

Plateforme Hidoop

Projet de données réparties

2020 - 2021

LAPLAGNE Chloé
RAZAFIMANANTSOA Nathan
GUILLAUD Thomas
GRUNIG Axel

Ce document présente le fonctionnement global de la plateforme.

Le répertoire *hidoop* correspond à l'arborescence de fichiers suivante :

- le répertoire **config** contient les fichiers d'initialisation pouvant être utiles lors du lancement de la plateforme
- le répertoire **data** accueille les fichiers de données de l'application
- le répertoire **doc** accueille les rapports attendus
- le répertoire **src** contient les codes sources

Configuration de Hidoop

La variable système **HIDOOOP_HOME** est nécessaire au fonctionnement de Hidoop. Il s'agit de la localisation du répertoire *hidoop* local.

La configuration se fait via un fichier *conf.xml* placé dans le répertoire **\$HIDOOOP_HOME/config/**. Un fichier d'exemple est donné ci-dessous :

```
<?xml version="1.0" encoding="UTF-8"?>
<config metadata="meta">
  <default-chunk-size value="64" unit="bytes" />
  <servers>
    <node ip="172.211.22.1"/>
    <node ip="chewie"/>
  </servers>
</config>
```

- **config**: l'attribut obligatoire *metadata* est le nom du fichier de métadonnées qui sera créé.
 - **default-chunk-size**: élément optionnel précisant la taille de chunk à utiliser par défaut dans HDFS. Si absent, cette taille est de 64MB.
Note : les unités de tailles supportées sont bytes, kB, MB, GB (non sensible à la casse). Par simplicité, une unité inconnue a le même effet que bytes.
 - **servers** : liste des serveurs. L'attribut *ip* d'un *node* correspond en réalité soit à l'adresse ip de la machine soit à son nom (hostname). Cette liste ne peut pas être vide.

Ce fichier est utilisé par la classe **config.AppData** permettant de charger la configuration de l'application.

Compilation et déploiement

Pour compiler les sources sur une machine, les commandes sont les suivantes :

```
$ export HIDOOOP_HOME=/path/to/hidoop
$ cd $HIDOOOP_HOME/src
$ javac application/MyMapReduce.java application/Count.java
  ordo/WorkerImpl.java hdfs/HdfsClient.java hdfs/HdfsServer.java
```

Note :

- Pour compiler seulement la partie serveur, remplacer la dernière commande par :
`$ javac ordo/WorkerImpl.java hdfs/HdfsServer.java`
- Pour compiler seulement la partie client, remplacer la dernière commande par :
`$ javac application/MyMapReduce.java hdfs/HdfsClient.java`

Pour spécifier un répertoire particulier pour les classes :

```
$ export HADOOP_HOME=/path/to/hadoop
$ export HADOOP_CLASSES=/path/to/class/files
$ cd $HADOOP_HOME/src
$ javac -d $HADOOP_CLASSES application/MyMapReduce.java
application/Count.java ordo/WorkerImpl.java hdfs/HdfsClient.java
hdfs/HdfsServer.java
```

Pour déployer la plateforme sur les serveurs distants, il suffit d'envoyer les fichiers du dossier **src** (avec scp par exemple) sur la machine distante, puis d'y exécuter les commandes données précédemment.

- Pour un déploiement rapide sur les machines de l'N7 depuis une machine personnelle utiliser **deployN7.sh** (nécessite d'avoir ajouté **HADOOP_HOME** dans le *.bash_profile* de la machine distante).

Script de lancement

Le script **hadoop.sh** ouvre un shell permettant d'interagir avec la plateforme. Il permet notamment de lancer et d'arrêter automatiquement les serveurs indiqués dans le fichier de configuration *conf.xml* via ssh (machines N7).

Les fichiers nécessaires sur la machine cliente sont donc ceux des répertoires **src** et **config**.

Le répertoire où les commandes sont exécutées est **\$HADOOP_HOME/src** par défaut, mais il est possible de définir une variable système **HADOOP_CLASSES** qui indique la localisation des classes java de l'application.

Il s'agit d'un shell bash supportant les commandes spécifiques suivante :

- **start** pour lancer les serveurs
- **stop** pour arrêter les serveurs
- **hdfs** raccourci pour java hdfs.HdfsClient, suivi des mêmes arguments
- **mmr** raccourci pour java application.MyMapReduce, suivi des mêmes arguments. Hadoop utilisant RMI, il peut être nécessaire de spécifier l'adresse IP à utiliser si la machine possède plusieurs interfaces (valeur de `java.rmi.server.hostname`). Pour cela, lancer **mmr -ip <ip_address>** suivi des arguments.

Sur chaque serveur, les sorties sont redirigées vers le fichier *<ip_serveur>.log* dans **\$HADOOP_HOME**.

Commande : `$./hadoop.sh <ssh_username>`

Utilisation

La plateforme Hidoop supporte les opérations suivantes :

- Écriture d'un fichier dans HDFS :

```
hidoop> hdfs -w <nom_fichier_local> options
```

Le fichier écrit dans HDFS aura le nom du fichier local. Son chemin peut être absolu (commençant par '/') ou relatif à **\$HIDOOOP_HOME/data/**.

Options :

- **-f ln|kv** : format du fichier (ln par défaut)
- **--chunks-size=<taille>** : taille des chunks (ex: 100B, 1MB, 1.2MB...). Si aucune unité (B, kB, MB, GB, TB) n'est fournie, la valeur est en bytes. Si la valeur est négative, cet argument est ignoré.
- **--rep=<facteur>** : facteur de réplication (entier positif), non supporté et toujours égal à 1 dans cette version

- Lecture d'un fichier dans HDFS

```
hidoop> hdfs -r <nom_fichier> options
```

Options :

- **<fichierLocal>** : fichier local de destination. Son chemin peut être absolu (commençant par '/') ou relatif à **\$HIDOOOP_HOME/data/**. Le fichier est lu dans '**r_<nom_fichier>**' si ce paramètre n'est pas spécifié.

- Liste des fichiers dans HDFS

```
hidoop> hdfs -l options
```

Options :

- **--detail** : informations détaillées sur les chunks


- Suppression d'un fichier de HDFS

```
hidoop> hdfs -d <nom_fichier>
```

- Exécution d'une application en Map/Reduce

```
hidoop> mmr <nom_fichier>
```

Le fichier créé contenant les résultats finaux est **<nom_fichier>-tot**

 **Attention** : Les noms de fichiers HDFS sont limités à 80 caractères et ne doivent pas contenir d'espace.