TP Hadoop en Python

Yves Denneulin, Vincent Leroy

2020

1 Introduction

Vous allez dans ce TP apprendre à écrire un programme Hadoop en Python. L'avantage de développer en Python par rapport à Java est un prototypage plus rapide, l'inconvénient est un contrôle moins fin des paramètres de la plate-forme et donc des performances obtenus.

2 Premier contact : wordcount

Pour démarrer vous allez écrire un programme permettant de compter les mots d'un fichier, comme nous l'avons vu en cours. Vous écrirez deux programmes Python, un pour la phase *Map* et un autre pour la phase *Reduce*. Écrivez et testez les d'abord en local sur votre machine avant de les exécuter sur le cluster Hadoop. Pour cela vous n'avez pas besoin d'installer autre chose que Python. Vous devez vous assurer que la sortie du mapper est une suite de lignes contenant un couple (clé, valeur), le séparateur entre les deux étant le caractère de tabulation.

2.1 Écriture du mapper

Dans cette partie vous devez écrire un programme Python `mapper.py` qui prend en entrée un fichier texte et affiche en sortie chaque mot du fichier avec la valeur 1.

Par exemple pour le fichier en entrée :

```
a b a c
```

le mapper doit afficher en sortie :

```
a 1
b 1
a 1
c 1
```

2.2 Écriture du reducer

Vous devez maintenant écrire le reducer, le programme Python `reducer.py` qui va prendre en entrée la sortie du mapper et afficher en sortie chaque mot avec son nombre d'occurences. Pour le fichier exemple précédent il affichera :

```
a 2
b 1
c 1
```

Important : Pour simplifier l'écriture du reducer, vous pouvez supposer que les couples sortis par le mapper sont triés avant d'arriver au reducer. Ce traitement est fait par la plateforme Hadoop entre les deux phases.

2.3 Tests en local

Pour tester en local votre mapper et votre reducer sur le contenu du fichier appelé `data` vous pouvez taper la commande suivante :

```
./mapper.py < data | sort | ./reducer.py
```

3 Utilisation du cluster Hadoop

Lorsque votre code Python tournera en local, vous pouvez l'essayer sur le cluster qui a été déployé pour les besoins du TP. Vous vous connectez dessus en utilisant le même login qu'à l'Ensimag avec le mot de passe fourni pendant la séance. Le serveur principal (NameNode) est accessible à l'adresse 152.77.81.30

3.1 HDFS

Exécutez les commandes suivantes :

1. Connectez-vous par ssh sur le noeud principal par la commande *ssh votrelogin*@152.77.81.30[1] [2]
2. exécutez $hdfs\ dfs$ pour voir les commandes disponibles sur le système de fichier HDFS
3. exécutez $hdfs\ dfs\ -ls\ /$ pour voir ce qui est à la racine du système de fichier du cluster
4. exécutez $hdfs\ dfs\ -ls$ pour voir ce qui est à la racine de votre répertoire sur le cluster

[1] Vous devez être sur le réseau de l'Ensimag soit physiquement soit connecté par le VPN.

[2] Pensez à changer votre mot de passe et à mettre votre clé publique ssh pour pouvoir vous connecter plus facilement ensuite

3.2 Exécution sur le cluster

Le programme que vous avez écrit dans la partie 2 peut être exécuté sur le cluster. Vous devez pour cela le transférer de la machine sur laquelle vous l'avez développée vers le cluster en utilisant les commandes $sftp$ ou *scp*.

Pour exécuter votre programme map/reduce avec Hadoop sur l'ensemble du texte des Misérables vous devez taper la commande suivante (sur une seule ligne) à partir du répertoire où se trouvent vos fichiers mapper.py et reducer.py :

```
hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.1.2.jar
-files mapper.py,reducer.py -mapper mapper.py  -reducer reducer.py
-input /data/miserables -output wc
```

Question Regardez attentivement le déroulement de l'exécution ainsi que l'affichage du comportement de Hadoop. Retrouvez le nombre de "splits" lus sur HDFS. A quel compteur ce nombre correspond-il ?

En préparation de la question suivante, conservez les traces d'exécution de ce programme dans un fichier temporaire, et veillez à conserver le répertoire qui contient les résultats.

Où sont les résultats du job Hadoop ? Pour les récupérer sur votre compte sur le cluster vous pouvez utiliser la commande $hdfs\ dfs\ -get$[3]

Vous pouvez aussi essayer votre programme sur d'autres sources de données. Pour cela, vous devez d'abord les transférer vers le cluster puis ensuite sur le système de fichiers Hadoop en utilisant par exemple la commande HDFS $hdfs\ dfs\ -put\ <fichier>$.

3.3 Ajout d'un combiner

Question À partir du cours, déduisez ce que doit faire le combiner pour le programme wordcount. Déduisez-en le code du programme Python *combiner.py*

La (longue) commande suivante vous permet d'utiliser un combiner :

```
hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.1.2.jar
-files mapper.py,reducer.py,combiner.py -mapper mapper.py -combiner combiner.py
-reducer reducer.py -input /data/miserables -output wc
```

Question Le résultat est il le même que sans combiner ? Sur quels compteurs voyez-vous l'effet du combiner ?

3.4 Plusieurs reducers

Dans toutes les exécutions menées jusqu'à présent, le reduce n'était exécuté que sur un nœud. Nous allons changer cela en exécutant le programme avec la commande suivante :

[3] regardez la documentation disponible en cas de besoin.

```
hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.1.2.jar
-files mapper.py,reducer.py -mapper mapper.py -reducer reducer.py
-numReduceTasks 3 -input /data/miserables -output wc
```

Question Comment se présente le résultat ? Pourquoi ?

4 Top-tags Flickr par pays

Dans la suite de ce TP vous allez étudier des méta-données de photos stockées dans un jeu de données de Flickr. À chaque photo est associé par l'utilisateur un ensemble de tags ainsi que les coordonnées GPS où la photo a été prise. Le but sera de trouver à partir de ces données pour chaque pays quels sont les tags les plus fréquents.

Dans l'archive que vous avez téléchargée se trouvent trois fichiers :

- *flickrSpecs.txt* contient le format du fichier de données.
- *flickrSample.txt* contient un extrait du fichier de données que vous pouvez utiliser pour vos test locaux.
- *country.py* qui contient le code de la fonction *getCountryAt* qui prend en paramètres une latitude et une longitude et renvoie le code du pays de cette coordonnée. N'oubliez pas d'appeler la fonction *init_pays* avant de l'appeler la première fois.

4.1 Map et Reduce

On veut trouver les K tags les plus utilisés par pays, une bonne clé intermédiaire entre le mapper et le reducer semble donc être le code du pays sur 2 caractères retourné par la fonction *getCountryAt* fournie.

Dans le reducer il faudra utiliser une structure de données stockant l'ensemble des tags et leur nombre d'occurence pour ensuite en extraire les plus fréquents.

Écrivez le programme map/reduce réalisant ce traitement. Testez-le localement puis sur le cluster.

4.2 Combiner

Pouvez-vous utiliser directement le reducer de la question précédente comme combiner ? Pourquoi ?

Réfléchissez à l'entrée et à la sortie du combiner. Écrivez le combiner et modifiez le reducer de la question précédente pour que le résultat soit correct.

Question : la structure de données utilisée dans le reducer contient l'ensemble des tags de l'ensemble des pays. Cela peut-il poser un problème pour traiter de grands volumes de données ?

4.3 Version en 2 passes

Proposez une version en 2 passes permettant de limiter l'utilisation de la mémoire sur le reducer.