



Table des matières

1) Démarrage du cluster Hadoop.....	1
2) Premier test de Map/Reduce.....	4
3) Hadoop Streaming (Map/Reduce sous python).....	6
4) Apache Spark.....	7
5) Traitement de données issues d'un routeur.....	10
6) Apache Kafka.....	10

1) Démarrage du cluster Hadoop

a) Je récupère l'image Hadoop sur le registre de l'IUT :

```
root@debian:~# docker pull registry.iutbeziers.fr/cb_hadoop:3.2.1
3.2.1: Pulling from cb_hadoop
ecb349c71af9: Pull complete
6ec1610bf9a1: Pull complete
0ee56b3ee87a: Pull complete
450478d68393: Pull complete
37aa56283f2c: Pull complete
8edb1addacf1: Pull complete
3f6f50d62aba: Pull complete
Digest:
sha256:120ba8339875ce91d522e99aec660ef554d11444bb2c31359a5776bd646
19846
Status: Downloaded newer image for
registry.iutbeziers.fr/cb_hadoop:3.2.1
registry.iutbeziers.fr/cb_hadoop:3.2.1
```

b) Je crée un pont réseau docker pour le cluster hadoop :

```
root@debian:~# docker network create --driver=bridge hadoop
0ca2d47869a59eb1df9d9b5eddd70a12b0a48a7c63776ceebdf071dea62456ce
```

c) Je récupère le fichier .zip sur moodle et je l'extrait. Ensuite je le scp vers ma machine virtuelle

Maintenant je lance les containers docker hadoop à l'aide du script bash :

```
root@debian:~/COLD2-TP-01-base# sh start-hadoop-cluster-docker.sh
Start hadoop-master container...
7a3edf7bb79e238fe9a3d739a38607696664e3ad89309b8695867c46ad4ba2ab
Start hadoop-slave1 container...
8adbba8fca262567229c45f50f97f1e51fac9be7c0a29af2d11572a8404c2e13
Start hadoop-slave2 container...
d1dffb319f4de755aa6aa676ca064284529938675e4694f5288ae784ec8d5adc
root@hadoop-master:~# start-all.sh
```

d) Je lance le script de démarrage du cluster hadoop :

```
root@hadoop-master:~# start-all.sh
Starting namenodes on [hadoop-master]
hadoop-master: Warning: Permanently added 'hadoop-
master,172.18.0.2' (ECDSA) to the list of known hosts.
Starting datanodes
hadoop-slave2: Warning: Permanently added 'hadoop-
slave2,172.18.0.4' (ECDSA) to the list of known hosts.
hadoop-slave1: Warning: Permanently added 'hadoop-
slave1,172.18.0.3' (ECDSA) to the list of known hosts.
Starting secondary namenodes [hadoop-master]
Starting resourcemanager
Starting nodemanagers
```

Une fois dans le hadoop-master il ne faut jamais en ressortir sinon il faudra recommencer les manipulations des question c, d et i

e) J'affiche la liste des processeurs JAVA en fonctionnement :

```
root@hadoop-master:~# jps
1027 Jps
713 ResourceManager
234 NameNode
477 SecondaryNameNode
```

f) J'affiche l'état du système HDFS et je liste le nom des deux machines esclaves du cluster :

```
root@hadoop-master:~# hdfs dfsadmin -report
Configured Capacity: 57229352960 (53.30 GB)
Present Capacity: 34981765120 (32.58 GB)
DFS Remaining: 34981715968 (32.58 GB)
DFS Used: 49152 (48 KB)
DFS Used%: 0.00%
```

```
Replicated Blocks:
  Under replicated blocks: 0
  Blocks with corrupt replicas: 0
  Missing blocks: 0
  Missing blocks (with replication factor 1): 0
  Low redundancy blocks with highest priority to recover: 0
  Pending deletion blocks: 0
Erasure Coded Block Groups:
  Low redundancy block groups: 0
  Block groups with corrupt internal blocks: 0
  Missing block groups: 0
  Low redundancy blocks with highest priority to recover: 0
  Pending deletion blocks: 0
-----
Live datanodes (2):
Name: 172.18.0.3:9866 (hadoop-slave1.hadoop)
Hostname: hadoop-slave1
...
Name: 172.18.0.4:9866 (hadoop-slave2.hadoop)
Hostname: hadoop-slave2
...
```

g) Je le compare aux informations disponibles à l'URL de ma VM : <http://192.168.1.20:9870>

On peut voir beaucoup plus d'informations sur le site comme la sécurité active ou non, l'endroit du stockage des nodes...

h)

i) Je crée le répertoire basique pour l'utilisateur root :

```
root@hadoop-master:~# hadoop fs -mkdir -p /user/root
```



j) Je teste différentes options de la commande `hadoop fs` :


```
root@hadoop-master:~# hadoop fs -ls
root@hadoop-master:~# hadoop fs -mkdir test
root@hadoop-master:~# hadoop fs -ls
Found 1 items
drwxr-xr-x - root supergroup          0 2021-01-14 09:03 test
root@hadoop-master:~# hadoop fs -mv test test2
root@hadoop-master:~# hadoop fs -ls
Found 1 items
```

```
drwxr-xr-x - root supergroup 0 2021-01-14 09:03 test2
```

k) Je test les différents menus de la page WEB :

Browse Directory



Show entries

Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Jan 14 10:03	0	0 B	tonpere

Showing 1 to 1 of 1 entries

2) Premier test de Map/Reduce

a) Je crée le répertoire data_in sur HDFS :

```
root@hadoop-master:~# hadoop fs -mkdir data_in
root@hadoop-master:~# hadoop fs -ls
Found 1 items
drwxr-xr-x - root supergroup 0 2021-01-14 09:47 data_in
```

Maintenant je copie le dossier files.tar.gz dans le container docker :

```
root@debian:~/COLD2-TP-01-base# docker cp files.tar.gz hadoop-master:/root
```

Je peux maintenant voir que le dossier est dans mon container et je peux l'extraire puis je peux envoyer les fichiers .txt dans le dossier data_in

```
root@hadoop-master:~# ls
files.tar.gz  hdfs
root@hadoop-master:~# tar xvf files.tar.gz
file1.txt
file2.txt
file3.txt
```

```

file4.txt
root@hadoop-master:~# hadoop fs -put file1.txt data_in
root@hadoop-master:~# hadoop fs -ls data_in
Found 1 items
-rw-r--r--    2 root supergroup          269 2021-01-14 09:49
data_in/file1.txt
root@hadoop-master:~# hadoop fs -put file2.txt data_in
root@hadoop-master:~# hadoop fs -put file3.txt data_in
root@hadoop-master:~# hadoop fs -put file4.txt data_in

```

b) Je lance les commandes suivantes :

```

root@hadoop-master:~#
JAR=$HADOOP_HOME/share/hadoop/mapreduce/sources/hadoop-mapreduce-
examples-3.2.0-sources.jar
root@hadoop-master:~# hadoop jar $JAR
org.apache.hadoop.examples.WordCount data_in data_out

```

c) J'affiche le contenu du répertoire data_out :

```

root@hadoop-master:~# hadoop fs -ls data_out
Found 2 items
-rw-r--r--    2 root supergroup           0 2021-01-14 10:06
data_out/_SUCCESS
-rw-r--r--    2 root supergroup        3272 2021-01-14 10:06
data_out/part-r-00000

```

d) Je vais sur le site web <http://192.168.1.20:8088> et je regarde les détails de l'application Map/Reduce qui vient d'être soumise au cluster :

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running
1	0	0	1	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes
2	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocated
Capacity Scheduler	[memory-mb (unit=Mb), vcores]	<memory:1024, vCores:1>

Show 20 entries									
ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	
application_1610617566281_0001	root	word count	MAPREDUCE	default	0	Thu Jan 14 11:05:46 +0100 2021	Thu Jan 14 11:05:47 +0100 2021	Thu Jan 14 11:06:17 +0100 2021	

Showing 1 to 1 of 1 entries

3) Hadoop Streaming (Map/Reduce sous python)

a) Je test le programme python en local :

```
root@debian:~/COLD2-TP-01-base# head -50 file1.txt | ./mapper.py |  
sort | ./reducer.py  
Aimant      1  
ça. 1  
CAREME      1  
cela,       1  
...
```

b) Je peux maintenant e lancer dans le cluster hadoop :

```
root@hadoop-master:~#  
JAR=$HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-3.2.0.jar  
root@hadoop-master:~# hadoop jar $JAR -files mapper.py,reducer.py  
-mapper mapper.py -reducer reducer.py -input data_in -output  
data_out2
```

Je peux regarder le contenu du fichier part-00000 et je vois bien qu'il a compté les mots dans le fichier :

```
root@hadoop-master:~# hadoop fs -tail data_out2/part-00000  
phares      1  
physique    1  
pied 1  
places      1
```

4) Apache Spark

a) Je lance le pyspark sur mon hadoop-master :

```
root@hadoop-master:~# pyspark  
Python 2.7.13 (default, Sep 26 2018, 18:42:22)  
[GCC 6.3.0 20170516] on linux2  
Type "help", "copyright", "credits" or "license" for more  
information.  
Setting default log level to "WARN".  
To adjust logging level use sc.setLogLevel(newLevel). For SparkR,  
use setLogLevel(newLevel).  
Welcome to
```



```
Using Python version 2.7.13 (default, Sep 26 2018 18:42:22)
SparkSession available as 'spark'.
>>>
```

b) J'affiche le contenu de la variable sc :

```
>>> print(sc)
<SparkContext master=local[*] appName=PySparkShell>
```

c) Je crée un premier RDD contenant tous les mots de toutes les lignes :

```
>>> lines=sc.textFile('file:///root/file1.txt')
>>> print(lines)
file:///root/file1.txt MapPartitionsRDD[1] at textFile at
NativeMethodAccessorImpl.java:0
>>> lines.collect()
[u'Mon petit chat', u'', u'J'ai un petit chat,', u'Petit comme \
xe7a.', u'Je l'appelle Orange.', u'', u'Je ne sais pourquoi',
u'Jamais il ne mange', u'Ni souris ni rat.', u'', u'C'est un
chat \xe9trange", u'Aimant le nougat', u'Et le chocolat.', u'',
u'Mais c'est pour cela,', u'Dit tante Solange,', u'Qu'il ne
grandit pas!', u'', u'Maurice CAREME']
```

d) Je découpe les lignes pour obtenir les mots de toutes les lignes :

```
>>> words=lines.flatMap(lambda x: x.split())
>>> words.collect()
[u'Mon', u'petit', u'chat', u'J'ai', u'un', u'petit', u'chat,',
u'Petit', u'comme', u'\xe7a.', u'Je', u'l'appelle", u'Orange.',
u'Je', u'ne', u'sais', u'pourquoi', u'Jamais', u'il', u'ne',
u'mange', u'Ni', u'souris', u'ni', u'rat.', u"C'est", u'un',
u'chat', u'\xe9trange', u'Aimant', u'le', u'nougat', u'Et', u'le',
u'chocolat.', u'Mais', u"c'est", u'pour', u'cela,', u'Dit',
u'tante', u'Solange,', u'Qu'il", u'ne', u'grandit', u'pas!',
u'Maurice', u'CAREME']
```

e) Je crée un RDD avec des couples :

```
>>> tuples=words.map(lambda x: (x, 1))
>>> tuples.collect()
```

```
[('Mon', 1), ('petit', 1), ('chat', 1), ('J'ai', 1), ('un', 1), ('petit', 1), ('chat', 1), ('Petit', 1), ('comme', 1), ('\xe7a.', 1), ('Je', 1), ('l'appelle', 1), ('Orange.', 1), ('Je', 1), ('ne', 1), ('sais', 1), ('pourquoi', 1), ('Jamais', 1), ('il', 1), ('ne', 1), ('mange', 1), ('Ni', 1), ('souris', 1), ('ni', 1), ('rat.', 1), ('C'est', 1), ('un', 1), ('chat', 1), ('\xe9trange', 1), ('Aimant', 1), ('le', 1), ('nougat', 1), ('Et', 1), ('le', 1), ('chocolat.', 1), ('Mais', 1), ('c'est', 1), ('pour', 1), ('cela,', 1), ('Dit', 1), ('tante', 1), ('Solange,', 1), ('Qu'il', 1), ('ne', 1), ('grandit', 1), ('pas!', 1), ('Maurice', 1), ('CAREME', 1)]
```

f) Je réalise l'opération de réduction :

```
>>> res=tuples.reduceByKey(lambda a,b: a+b)
>>> res.collect()
[('Ni', 1), ('chocolat.', 1), ('Mais', 1), ('petit', 2), ('C'est', 1), ('Orange.', 1), ('comme', 1), ('souris', 1), ('pourquoi', 1), ('il', 1), ('chat', 2), ('CAREME', 1), ('chat,', 1), ('ni', 1), ('le', 2), ('c'est', 1), ('ne', 3), ('J'ai', 1), ('sais', 1), ('Mon', 1), ('Je', 2), ('nougat', 1), ('Solange,', 1), ('Aimant', 1), ('cela,', 1), ('Qu'il', 1), ('pas!', 1), ('grandit', 1), ('mange', 1), ('Dit', 1), ('\xe9trange', 1), ('l'appelle', 1), ('tante', 1), ('\xe7a.', 1), ('Petit', 1), ('Maurice', 1), ('pour', 1), ('un', 2), ('Et', 1), ('rat.', 1), ('Jamais', 1)]
```

g) Je trie les valeurs dans l'ordre décroissant et je ne garde que les 5 premiers caractères :

```
>>> mostUsed=sc.parallelize([res.takeOrdered(5, key = lambda x: -x[1])])
>>> mostUsed.collect()
[('ne', 3), ('petit', 2), ('chat', 2), ('le', 2), ('Je', 2)]
```

h) Je sauvegarde le résultat en local :

```
>>> mostUsed.saveAsTextFile('file:///root/data_out3')
```

i) Je récapitule toutes les commandes dans un script en créant un contexte au début :

```
#!/usr/bin/env python # -*- coding: utf-8 -*- from pyspark import
SparkContext sc=SparkContext(appName="WordStats") print(sc)
lines=sc.textFile('file:///root/file1.txt') print(lines) lines.collect()
words=lines.flatMap(lambda x: x.split()) words.collect()
tuples=words.map(lambda x: (x, 1)) tuples.collect()
```



```
res=tuples.reduceByKey(lambda a,b: a+b) res.collect()
mostUsed=sc.parallelize([res.takeOrdered(5, key = lambda x: -x[1])])
mostUsed.collect() mostUsed.saveAsTextFile('file:///root/data_out3')
```

j) Je lance le script avec spark-submit :

```
root@hadoop-master:~# spark-submit wordStats.py
2021-01-14 13:14:46 INFO SparkContext:54 - Running Spark version 2.4.0
2021-01-14 13:14:46 INFO SparkContext:54 - Submitted application: WordStats
2021-01-14 13:14:46 INFO SecurityManager:54 - Changing view acls to: root
2021-01-14 13:14:46 INFO SecurityManager:54 - Changing modify acls to: root
...
```

k) Pour réaliser la même chose sur des fichiers du cluster hdfs il suffit de modifier les URL dans le script :

```
#!/usr/bin/env python # -*- coding: utf-8 -*- from pyspark import
SparkContext sc=SparkContext(appName="WordStats") print(sc)
lines=sc.textFile('hdfs:///user/root/file1.txt') print(lines)
lines.collect() words=lines.flatMap(lambda x: x.split()) words.collect()
tuples=words.map(lambda x: (x, 1)) tuples.collect()
res=tuples.reduceByKey(lambda a,b: a+b) res.collect()
mostUsed=sc.parallelize([res.takeOrdered(5, key = lambda x: -x[1])])
mostUsed.collect() mostUsed.saveAsTextFile('hdfs:///user/root/data_out3')
```

Maintenant je dis au serveur qu'il faut utiliser YARN :

```
root@hadoop-master:~# spark-submit --master yarn wordStatsHdfs.py
```

l) Je retourne sur la page web en 8088 et je regarde le détail des applications soumises au cluster :

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Completed
3	0	0	3	0	0 B

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes
2	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation
Capacity Scheduler	[memory-mb (unit=M), vcores]	<memory:1024, vCores:1>

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime
application_1610617566281_0003	root	WordStats	SPARK	default	0	Thu Jan 14 14:23:50 +0100 2021	Thu Jan 14 14:23:50 +0100 2021

5) Traitement de données issues d'un routeur

a)

```
#!/usr/bin/env python # -*- coding: utf-8 -*- import pyspark.sql.functions
as F from pyspark import SparkContext, SparkSession
sc=SparkContext(appName="sparkCSV") df =
SparkSession.read.csv('file:///root/00-00-head10.csv', header=True)
df.agg(F.sum("BYTES")).collect()[0][0]
```

6) Apache Kafka

Pour commencer cette partie je dois sortir de l'image hadoop-master et changer dans le script start-hadoop-cluster-docker.sh l'image utilisée. Avant de modifier le script je dois stop l'image avec le script stop-.....

Je dois passer de la 3.2.0 à la 3.2.1 :

```
#!/bin/sh
IMAGE_NAME=cborelly/hadoop:3.2.1
IMAGE_NAME=registry.iutbeziers.fr/cb_hadoop:3.2.1
```

Maintenant je peux relancer le hadoop-master et commencer la partie

e) Je démarre le serveur zookeeper et un serveur kafka :

```
root@hadoop-master:~# zookeeper-server-start.sh -daemon
$KAFKA_HOME/config/zookeeper.properties
root@hadoop-master:~# kafka-server-start.sh -daemon
$KAFKA_HOME/config/server-master.properties
```

f) Je vérifie que je vois bien les processus JAVA Kafka et QuorumPeerMain avec jps :

```
root@hadoop-master:~# jps
689 Jps
627 Kafka
329 QuorumPeerMain
```

g) Je crée un premier sujet :

```
root@hadoop-master:~# kafka-topics.sh --create --zookeeper hadoop-
master:2181 --replication-factor 1 --partitions 1 --topic test
Created topic "test".
```

h) Je vérifie qu'il apparaît bien dans la liste des sujets du serveur zookeeper :

```
root@hadoop-master:~# kafka-topics.sh --list --zookeeper hadoop-
master:2181
test
```

i) J'essaie de créer un 2ème sujet avec un facteur de réplication de 2 :

```
root@hadoop-master:~# kafka-topics.sh --create --zookeeper hadoop-
master:2181 --replication-factor 2 --partitions 1 --topic test
Error while executing topic command : Replication factor: 2 larger
than available brokers: 1.
[2021-01-14 15:05:42,838] ERROR
org.apache.kafka.common.errors.InvalidReplicationFactorException:
Replication factor: 2 larger than available brokers: 1.
(kafka.admin.TopicCommand$)
```

Le facteur de réplication est invalide. Le facteur maximal disponible est de 1

j) J'essaie de créer un sujet mais avec 2 partitions :

```
root@hadoop-master:~# kafka-topics.sh --create --zookeeper hadoop-
master:2181 --replication-factor 1 --partitions 2 --topic coucou
Created topic "coucou".
```

J'utilise l'option --describe pour afficher les propriétés :

```
root@hadoop-master:~# kafka-topics.sh --describe --zookeeper
hadoop-master:2181
Topic:coucou PartitionCount:2 ReplicationFactor:1 Configs:
  Topic: coucou Partition: 0 Leader: 0 Replicas: 0 Isr: 0
  Topic: coucou Partition: 1 Leader: 0 Replicas: 0 Isr: 0
Topic:test PartitionCount:1 ReplicationFactor:1 Configs:
  Topic: test Partition: 0 Leader: 0 Replicas: 0 Isr: 0
Topic:testPartitions PartitionCount:2 ReplicationFactor:1
  Configs:
  Topic: testPartitions Partition: 0 Leader: 0 Replicas: 0
  Isr: 0
  Topic: testPartitions Partition: 1 Leader: 0 Replicas: 0
  Isr: 0
```

k) Pour effacer un sujet de la liste on peut faire cette commande :

```
root@hadoop-master:~# kafka-topics.sh --delete --zookeeper hadoop-
master:2181 --topic coucou
Topic coucou is marked for deletion.
```

```
Note: This will have no impact if delete.topic.enable is not set
to true.
root@hadoop-master:~# kafka-topics.sh --list --zookeeper hadoop-
master:2181
test
testPartitions
```

l) Maintenant je crée un producteur de données dans la console courante sur un topic existant :

```
root@hadoop-master:~# kafka-console-producer.sh --broker-list
hadoop-master:9092 --topic test
>
```

m) Je dois maintenant lancer un autre terminal sur le container hadoop-master et je lance un consommateur de données sur le même topic :

```
root@debian:~# docker exec -it hadoop-master bash
root@hadoop-master:~# kafka-console-consumer.sh --bootstrap-server
hadoop-master:9092 --topic test
```

n) Je tape plusieurs lignes de texte dans le producteur et je dois le revoir dans le consommateur :

```
root@hadoop-master:~# kafka-console-producer.sh --broker-list
hadoop-master:9092 --topic test
>bonjour
```

```
root@hadoop-master:~# kafka-console-consumer.sh --bootstrap-server
hadoop-master:9092 --topic test
bonjour
```

o) J'ouvre un 3ème terminal et j'essaye de voir si je peux récupérer tous les messages postés sur le sujet depuis le début avec l'option --from-beginning :

```
root@debian:~# docker exec -it hadoop-master bash
root@hadoop-master:~# kafka-console-consumer.sh --bootstrap-server
hadoop-master:9092 --topic test --from-beginning
bonjour
bonjour
coucou
test
```

p) Je vérifie que le module kafka-python est bien présent sur mon container hadoop-master avec pip :

```
root@hadoop-master:~# python -m pip list
DEPRECATION: The default format will switch to columns in the
future. You can use --format=(legacy|columns) (or define a
format=(legacy|columns) in your pip.conf under the [list] section)
to disable this warning.
kafka-python (1.4.4)
pip (9.0.1)
```

q)