se comportent pas comme attendu et altèrent le résultat de nos expériences.

Scénario de l'expérience :

1. Exécuter une série de put request - clé valeur générique - avec get request associé sur le cluster de serveurs
2. Faire varier leur nombre et comparer ainsi les temps à émettre et recevoir les paires de clé valeur

Le graphique ci-contre projette les données obtenues :



Exploitation des données

Il montre que le temps de complétion est quasi proportionnel au nombre de requêtes envoyés, ici le coefficient de proportionnalité est peu important car il dépend de nombreuses conditions externes, mais l'aspect général du graphe est intéressant, malgré la variabilités des données récoltés suivant l'heure, l'endroit, etc (parfois faisant varier les temps de complétion du quitte au double), l'aspect général du graphe reste celui d'une droite.

Analyse critique du projet

Bien que notre projet de DHT soit fonctionnel, il a encore ses limites et pourrait difficilement fonctionner tel quel dans un environnement de production. Les temps de réponses dans le cluster EPFL ne sont pas assez courts et pourraient être optimisés par une architecture

réseau adaptée. On voit aussi que ce principe de hashtable distribuée pose également un problème d'optimisation assez complexe, qui est de savoir comment choisir les paramètres N, W et R afin d'avoir une réponse fiable, la plus rapide possible, en utilisant le moins de ressources possible. Pour un cluster si petit que le nôtre cela semble être une question négligeable mais pour une compagnie comme Amazon c'est une question qui peut valoir beaucoup d'argent, tout comme la question du trade off performance/stockage inhérente à la hashtable évoquée au début de ce rapport.