Implémentation parallèle de certains algorithmes de fouille de données avec le framework MapReduce

Master 1 : Big Data Analytics

Ousmane DIA :
Abdoulaye Bara DIAW:

Prérequis

PLAN

1-Classique du MapReduce

2-Parallélisation de l’algorithme K-means:
-Implémentation de K-means sur MapReduce

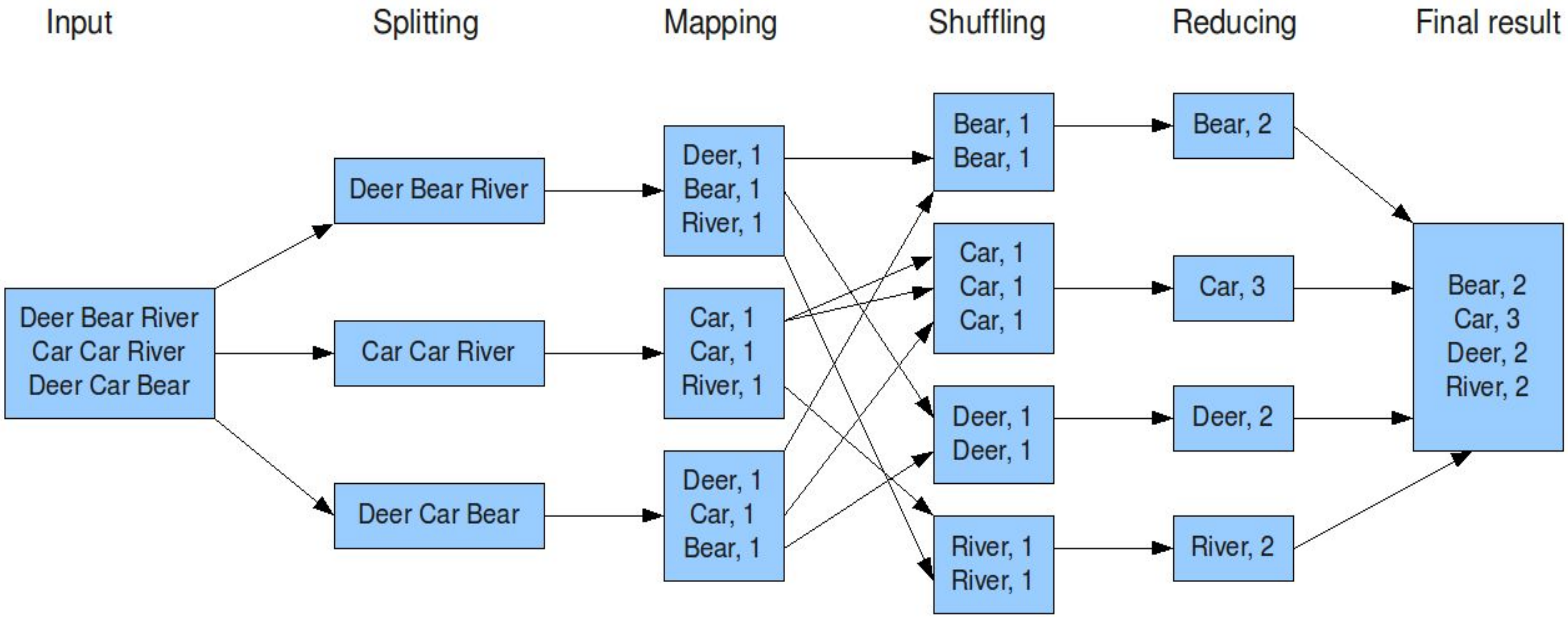
3-Parallélisation de l’algorithme Apriori
-Implémentation de Apriori sur MapReduce

1-Classique du Mapreduce

Description et méthode de Mapreduce

Description et Méthode

The overall MapReduce word count process



1-Classique du Mapreduce

Implémentation de Mapreduce

Implémentation

Data: input.txt

id , age , sexe , adresse , salaire

```
0, 25,homme,oran,28000
1, 33,homme,oran,28000
2, 46,homme,oran,54000
3, 35,famme,oran,33000
4, 23,famme,oran,25000
5, 25,famme,mascara,25000
6, 25,homme,oran,38000
7, 33,homme,oran,38000
8, 46,homme,oran,54000
9, 35,famme,oran,33000
10,23,famme,oran,29000
11,25,famme,mascara,25000
12,25,homme,oran,28000
13,19,homme,oran,18000
14,46,homme,oran,45000
15,35,famme,oran,33000
16,23,famme,oran,23000
17,25,famme,mascara,21000
```

Calculer le maximum et le minimum du Salaire

Implémentation

Démarrage du CLuster Hadoop

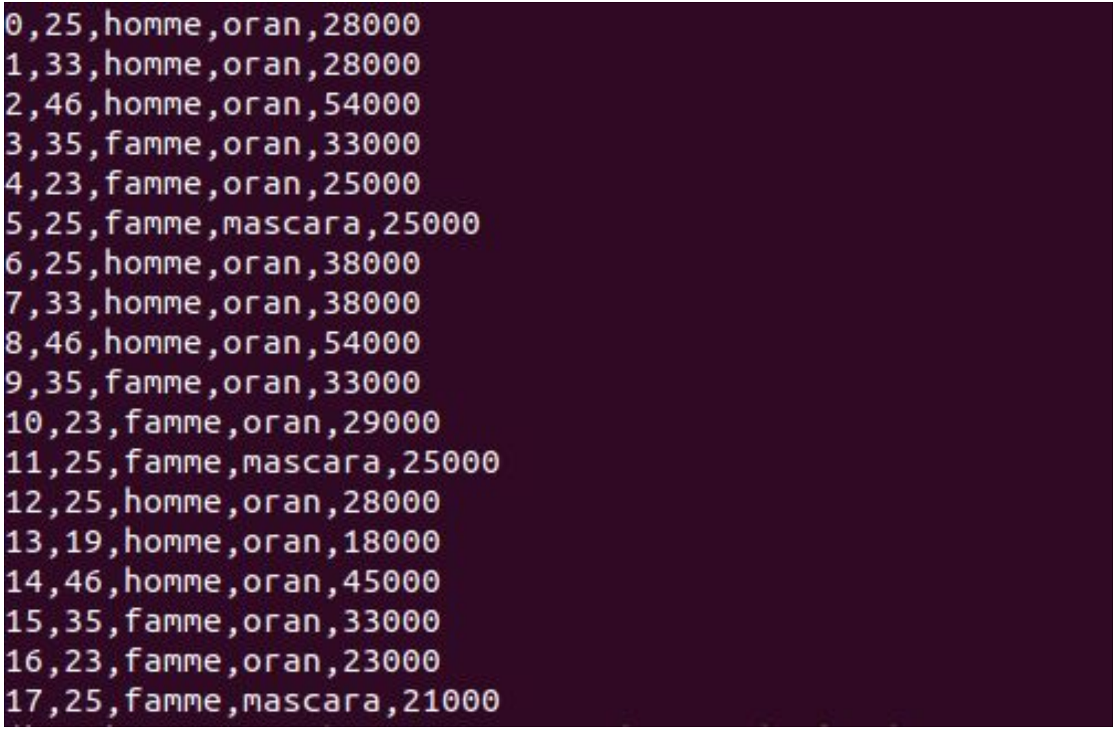
```
start-all.sh
```

Création d’un répertoire HDFS

```
hdfs dfs -mkdir /user
hdfs dfs -mkdir /user/ousmanealhayri/
hdfs dfs -mkdir /user/ousmanealhayri/OusmaneAlhayri
alhayri1234
hdfs dfs -mkdir datainput
hdfs dfs -put data/input.txt datainput/
hdfs dfs -put data/input.txt datainput/
```

Implémentation

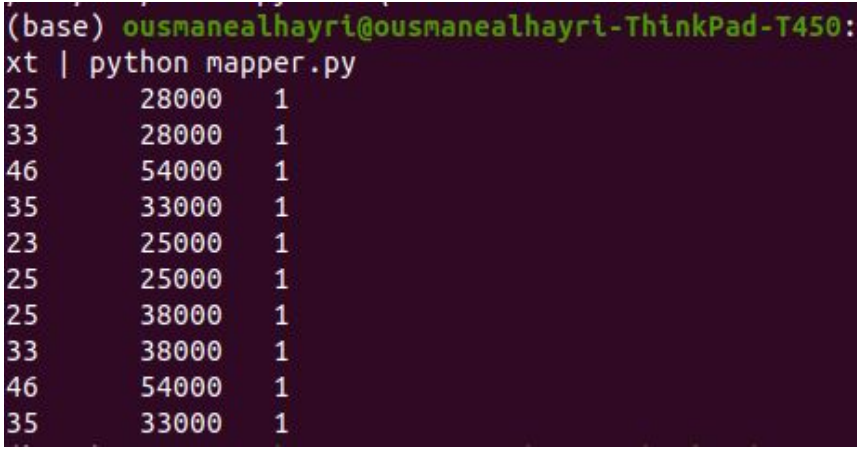
Affichage des données : `hdfs dfs -head datainput/input.txt`



```
0,25,homme,oran,28000
1,33,homme,oran,28000
2,46,homme,oran,54000
3,35,famme,oran,33000
4,23,famme,oran,25000
5,25,famme,mascara,25000
6,25,homme,oran,38000
7,33,homme,oran,38000
8,46,homme,oran,54000
9,35,famme,oran,33000
10,23,famme,oran,29000
11,25,famme,mascara,25000
12,25,homme,oran,28000
13,19,homme,oran,18000
14,46,homme,oran,45000
15,35,famme,oran,33000
16,23,famme,oran,23000
17,25,famme,mascara,21000
```

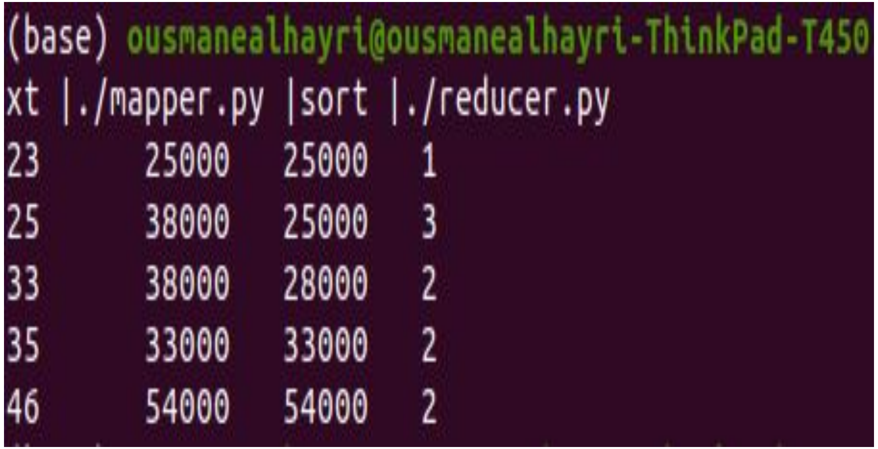
Implémentation

Mapper : `head ../data/input.txt | python mapper.py`



```
(base) ousmanealhayri@ousmanealhayri-ThinkPad-T450:
xt | python mapper.py
25      28000   1
33      28000   1
46      54000   1
35      33000   1
23      25000   1
25      25000   1
25      38000   1
33      38000   1
46      54000   1
35      33000   1
```

Reducer: `head ../data/input.txt |./mapper.py |sort |./reducer.py`



```
(base) ousmanealhayri@ousmanealhayri-ThinkPad-T450
xt |./mapper.py |sort |./reducer.py
23      25000   25000   1
25      38000   25000   3
33      38000   28000   2
35      33000   33000   2
46      54000   54000   2
```

Mapper Code Source:
https://github.com/Data-Mining-on-Hadoop-Mapreduce/Simple-Exemple-of-Mapreduce/blob/main/Code/mapper.py

Reducer Code Source:
https://github.com/Data-Mining-on-Hadoop-Mapreduce/Simple-Exemple-of-Mapreduce/blob/main/Code/reducer.py

2-Parallélisation de l’algorithme K-means

Description et méthode de l’algo k-means

Description et Méthode

Définition :

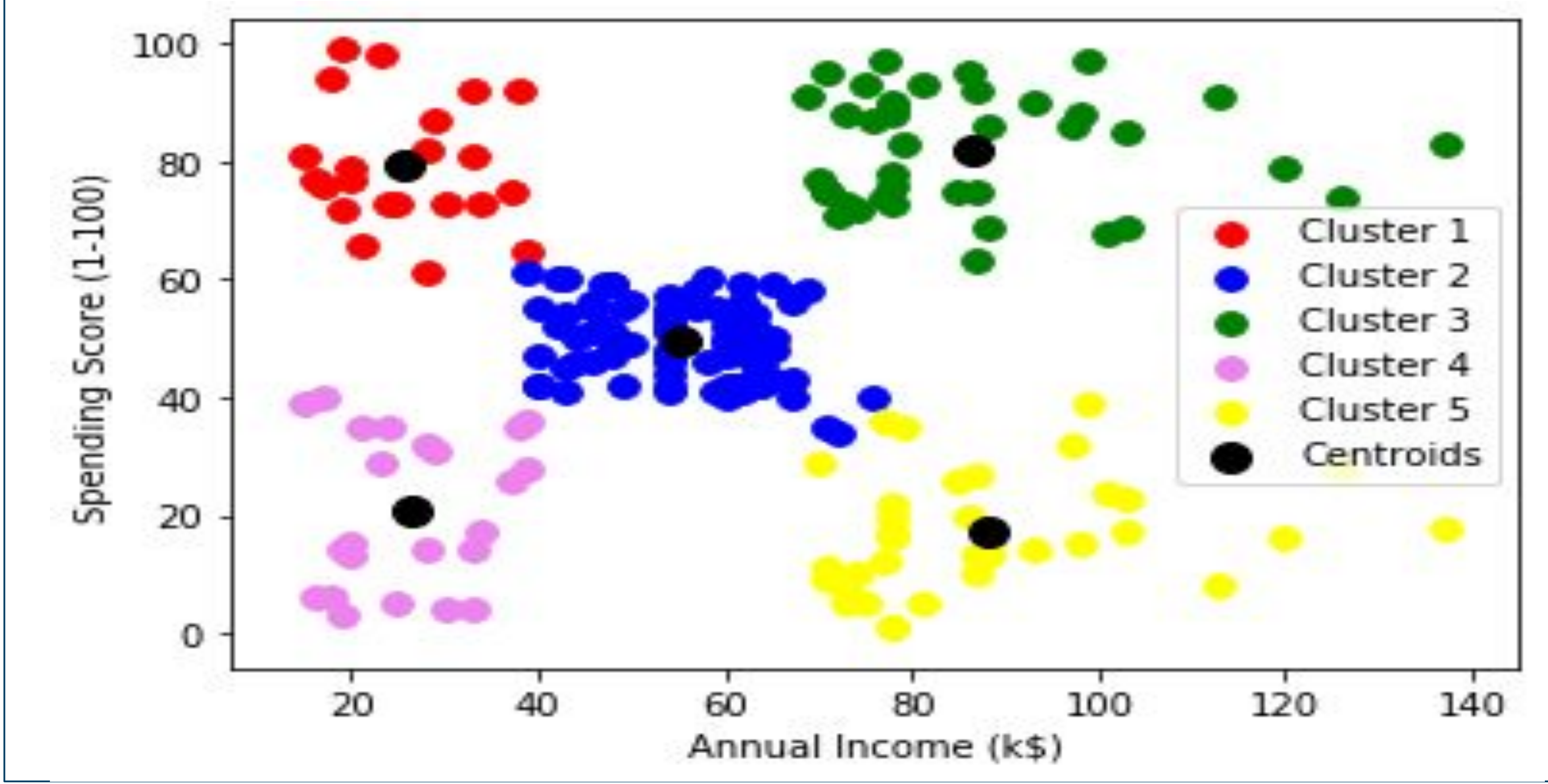
Algorithme de clustering qui regroupe les éléments d'un dataset en groupes qui se ressemblent appelés ***clusters.***

Pour ce faire, l'algorithme s'appuie sur deux éléments essentiels : **le centroïde et la distance**.

Algorithme :

- Initialisation (choix au hasard de k centroïdes)
- Construction de k clusters
- Calcul des nouveaux centroïde
- On recommence jusqu'à ce qu'il y ait convergence

Description et Méthode



2-Parallélisation de l’algorithme K-means

Implémentation de k-means sur Mapreduce

Préparation de l’environnement

hdfs utilise le processeur en le subdivisant en noeuds.
Après notre première opération les nœuds ont été chargés, donc il faut les libérer pour pouvoir effectuer d’autres opérations.

Pour ce faire, nous procédons en trois étapes :

- Supprimer les fichiers temporaires:

```
sudo rm -R /tmp/*
```

- Supprimer les fichiers du datanote/

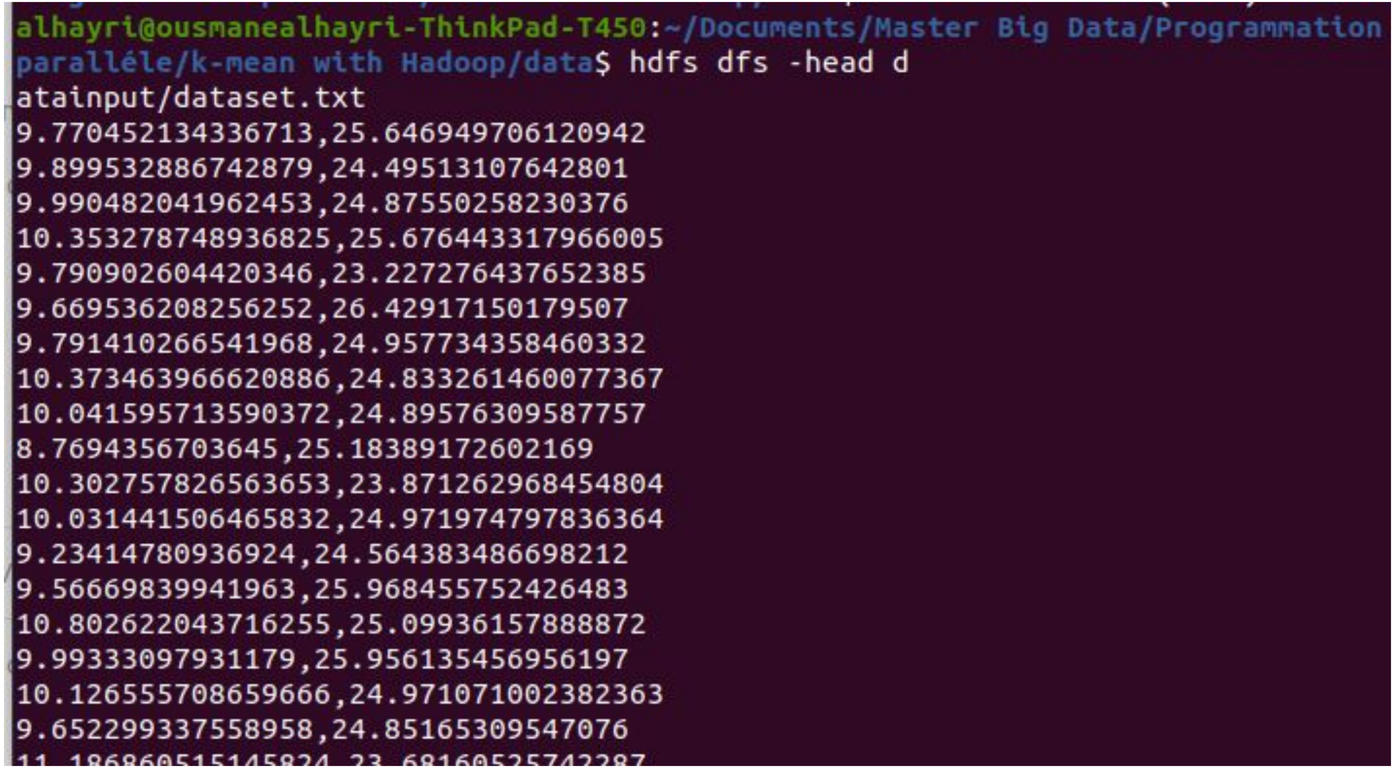
```
rm -rvf /hadoop/hdfs/datanode/*
```

- Formater et redémarrer

```
stop-all.sh
hdfs namenode -format
start-all.sh
jps
```

Implémentation

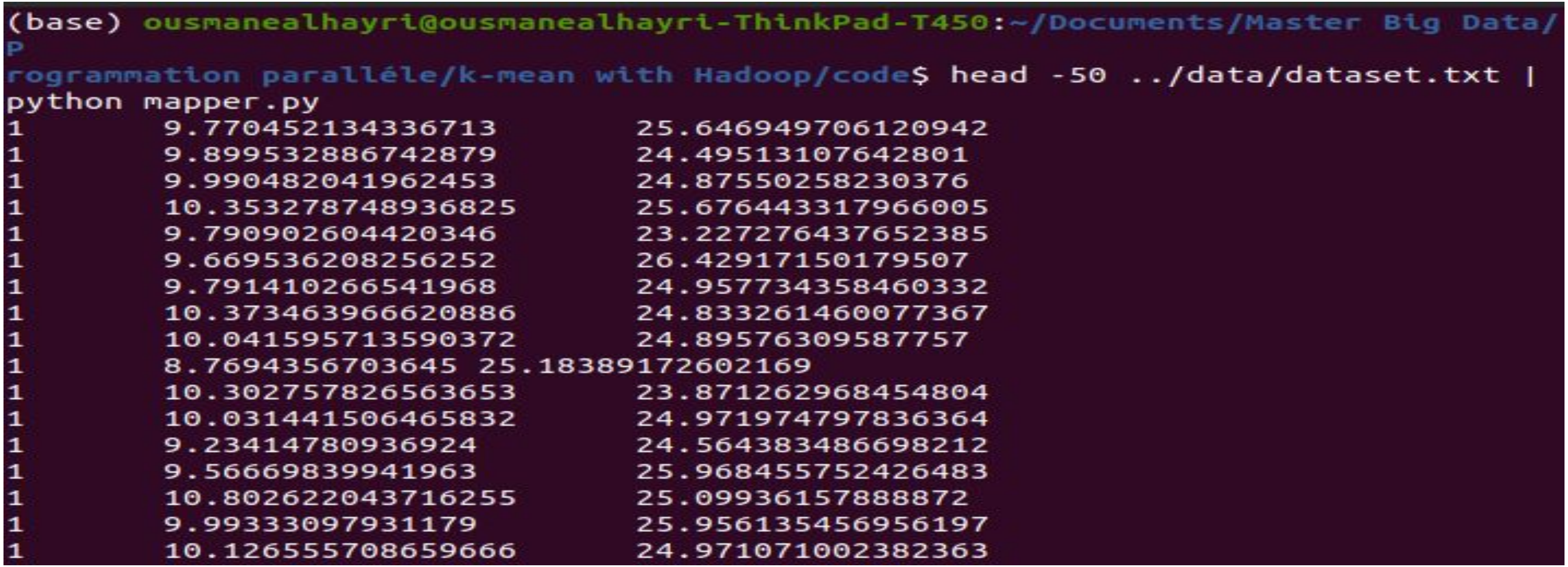
Affichage des données : `hdfs dfs -head datainput/dataset.txt`



```
alhayri@ousmanealhayri-ThinkPad-T450:~/Documents/Master Big Data/Programmation
paralléle/k-mean with Hadoop/data$ hdfs dfs -head d
atainput/dataset.txt
9.770452134336713,25.646949706120942
9.899532886742879,24.49513107642801
9.990482041962453,24.87550258230376
10.353278748936825,25.676443317966005
9.790902604420346,23.227276437652385
9.669536208256252,26.42917150179507
9.791410266541968,24.957734358460332
10.373463966620886,24.833261460077367
10.041595713590372,24.89576309587757
8.7694356703645,25.18389172602169
10.302757826563653,23.871262968454804
10.031441506465832,24.971974797836364
9.23414780936924,24.564383486698212
9.56669839941963,25.968455752426483
10.802622043716255,25.09936157888872
9.99333097931179,25.956135456956197
10.126555708659666,24.971071002382363
9.652299337558958,24.85165309547076
11.186860515145824,23.68160525742287
```

Implémentation

Mapper : `head ../data/dataset.txt | python mapper.py`

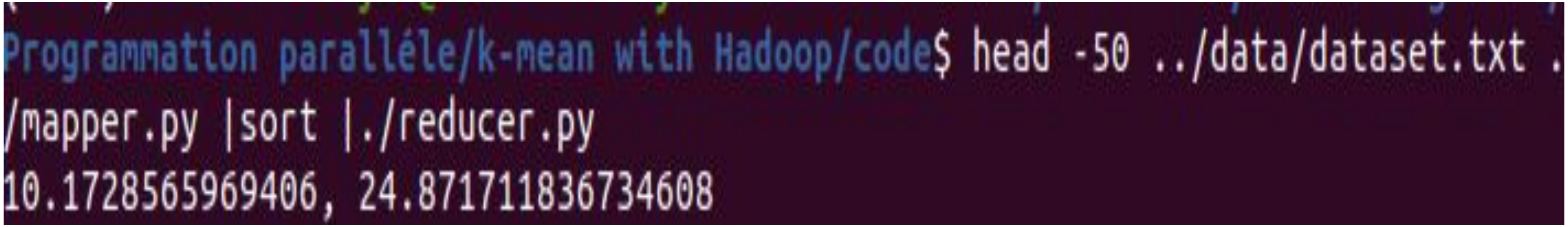


```
(base) ousmanealhayri@ousmanealhayri-ThinkPad-T450:~/Documents/Master Big Data/P
rogrammation paralléle/k-mean with Hadoop/code$ head -50 ../data/dataset.txt |
python mapper.py
1       9.770452134336713       25.646949706120942
1       9.899532886742879       24.49513107642801
1       9.990482041962453       24.87550258230376
1       10.353278748936825      25.676443317966005
1       9.790902604420346       23.227276437652385
1       9.669536208256252       26.42917150179507
1       9.791410266541968       24.957734358460332
1       10.373463966620886      24.833261460077367
1       10.041595713590372      24.89576309587757
1       8.7694356703645 25.18389172602169
1       10.302757826563653      23.871262968454804
1       10.031441506465832      24.971974797836364
1       9.23414780936924        24.564383486698212
1       9.56669839941963        25.968455752426483
1       10.802622043716255      25.09936157888872
1       9.99333097931179        25.956135456956197
1       10.126555708659666      24.971071002382363
```

k-means Mapper Code Source :
https://github.com/Data-Mining-on-Hadoop-Mapreduce/K-means-Algorithm-on-Hadoop/blob/main/code/mapper.py

Implémentation

Reducer: `head ../data/dataset.txt |./mapper.py |sort |./reducer.py`



```
Programmation paralléle/k-mean with Hadoop/code$ head -50 ../data/dataset.txt .
/mapper.py |sort |./reducer.py
10.1728565969406, 24.871711836734608
```

Reducer Code Source :
https://github.com/Data-Mining-on-Hadoop-Mapreduce/K-means-Algorithm-on-Hadoop/blob/main/code/reducer.py

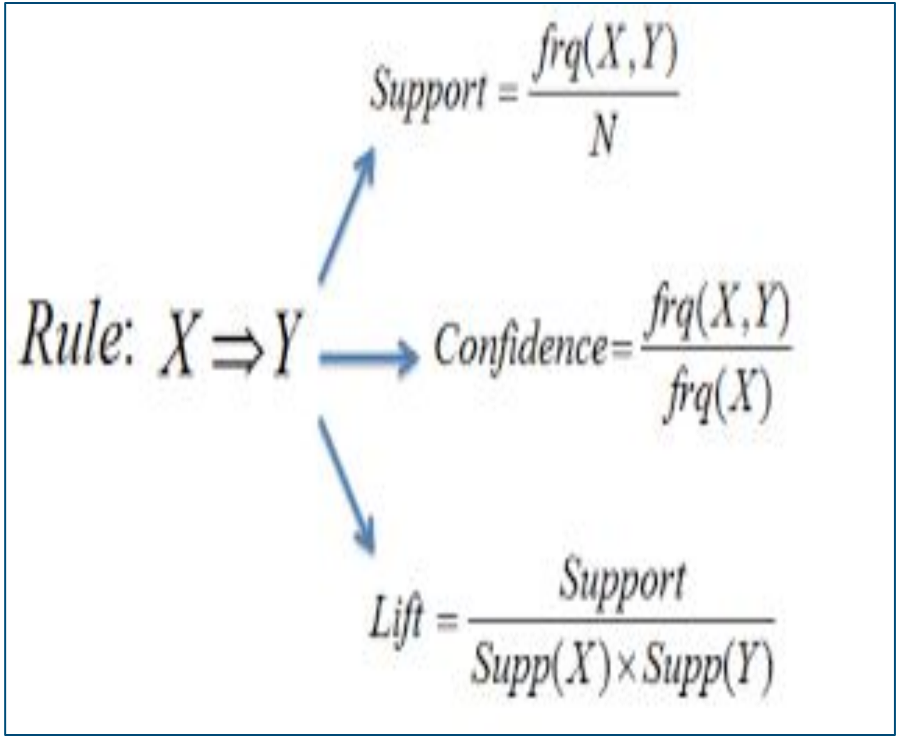
2-Parallélisation de l’algorithme Apriori

Description et méthode de Apriori

Description et méthode

L'algorithme a priori fait référence à l'algorithme utilisé pour calculer les règles d'association entre les objets. Cela signifie savoir comment deux ou plusieurs objets sont liés les uns aux autres.

Description et méthode

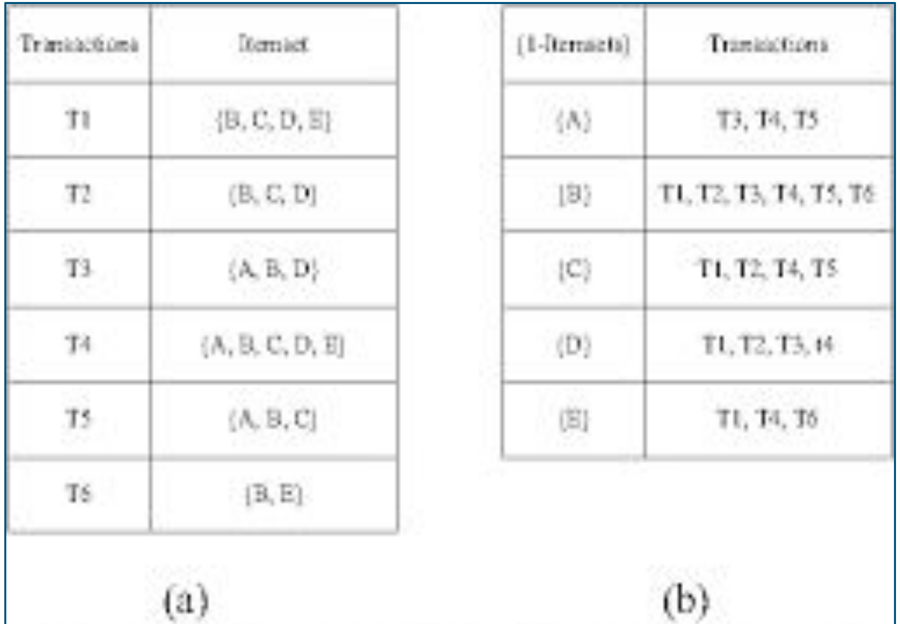


$$Rule:\ X \Rightarrow Y$$

$$Support = \frac{frq(X,Y)}{N}$$

$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

$$Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$



Transactions	Itemset
T1	{B, C, D, E}
T2	{B, C, D}
T3	{A, B, D}
T4	{A, B, C, D, E}
T5	{A, B, C}
T6	{B, E}

(a)

{1-Itemsets}	Transactions
{A}	T3, T4, T5
{B}	T1, T2, T3, T4, T5, T6
{C}	T1, T2, T4, T5
{D}	T1, T2, T3, T4
{E}	T1, T4, T6

(b)

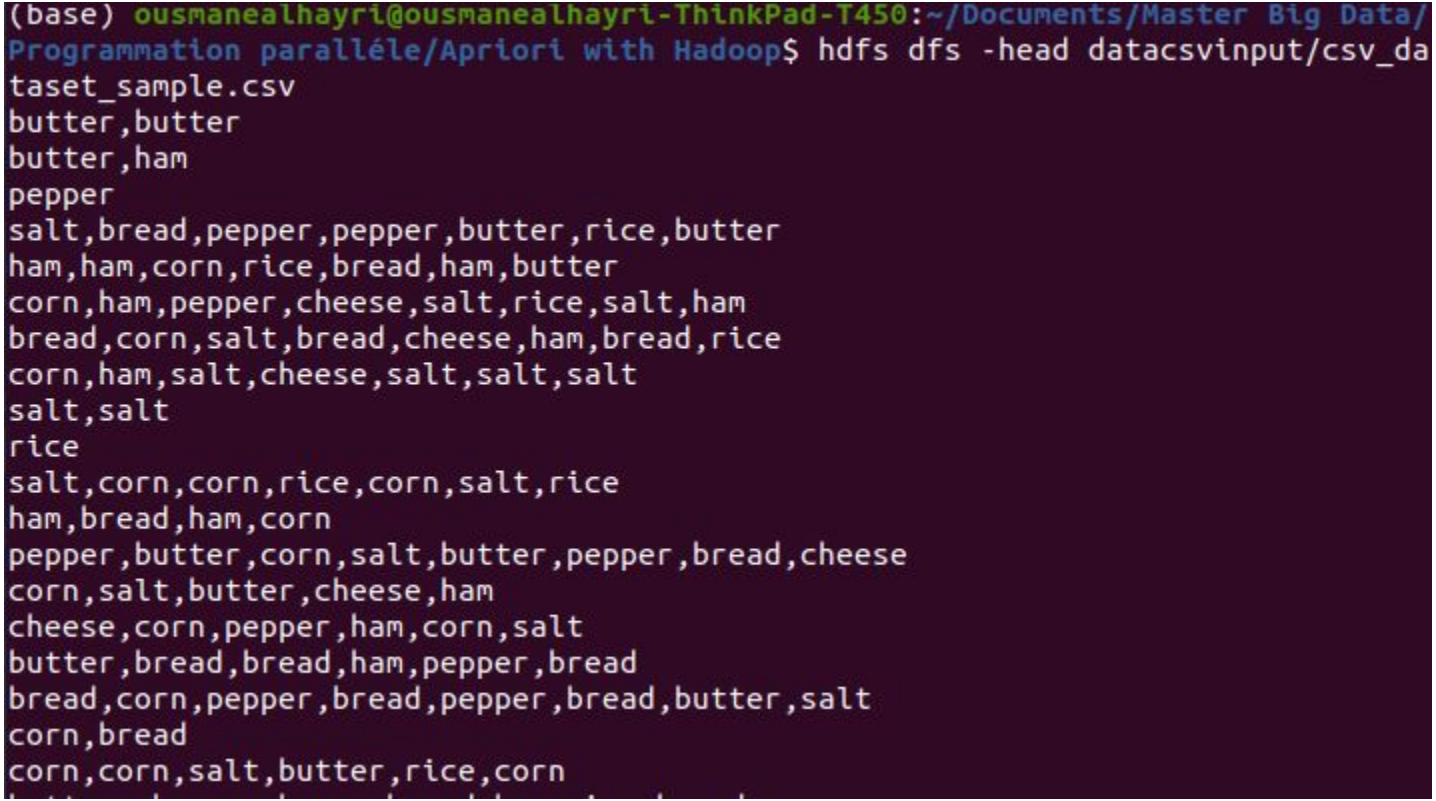
Figure 1. An example to illustrate frequent itemsets mining

2-Parallélisation de l’algorithme Apriori

Implémentation de Apriori sur Mapreduce

Implémentation

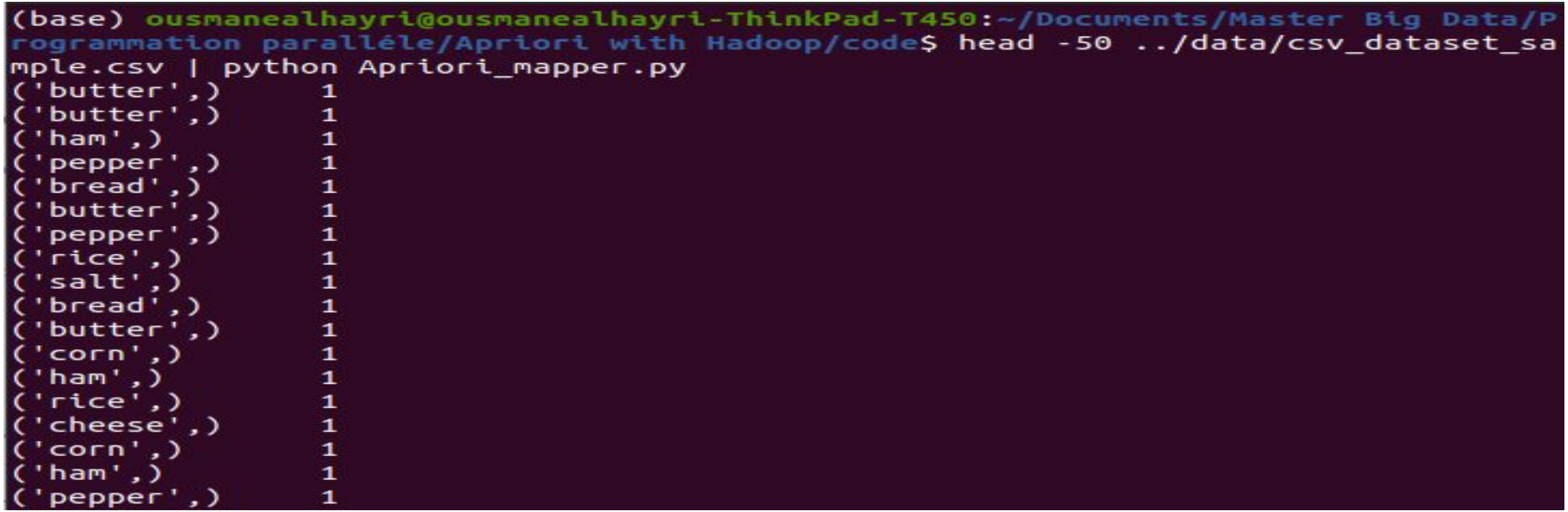
Affichage des données : `hdfs dfs -head datacsvinput/csv_dataset_sample.csv`



```
(base) ousmanealhayri@ousmanealhayri-ThinkPad-T450:~/Documents/Master Big Data/
Programmation paralléle/Apriori with Hadoop$ hdfs dfs -head datacsvinput/csv_da
taset_sample.csv
butter,butter
butter,ham
pepper
salt,bread,pepper,pepper,butter,rice,butter
ham,ham,corn,rice,bread,ham,butter
corn,ham,pepper,cheese,salt,rice,salt,ham
bread,corn,salt,bread,cheese,ham,bread,rice
corn,ham,salt,cheese,salt,salt,salt
salt,salt
rice
salt,corn,corn,rice,corn,salt,rice
ham,bread,ham,corn
pepper,butter,corn,salt,butter,pepper,bread,cheese
corn,salt,butter,cheese,ham
cheese,corn,pepper,ham,corn,salt
butter,bread,bread,ham,pepper,bread
bread,corn,pepper,bread,pepper,bread,butter,salt
corn,bread
corn,corn,salt,butter,rice,corn
```

Implémentation

Mapper : `head -50 ../data/csv_dataset_sample.csv | python Apriori_mapper.py`

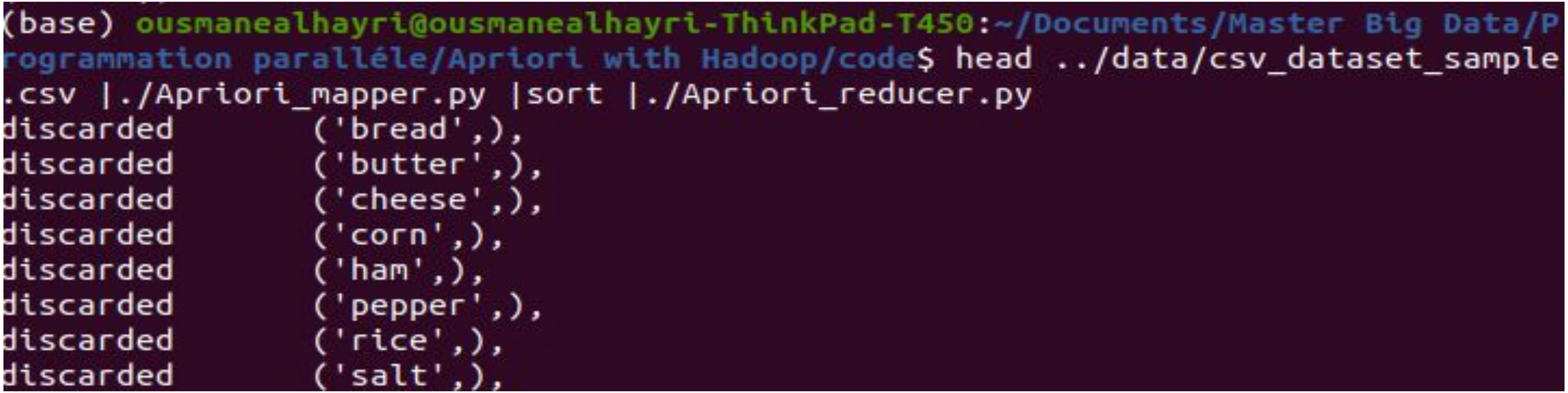


```
(base) ousmanealhayri@ousmanealhayri-ThinkPad-T450:~/Documents/Master Big Data/P
rogrammation paralléle/Apriori with Hadoop/code$ head -50 ../data/csv_dataset_sa
mple.csv | python Apriori_mapper.py
('butter',)     1
('butter',)     1
('ham',)        1
('pepper',)     1
('bread',)      1
('butter',)     1
('pepper',)     1
('rice',)       1
('salt',)       1
('bread',)      1
('butter',)     1
('corn',)       1
('ham',)        1
('rice',)       1
('cheese',)     1
('corn',)       1
('ham',)        1
('pepper',)     1
```

AprioriMapper Code source :
https://github.com/Data-Mining-on-Hadoop-Mapreduce/Apriori-Algorithm-on-Hadoop/blob/main/code/Apriori_mapper.py

Implémentation

Reducer : `head ../data/csv_dataset_sample.csv |./Apriori_mapper.py |sort |./Apriori_reducer.py`



```
(base) ousmanealhayri@ousmanealhayri-ThinkPad-T450:~/Documents/Master Big Data/P
rogrammation paralléle/Apriori with Hadoop/code$ head ../data/csv_dataset_sample
.csv |./Apriori_mapper.py |sort |./Apriori_reducer.py
discarded       ('bread',),
discarded       ('butter',),
discarded       ('cheese',),
discarded       ('corn',),
discarded       ('ham',),
discarded       ('pepper',),
discarded       ('rice',),
discarded       ('salt',),
```

AprioriReducer Code Source:
https://github.com/Data-Mining-on-Hadoop-Mapreduce/Apriori-Algorithm-on-Hadoop/blob/main/code/Apriori_reducer.py

FIN