RAPPORT DU PROJET HIDOOP

1. Partie Hidoop

1.1. L'achitecture

La partie Hidoop se concentre principalement sur les classes *Job*, *DaemonImpl* ainsi que les différents Thread, *MapThread* et *ReduceThread*.

1.1.1. La classe Job

La classe *Job* s'occupe de dialoguer avec *HDFS* afin de gérer les fichiers et de de lancer le mécanisme de MapReduce. Premièrement, il contacte *HDFS* afin de découper en fragments le fichier. Puis, *Job* s'occupe de récupérer les différents Daemon associés à ces fichiers par RMI. Il exécute sur chaque Daemon un *runMap*. A l'issu des exécutions, il demande à chaque *Daemon* de faire un *reduce* sur les différents fichiers qu'il possède (il peut y en avoir qu'un seul) ensuite il appelle *HDFS* afin qu'il rassemble tous les fichiers obtenus en un seul. Puis il exécute un *reduce* sur ce fichier final.

1.1.2. La classe *DaemonImpl*

Le *DaemonImpl* est une petite application qui tourne en arrière plan. Il s'enregistre auprès d'un RMI et attend des instructions d'un *Job*. Il permet d'exécuter un *map* et un *reduce* sur un fichier local que *HDFS* lui aura envoyé.

1.1.3. Les classes Thread

Les deux classes *MapThread* et *ReduceThread* s'occupent respectivement d'exécuter un map sur un fichier et d'exécuter une reduce sur un fichier. Elles permettent de faire un *Daemon* multi-threads.

1.2. Futur du Hidoop

Pour l'instant la liste des *Daemon* est connue, dans le futur nous aimerions implanté un système d'enregistrement pour chaque *Daemon* afin de ne pas avoir un enregistrement statique. Pour l'instant chaque *Daemon* ne gère qu'un seul *Chunk* mais nous aimerions aussi ajouté la fonctionnalité de gérer plusieurs *Chunk* par *Daemon* (toujours en multi-thread) afin de mieux optimiser les ressources disponibles.

2. Partie HDFS

2.1. L'achitecture

La partie HDFS se concentre principalement sur les classes *HdfsClient*, *DataNodeMain*, NameNodeMain.

2.1.1. La classe HdfsClient

Cette classe est l'interface entre l'utilisateur et HDFS, trois commandes sont disponibles: write, read et delete.

Write permet d'ajouter un fichier dans HDFS, read de lire un fichier d'HDFS et delete de supprimer un fichier de HDFS.

2.1.2. La classe *DataNodeMain*

Cette classe crée un server qui envoie ses informations (IP, port) au Namenode afin que le HdfsClient puisse venir communiquer et transférer des fichiers. C'est sur cette machine que sont stockés les fragments d'un ficher de HDFS.

2.1.3. La classe NameNodeMain

Cette classe est aussi un server qui va stocker les informations des datanodes ainsi que des fichiers et leurs fragments dans HDFS.

2.2. Futur de HDFS

Certaines classes intermédiaires possèdent des attributs non nécessaires, il faudrait les enlever. On peut aussi rajouter des serveurs RMI pour faciliter la communications entre les différents serveurs. Il faudrait également améliorer l'implémentation de Hidoop dans HDFS.