Evaluation de la partie Hidoop

Travail mené par Adrien Gallego et Duc-Do Trinh.
Evalué par Matthieu Hamel, Thibaud Merieux et Ignacio Oros Campo.

Partie technique

Le travail effectué dans le dossier « ordo » permet de réaliser des map sur chacun des fragments créés par la partie HDFS, pour ensuite réaliser un reduce sur le fichier obtenu. « Job » réalise bien un appel à « HdfsClient » qui lui-même appelle « HdfsRead ».
L'utilisation de commentaires est bien réalisée et pertinente car elle facilite bien la compréhension du code.

Synthèse

- **Correction :**

Pour le moment, il n'est pas possible d'effectuer des tests car les parties HDFS et Hidoop n'ont pas encore été assemblées. Il est prévu de réaliser une partie HDFS simplifiée permettant d'effectuer les tests de la partie Hidoop, avant de mettre en commun les 2 parties.

- **Complétude :**

La spécification attendue a bien été abordée, « Job » avec la méthode « StartJob » lance bien un Worker sur chaque machine, qui vont jouer le rôle de démons. Un sémaphore a été utilisé pour attendre la terminaison de tous les Workers, par le biais d'un CallBack envoyé par chacun des Workers à la fin de leur tâche. On notera que des suffixes sont bien ajoutés aux fichiers writer afin de les différencier.
Lorsque tous les Workers ont fini de produire leurs résultats, on a un appel à HDFSClient qui va produire le fichier des résultats agrégés par le biais de notre HDFSRead afin d'être traité par le reduce final.
La fonction setInputFname a bien été réalisée pour pouvoir être utilisée par Count.java.

- **Pertinence :**

Le travail réalisé permet (du moins en suivant manuellement le code) de réaliser les fonctions attendues. On a bien le choix du inputFormat (KV ou Line) qui sera pris en compte pour le lancement du runMap côté Worker. De plus WorkerImpl utilise bien les threads afin d'avoir une parallélisation. Pour les communications entre Job et Callback, l'utilisation des sémaphores ne fait pas d'attente active.

- **Cohérence :**

La structure réalisée permet bien de faire ce qui est demandé : on a d'abord la "répartition" du travail en lançant les Maps sur chaque Worker ensuite on va récupérer les résultats afin de faire le reduce.
On n'a pas particulièrement trouvé de fonctions qui se recoupent ou font doublon.

- **Points d'amélioration :**

1) L'appel à la fonction main de HDFSClient n'est pas très propre lorsqu'on agrège les résultats. Il serait souhaitable de directement appeler la fonction HdfsRead qui fera ce qui est recherché.

2) L'utilisation des sémaphores dans Job pour l'attente de la terminaison de tous les Workers fonctionne mais paraît un peu compliqué et pourrait être simplifiée. En effet on va boucler inutilement (NbDeMachines – 1) fois. Dans le cadre du projet on ne travaille pas avec beaucoup de machines mais si on veut rendre l'application plus générale on pourrait plutôt utiliser des moniteurs si le temps le permet.