Rapport Projet Données Réparties

Thomas BOCANDE, Margaux GARRIC

Introduction

Le but de ce projet est de permettre l'exécution en parallèle de traitements sur un grand volume de données. Pour essayer de mettre en place le système *Hadoop* nous avons mis en place 2 parties importantes.

Le service HDFS

Principe:

Pour commencer le service **HDFS** (*Hadoop Distributed File System*). Ce système va nous permettre de gérer la gestion de fichiers répartis. En effet nous allons utiliser le service pour découper un fichier en fragment et stocker sur des nœuds du cluster.

Ce service va nous permettre de réaliser plusieurs actions.

L'écriture d'un fichier du système de fichiers local (**FS**) sur **HDFS**, la lecture d'un fichier du **HDFS** sur **FS** et la suppression d'un fichier du **HDFS**.

Pour communiquer entre ces systèmes de fichiers nous utilisons des sockets en mode **TCP**.

Pour lire ou écrire des clés-valeurs (`KV`) à partir de fichiers locaux, le système utilise des objets `ReaderWriter`.

Implementation:

Cette implémentation est tout d'abord composée du `HdfsServer` et du `HdfsClient`. `HdfsClient` va nous permettre de manipuler **HDFS** depuis un shell. En fonction de l'action demandée la classe `HdfsClient` va fournir les méthodes `HdfsWrite`, `HdfsDelete` et `HdfsRead`.

Ensuite nous avons créé un classe `Request`, un objet qui sera transmis lors d'une demande du client au serveur et qui contiendra les informations nécessaires à la réalisation de cette requête.

L'implémentation de `FileReaderWriter` nous permettra de réaliser des actions sur des fichiers. Par exemple, lire un fichier texte ou *kv* et renvoyer le *kv* correspondant ou écrire un *kv* dans un fichier.

Mise en œuvre:

Pour tester notre service il a fallu créer un fichier `config.txt`. Ce fichier contient l'identifiant des machines et le port utilisés par les serveurs du **HDFS**. Il sera lu afin de connaître les serveurs disponibles et nous guider pour par exemple la découpe du fichier à écrire ou la localisation de tous les fichiers à supprimer.

Après essais sur un fichier simple de *kv* ou *txt* en local avec une liste de ports les commandes delete et write fonctionnent bien.

Le service Hagidoop

Principe :

Le service Hagidoop fournit le support pour l'exécution répartie du schéma de programmation *MapReduce* qui permet de traiter et d'analyser des données massives de manière distribuée.

Implémentation et mise en œuvre

Il a fallut d'abord implémenter les `Worker` qui sur chaque nœud du cluster lance les `Map` en parallèle et les renvoie au client via un `NetworkReaderWriter` les résultats. Puis nous avons continué avec l'implémentation de `startJob` qui permet au client de lancer chaque `Worker` de manière concurrente avec un `Map` donné (via `runMap` récupéré par **RMI**). Il récupère les différents résultats aussi via `NetworkReaderWriter` et les traites avec le `Reduce`

Conclusion

En résumé ce projet nous a permis de mieux comprendre l'utilisation des *sockets*, de *RMI* et des outils concurrent ainsi qu'une application concrète de ceci.