

Práctica de ejecución de trabajos MapReduce python en Hadoop

Índice

Introducción

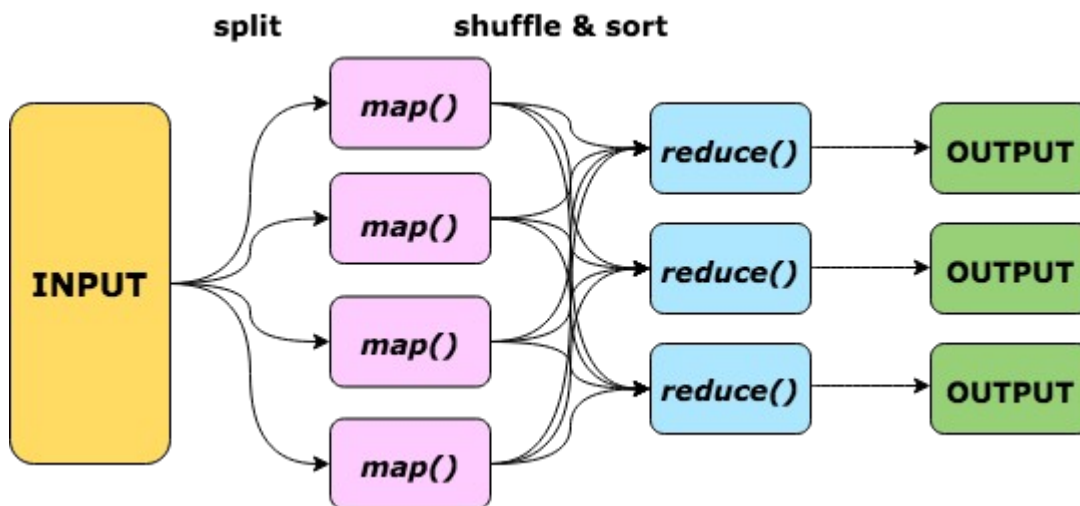
distribuído de grandes cantidades de datos

procesamiento paralelo

mapeadores

reductores

mapeadores



Hadoop Streaming

mapeador

reductor

mapeador

reductor

stdin

stdout

chave

valor

TextInputFormat

- *Hadoop Streaming*

-

```
$ mapred streaming --help
```

Interface para múltiples lenguajes

map

reduce

- **Hadoop Streaming**

-
-

streams

-

- **Hadoop Pipes**

-
-

streaming sockets

■

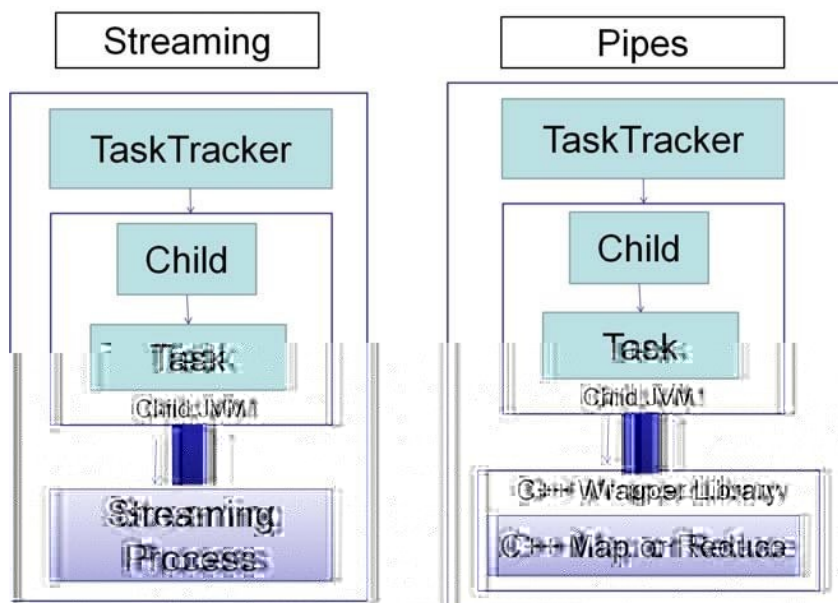
Mapper Reducer

•

-

-

-



Fonte: https://www.researchgate.net/publication/224217002_Hybrid_Map_Task_Scheduling_for_GPU-Based_Heterogeneous_Clusters#pf3

MapReduce en Python: ejemplo de wordcount

Hadoop Streaming

|

wordcount

\$ python3

```
hduser@hadoop-master:~/mr-wordcount-python$ python3
Python 3.8.10 (default, May 26 2023, 14:05:08)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
hduser@hadoop-master:~/mr-wordcount-python$
```

el_quijote.txt

```
hduser@hadoop-master:~/mr-wordcount-python$ ls -l
total 8
-rw-rw-r-- 1 hduser hduser 296 Nov  5 20:03 mapper.py
-rw-rw-r-- 1 hduser hduser 600 Nov  5 20:35 reducer.py
hduser@hadoop-master:~/mr-wordcount-python$ ls -l ../el_quijote.txt
-rw-rw-r-- 1 hduser hduser 1060259 Nov  5 13:13 ../el_quijote.txt
hduser@hadoop-master:~/mr-wordcount-python$
```

Hadoop Streaming: Mapper

parsear

-
-

```
$ nano mapper.py
```

```
#!/usr/bin/python3
# A primeira liña indica que é un script python

import sys
# Importamo-la librería sys para poder ler e escribir datos en STDIN e STDOUT

# Ler unha liña de STDIN (standard input):
# pasarémoslle un arquivo sobre o que conta-las palabras
for linha in sys.stdin:
    # Para cada liña eliminamo-los espazos ó principio e final
    linha = linha.strip()
    # Separa as palabras tomando como separador, por defecto, o espazo en branco
    # e gárdalas nunha lista
    palabras = linha.split()
    # Percorre cun bucle a lista de palabras imprimindo na saída estándar STDOUT
    # a palabra co valor de conteo 1
    for palabra in palabras:
        # Escribe os resultados no STDOUT (standard output);
        # o que se poña aquí será a entrada para o paso Reduce
        # Esta saída será o input para reducer.py
        print(palabra, "\t1")
    # creamos tuplas de (palabra, 1)
```

mapper

```
$ cat ../el_quijote.txt | python3 mapper.py
```

```
don 1
Quijote. 1
Forse 1
altri 1
canterà 1
con 1
miglior 1
plettro. 1
FINIS 1
hduser@hadoop-master:~/mr-wordcount-python$ cat el_quijote.txt | python3 mapper.py
```

Hadoop Streaming: Reducer

reducer

mapper

-
-

```
$ nano reducer.py
```

```
#!/usr/bin/python3
# A primeira liña indica que é un script python

import sys
# Importamo-la librería sys para poder ler e escribir datos en STDIN e STDOUT

# Inicializamo-lo diccionario
dicPalabras = {}

# ler unha liña de STDIN (standard input):
# pasarémoslle un arquivo cunha palabra co valor 1 en cada liña
for linha in sys.stdin:
    # Para cada liña eliminamo-los espazos ó principio e final
    linha = linha.strip()
    # Parseamo-la entrada de mapper.py;
    # separa as palabras tomando o tabulador como separador:
    # o formato proporcionado por mapper.py era unha palabra e un número (un),
    # así que só se fai 1 troceado, para obte-los dous valores
    palabra, conta = linha.split('\t', 1)
    # Converter conta (que realmente é string) a int
    try:
        conta = int(conta)
    # Cando conta non é un número, sixilosamente,
    # se ignora/descarta esta liña
    except ValueError:
        continue
    # Se incrementa o valor gardado no diccionario para palabra
    # grazas a que esta é a chave do mesmo
    try:
        dicPalabras[palabra] += conta
    except:
        # Se palabra aínda non estaba no diccionario se mete co valor conta (1)
        dicPalabras[palabra] = conta

# Percórrese todo o diccionario
for palabra in dicPalabras.keys():
    # Escribese cada palabra e a súa conta na STDOUT
    print(palabra, "\t", dicPalabras[palabra])
```

```
$ cat ../el_quijote.txt | python3 mapper.py | python3 reducer.py
> ../conta_quijote.tsv
```

```

hduser@hadoop-master:~/nr-wordcount-python$ cat ../el_quijote.txt | python3 mapper.py | python3 reducer.py > ../conta_quijote.tsv
hduser@hadoop-master:~/nr-wordcount-python$ head ../conta_quijote.tsv
DON      6
QUIJOTE  6
DE       17
LA       13
MANCHA   2
Miguel   3
de       8947
Cervantes 1
Saavedra 1
PRIMERA  1
hduser@hadoop-master:~/nr-wordcount-python$ tail ../conta_quijote.tsv
declárase. 1
Tiénesa 1
vigilias 1
sacallos 1
Forse 1
altrí 1
canterá 1
miglior 1
pletto. 1
FINIS 1

```

permisos de ejecución

Hadoop Streaming

```

$ chmod u+x mapper.py
$ chmod u+x reducer.py

```

```

$ cat ../el_quijote.txt | ./mapper.py | ./reducer.py > ../conta_quijote.tsv

```


importante diferencia a cando se execute con hadoop streaming!

Con **hadoop streaming** os datos que saen do proceso *map* son ordeados pola chave automaticamente antes de pasarse ó proceso *reduce*, sempre e cando haxa algún proceso *reduce*

mapper

reducers

```
$ cat ../el_quijote.txt | python3 mapper.py | python3 reducer.py | sort -t$'\t' -k1,1 > ../conta_quijote_ordeada.tsv
```

```
hduser@hadoop-master:~/mr-wordcount-python$ head ../conta_quijote_ordeada.tsv
,      1
-      4
.      2
i      1
-      1
1:     1
10:    1
11:    1
12:    1
13:    1
```

```
$ cat ../el_quijote.txt | python3 mapper.py | python3 reducer.py | sort -t$'\t' -nk2 > ../conta_quijote_ordeada_conta.tsv
```

```
hduser@hadoop-master:~/mr-wordcount-python$ tail ../conta_quijote_ordeada_conta.tsv
los      2122
se       2382
no       2786
el       3726
en       3883
a        4725
la       4941
y        8042
de       8947
que     10351
```

itertools

groupby()

Hadoop Streaming: envío de trabajo a hadoop

imos procesalos dentro
de Hadoop para aproveita-la computación distribuída

Hadoop Streaming

stdin stdout stdin mapper reducer stdout

Hadoop-streaming

-input -output
-mapper -reducer

Parámetro		Exemplo	Observacións

Referencia: <https://hadoop.apache.org/docs/stable/hadoop-streaming/HadoopStreaming.html>

Atención!:

-files -archives -input
-D

Nota

mapper.py python3 mapper.py

```
$ chmod +x *.py
```

jobs

```
mapred streaming \  
-files mapper.py, reducer.py \  
-input meuCartafoIEntradaHDFS \  
-output meuCartafoISaídaHDFS \  
-mapper scriptMapper \  
-reducer scriptReducer
```

Nota mapred hadoop jar

mapred hadoop jar

```
hadoop jar rutaDeHadoopStreaming.jar \  
-files mapper.py, reducer.py
```

```
-input meuCartafoIEntradaHDFS \
-output meuCartafoISaídaHDFS \
-mapper scriptMapper \
-reducer scriptReducer
```

\$HADOOP_HOME/share/hadoop/tools/lib

para poder aproveita-la capacidade de computación distribuída, hai que subir ó sistema hdfs o ficheiro a tratar *el_quijote.txt*

```
$ hdfs dfs -put ../el_quijote.txt
```

scripts python podemos collelos do sistema local

-file

```
$ mapred streaming -input el_quijote.txt -output saida_quijote -mapper mapper.py
-file mapper.py -reducer reducer.py -file reducer.py
```

-file

-files

mapper

reducer

```
-files ruta/reducer.py,ruta/mapper.py
```

-files

-input

-files

```
$ mapred streaming -files ./mapper.py,./reducer.py -input el_quijote.txt -output
saida_quijote -mapper mapper.py -reducer reducer.py
```

job

```
$ hdfs dfs -head /user/hduser/saida_quijote/part-00000
```

```
hduser@hadoop-master:~/mr-wordcount-python$ hdfs dfs -head /user/hduser/saida_quijote/part-00000
"Apenas          1
"Caballero       4
"Conde           1
"Donde           1
"Más            1
"Miau",          1
"No              1
"Rastrea         1
"Ricamonte",     1
```

permisos de execución ós scripts

Hadoop Streaming

```
$ chmod u+x mapper.py
$ chmod u+x reducer.py
```

```
$ mapred streaming \
-files ./mapper.py,./reducer.py \
-mapper mapper.py \
-reducer reducer.py \
-input ./el_quijote.txt \
-output ./contando_quijote_sen_x8
```

```
hduser@hadoop-master:~/mr-wordcount-python$ mapred streaming \
> -files ./mapper.py,./reducer.py \
> -mapper mapper.py \
> -reducer reducer.py \
> -input ./el_quijote.txt \
> -output ./contando_quijote_sen_x8
packageJobJar: [] [/usr/share/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.1.jar] /tmp/streamjob4014838181292134924.jar tmpDir=null
2023-11-07 22:35:11,844 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at hadoop-master/10.0.2.12:8032
2023-11-07 22:35:12,024 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at hadoop-master/10.0.2.12:8032
2023-11-07 22:35:12,334 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hduser/.staging/job_1699388564941_0008
2023-11-07 22:35:13,137 INFO mapred.FileInputFormat: Total input files to process : 1
2023-11-07 22:35:13,383 INFO mapreduce.JobSubmitter: number of splits:2
2023-11-07 22:35:13,664 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1699388564941_0008
```

Advertencia

mapreduce

verbose output

```
INFO mapreduce.Job: Job ... completed successfully
```

Erro habitual

```
ERROR streaming.StreamJob: Error Launching job : Output directory ... already exists
Streaming Command Failed!
```

hadoop-streaming *hadoop jar:*

```
$ hdfs dfs -put ../el_quijote.txt / # non fai falta porque xa o subimos antes
```

hadoop-streaming

```
/usr/share/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.1.jar
```

mapper

reducer

```
$ /usr/share/hadoop/bin/hadoop jar
/usr/share/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.1.jar -file
./mapper.py -mapper 'python3 mapper.py' -file ./reducer.py -reducer 'python3
reducer.py' -input ./el_quijote.txt -output ./contando_quijote
```

```
hduser@hadoop-master:~/mr-wordcount-python$ /usr/share/hadoop/bin/hadoop jar /usr/share/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.1.jar
-file ./mapper.py -mapper 'python3 mapper.py' -file ./reducer.py -reducer 'python3 reducer.py' -input ./el_quijote.txt -output ./contando_quijote
_sen_x8
2023-11-07 22:08:01,271 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [./mapper.py, ./reducer.py, /tmp/hadoop-unjar4149062273524316776/] [] /tmp/streamjob1495805255852789092.jar tmpDir=null
2023-11-07 22:08:02,893 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at hadoop-master/10.0.2.12:8032
2023-11-07 22:08:03,050 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hduser/.staging/job_1699388564941_0008
2023-11-07 22:08:03,291 INFO mapred.FileInputFormat: Total input files to process : 1
2023-11-07 22:08:03,531 INFO mapreduce.JobSubmitter: number of splits:2
```

-files

```
$ /usr/share/hadoop/bin/hadoop jar /usr/share/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.1.jar -files ./mapper.py,./reducer.py -mapper 'python3 mapper.py' -reducer 'python3 reducer.py' -input ./el_quijote.txt -output ./contando_quijote_sen_x4
```

permisos de ejecución ós scripts

Hadoop Streaming

Hadoop Streaming: saída resultante da execución dos traballos

-output

- ***_SUCCESS***

○

- ***part-X-nnnnn***

○

▪ ***m*** ***map*** ***reduce***

▪ ***r*** ***reduce***

▪ ***X***

part-00000

○ ***nnnnn***

-

reducers

part-r-

00000 part-r-00029

reducer

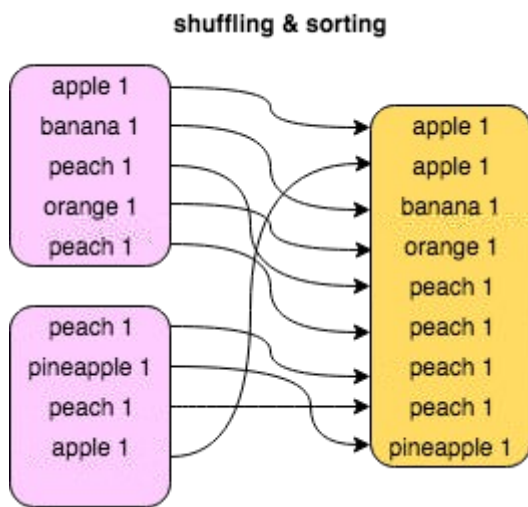
head tail cat

\$ hdfs dfs -head /user/hduser/contando_quijote_sen_x5/*

```
hduser@hadoop-master:~$ hdfs dfs -ls /user/hduser/contando_quijote_sen_x5/
Found 2 items
-rw-r--r--  2 hduser supergroup          0 2023-11-07 22:24 /user/hduser/contando_quijote_sen_x5/_SUCCESS
-rw-r--r--  2 hduser supergroup 1326047 2023-11-07 22:24 /user/hduser/contando_quijote_sen_x5/part-00000
hduser@hadoop-master:~$ hdfs dfs -cat /user/hduser/contando_quijote_sen_x5/_SUCCESS
hduser@hadoop-master:~$ hdfs dfs -head /user/hduser/contando_quijote_sen_x5/part-00000
"Apénas"
"Oaballero"
"Conde"
"Donde"
"Mas"
"Mas"
```

map-reduce

Con **hadoop streaming** os datos que saen do proceso *map* son ordeados pola chave automaticamente antes de pasarse ó proceso *reduce*, sempre e cando haxa algún proceso *reduce*



Hadoop Streaming: outro exemplo sinxelo de wordcount distribuído usando MapReduce, con binarios.

cat

wc

MapReduce

```
$ mapred streaming \  
-mapper /bin/cat \  
-reducer /usr/bin/wc \  
-input ./el_quijote.txt \  
-output ./contando_quijote_bin
```

-
-
-

```
$ hdfs dfs -cat ./contando_quijote_bin/*
```

```
$ wc ../el_quijote.txt
```

```
IO_ERROR=0  
WRONG_LENGTH=0  
WRONG_MAP=0  
WRONG_REDUCE=0  
File Input Format Counters  
  Bytes Read=1064355  
File Output Format Counters  
  Bytes Written=25  
2023-11-10 20:11:10,198 INFO streaming.StreamJob: Output directory: ./contando_quijote_bin  
hduser@hadoop-master:~/mr-wordcount-python$ hdfs dfs -cat ./contando_quijote_bin/*  
2186 187018 1062445  
hduser@hadoop-master:~/mr-wordcount-python$ wc ../el_quijote.txt  
2186 187018 1060259 ../el_quijote.txt  
hduser@hadoop-master:~/mr-wordcount-python$ hdfs dfs -ls ./contando_quijote_bin/*  
-rw-r--r--  2 hduser supergroup      0 2023-11-10 20:11 contando_quijote_bin/_SUCCESS  
-rw-r--r--  2 hduser supergroup    25 2023-11-10 20:11 contando_quijote_bin/part-00000  
hduser@hadoop-master:~/mr-wordcount-python$
```

el_quijote.txt

```
hduser@hadoop-master:~/mr-wordcount-python$ hdfs dfs -ls ../el_quijote.txt  
-rw-r--r--  2 hduser supergroup 1060259 2023-11-05 13:25 el_quijote.txt  
hduser@hadoop-master:~/mr-wordcount-python$ ls -ls ../el_quijote.txt  
1036 -rw-rw-r-- 1 hduser hduser 1060259 Nov  5 13:13 ../el_quijote.txt  
hduser@hadoop-master:~/mr-wordcount-python$
```


Hadoop Streaming: outro exemplo sinxelo de wordcount distribuído usando MapReduce, con scripts bash.

-

mapper

map.sh

map.sh

```
#!/bin/bash
while read line
do
  for word in $line
  do
    #sáltase as liñas baleiras
    if [[ "$line" =~ [^[:space:]] ]]
    then
      if [ -n "$word" ]
      then
        echo -e ${word} "\t1"
      fi
    fi
  done
done
```

froitaria.py

```
#!/usr/bin/env python3

# froitaria.py
# froitaria é unha modificación de charlatan.py crea un ficheiro de texto con
# froitas pseudoaleatorias que se toman
# dunha lista que pode definir o/a programador/a
#
# Pode definirse o número de palabras para crear un ficheiro máis grande

import random

ficheiro = open("froitas.txt","a")
pseudoPalabrasAleatorias = ["Mazá ", "Laranxa ", "Pera ", "Pexego ", "Figo ",
"Peladillo ", "Breva ", "Plátano ", "Níspero ", "Fatón ", "Ameixa ", "Cirola "]
cantidadPalabrasAleatorias = len(pseudoPalabrasAleatorias) - 1

#Aumenta NUMERO_PALABRAS_RESULTANTES para crear un ficheiro maior
#150000000 -> 112MB
#300000000 -> 231MB
```

```
#50 -> 1KB
NUMERO_PALABRAS_RESULTANTES = 50

indice = 0
for x in range(NUMERO_PALABRAS_RESULTANTES):
    indice = random.randint(0, cantidadPalabrasAleatorias)
    ficheiro.write(pseudoPalabrasAleatorias[indice])
    if x % 20 == 0:
        ficheiro.write('\n')
```

froitas.txt

```
$ /usr/bin/python3 froitaria.py
```

```
$ cat froitas.txt
```

mapper

```
$ cat froitas.txt | bash map.sh
```

map.sh

```
$ chmod u+x map.sh
$ cat froitas.txt | ./map.sh
hduser@hadoop-master:~/mr-wordcount-bash$ ls -l map.sh
-rw----- 1 hduser hduser 208 Nov 11 16:00 map.sh
hduser@hadoop-master:~/mr-wordcount-bash$ cat froitas.txt | ./map.sh
bash: ./map.sh: Permiso denegado
hduser@hadoop-master:~/mr-wordcount-bash$ chmod u+x map.sh
hduser@hadoop-master:~/mr-wordcount-bash$ ls -l map.sh
-rwx----- 1 hduser hduser 208 Nov 11 16:00 map.sh
hduser@hadoop-master:~/mr-wordcount-bash$ cat froitas.txt | ./map.sh
Cirola 1
Ameixa 1
Figo 1
Figo 1
```

froitaria.py

```
$ hdfs dfs -mkdir -p mr-wordcount-bash/input
$ hdfs dfs -put froitas.txt mr-wordcount-bash/input
$ hdfs dfs -ls -l -R mr-wordcount-bash/input
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -rm -R mr-wordcount-bash
Deleted mr-wordcount-bash
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -mkdir -p mr-wordcount-bash/input
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -put froitas.txt mr-wordcount-bash/input
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -ls -h -R mr-wordcount-bash/input
-rw-r--r-- 2 hduser supergroup 206.5 M 2023-11-11 17:08 mr-wordcount-bash/input/froitas.txt
hduser@hadoop-master:~/mr-wordcount-bash$
```

map.sh

reducer

/bin/cat

```
$ mapred streaming \
-files map.sh \
-input mr-wordcount-bash/input \
```

```
-output mr-wordcount-bash/output \  
-mapper map.sh \  
-reducer /bin/cat
```

```
$ hdfs dfs -ls -h -R mr-wordcount-bash/output  
$ hdfs dfs -head mr-wordcount-bash/output/part-00000  
$ hdfs dfs -tail mr-wordcount-bash/output/part-00000
```

```
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -head mr-wordcount-bash/output/part-00000  
Ameixa 1  
Ameixa 1  
Ameixa 1  
Ameixa 1  
Ameixa 1  
Ameixa 1  
Ameixa 1  
Ameixa 1  
Ameixa 1  
Ameixa 1
```

```
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -tail mr-wordcount-bash/output/part-00000  
1  
Plátano 1  
Plátano 1  
Plátano 1  
Plátano 1  
Plátano 1  
Plátano 1  
Plátano 1
```

hadoop streaming os datos
que saen do proceso *map* son ordeados pola chave automaticamente antes de pasarse ó
proceso *reduce*, sempre e cando haxa algún proceso *reduce* *mapper*
reducers

reducer

reduce.sh

reducer

reduce.sh

```
#!/bin/bash
currkey=""
currcount=0
while IFS=$'\t' read -r key val
do
    if [[ $key == $currkey ]]
    then
        currcount=$(( currcount + val ))
    else
        if [ -n "$currkey" ]
        then
            echo -e ${currkey} "\t" ${currcount}
        fi
        currkey=$key
        currcount=1
    fi
done
# last one
echo -e ${currkey} "\t" ${currcount}
```

reducer

```
$ chmod u+x reduce.sh
$ cat froitas.txt | ./map.sh |sort| ./reduce.sh
hduser@hadoop-master:~/mr-wordcount-bash$ cat froitas.txt | ./map.sh |sort| bash ./reduce.sh
Ameixa      75
Breva       83
Cirola      93
Fatón       80
Figo        82
Laranxa          75
Mazá        83
Nispero          82
Peladillo     75
Pera         77
Pexego       87
Plátano       79
```

```
$ hdfs dfs -ls -R -h mr-wordcount-bash/output
$ hdfs dfs -rm -R mr-wordcount-bash/output
$ mapred streaming \
  -files map.sh,reduce.sh \
  -input mr-wordcount-bash/input \
```

```
-output mr-wordcount-bash/output \  
-mapper map.sh \  
-reducer reduce.sh
```

```
$ hdfs dfs -ls -h -R mr-wordcount-bash/output  
$ hdfs dfs -head mr-wordcount-bash/output/part-00000  
$ hdfs dfs -tail mr-wordcount-bash/output/part-00000
```

```
File Output Format Counters  
  Bytes Written=206  
2023-11-11 18:56:41,346 INFO streaming.StreamJob: Output directory: mr-wordcount-bash/output  
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -ls -h -R mr-wordcount-bash/output  
-rw-r--r--  2 hduser supergroup      0 2023-11-11 18:56 mr-wordcount-bash/output/_SUCCESS  
-rw-r--r--  2 hduser supergroup 206 2023-11-11 18:56 mr-wordcount-bash/output/part-00000  
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -head mr-wordcount-bash/output/part-00000  
Ameixa 2497248  
Breva 2499759  
Cirola 2501110  
Fatón 2500401  
Figo 2501498  
Laranxa 2500812  
Mazá 2499635  
Nispero 2497593  
Peladillo 2498881  
Pera 2500684  
Pexego 2501307  
Plátano 2501122  
hduser@hadoop-master:~/mr-wordcount-bash$
```

Resolvendo outras consultas: froitas máis e menos repetidas

sort

k2 n r

```
$ hdfs dfs -cat mr-wordcount-bash/output/part-00000 | sort -k2nr  
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -cat mr-wordcount-bash/output/part-00000 | sort -k2nr  
Figo 2501498  
Pexego 2501307  
Plátano 2501122  
Cirola 2501110  
Laranxa 2500812  
Pera 2500684  
Fatón 2500401  
Breva 2499759  
Mazá 2499635  
Peladillo 2498881  
Nispero 2497593  
Ameixa 2497248  
hduser@hadoop-master:~/mr-wordcount-bash$
```

Ordea-los resultados mediante un traballo MapReduce

swap_keyval.sh

```
#!/bin/bash
# This script will read one line at a time and swap key/value
# For instance, the line "word 100" will become "100 word"

while read key val
do
    printf "%s\t%s\n" "$val" "$key"
done
```

```
$ chmod u+x swap_keyval.sh
```

output

output_sorted

Nota Apache Spark

deficiencias de MapReduce

escribir datos no disco en cada paso dun pipeline de transformación de datos (o cal leva tempo e iso pode ser moi custoso para canalizacións de datos máis longas)

```
$ hdfs dfs -rm -R mr-wordcount-bash/outputsorted
$ mapred streaming \
  -files swap_keyval.sh \
  -input mr-wordcount-bash/output \
  -output mr-wordcount-bash/outputsorted \
  -mapper swap_keyval.sh
```

```
$ hdfs dfs -ls -h -R mr-wordcount-bash/outputsorted
$ hdfs dfs -head mr-wordcount-bash/outputsorted/part-00000
$ hdfs dfs -tail mr-wordcount-bash/outputsorted/part-00000
```

```
Bytes Written=182
2023-11-11 19:10:30,981 INFO streaming.StreamJob: Output directory: mr-wordcount-bash/output2
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -ls -h -R mr-wordcount-bash/output2
-rw-r--r--  2 hduser supergroup      0 2023-11-11 19:10 mr-wordcount-bash/output2/_SUCCESS
-rw-r--r--  2 hduser supergroup    182 2023-11-11 19:10 mr-wordcount-bash/output2/part-00000
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -head mr-wordcount-bash/output2/part-00000
2497248 Anelxa
2497593 Nispero
2498081 Peladillo
2499635 Maza
2499759 Breva
2500401 Fatón
2500684 Pera
2500812 Laranxa
2501110 Cirola
2501122 Plátano
2501307 Pexego
2501498 Figo
hduser@hadoop-master:~/mr-wordcount-bash$
```

Configura-la orde de saída do mapper coa propiedade *KeyFieldBasedComparator*

KeyFieldBasedComparator

```
-D mapreduce.job.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFieldBasedComparator
```

```
sort -n -r  
-k pos1[,pos2]
```

mapreduce.map.output.key.field.separator

```
$ comparator_class=org.apache.hadoop.mapred.lib.KeyFieldBasedComparator  
$ mapred streaming \  
-D mapreduce.job.output.key.comparator.class=$comparator_class \  
-D mapreduce.partition.keycomparator.options=-nr \  
-files swap_keyval.sh \  
-input mr-wordcount-bash/output \  
-output mr-wordcount-bash/output3 \  
-mapper swap_keyval.sh
```

```
File Output Format Counters  
  Bytes Written=182  
2023-11-11 19:52:40,188 INFO streaming.StreamJob: Output directory: mr-wordcount-bash/output3  
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -head mr-wordcount-bash/output3/part-00000  
2501498 Figo  
2501307 Pexego  
2501122 Plátano  
2501110 Cirola  
2500812 Laranja  
2500684 Pera  
2500401 Fatón  
2499759 Breva  
2499635 Mazá  
2498881 Peladillo  
2497593 Nispero  
2497248 Ameixa  
hduser@hadoop-master:~/mr-wordcount-bash$
```

```
$ mapred streaming \  
-D mapreduce.job.output.key.comparator.class=$comparator_class \  
-D stream.num.map.output.key.fields=2 \  
-D mapreduce.partition.keycomparator.options='-k2r' \  
-files swap_keyval.sh \  
-input mr-wordcount-bash/output \  
-output mr-wordcount-bash/output4 \  
-mapper swap_keyval.sh
```

```

2023-11-11 21:02:09,677 INFO streaming.StreamJob: Output directory: mr-wordcount-bash/output4
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -head mr-wordcount-bash/output4/part-00000
2501122 Plátano
2501307 Pexego
2500684 Pera
2498881 Peladillo
2497593 Nispero
2499635 Mazá
2500812 Laranxa
2501498 Figo
2500401 Fatón
2501110 Cirola
2499759 Breva
2497248 Ameixa
hduser@hadoop-master:~/mr-wordcount-bash$ hdfs dfs -rm -R mr-wordcount-bash/output4

```

-D

mapred-default.xml

mapreduce.task.io.sort.mb

```
-D mapreduce.task.io.sort.mb=512
```

```

$ mapred streaming \
-D mapreduce.job.output.key.comparator.class=$comparator_class \
-D mapreduce.partition.keycomparator.options=-nr \
-D mapreduce.task.io.sort.mb=50 \
-files swap_keyval.sh \
-input mr-wordcount-bash/output \
-output mr-wordcount-bash/output5 \
-mapper swap_keyval.sh

```


-file ***-files***

-p

```
$ hdfs dfs -put -p mapper.py  
$ hdfs dfs -put -p reducer.py
```

Comandos para interactuar cos traballos

-

```
$ $HADOOP_HOME/bin/mapred job -list all
```

-

```
$ $HADOOP_HOME/bin/mapred job -status <JOB-ID>
```

```
$ $HADOOP_HOME/bin/mapred job -status job_201310191043_0004
```

-

```
$ $HADOOP_HOME/bin/mapred job -history <DIR-NAME>
```

```
$ $HADOOP_HOME/bin/mapred job -history /user/hduser/saida
```

-

```
$ $HADOOP_HOME/bin/mapred job -kill <JOB-ID>
```

```
$ $HADOOP_HOME/bin/mapred job -kill job_201310191043_0004
```

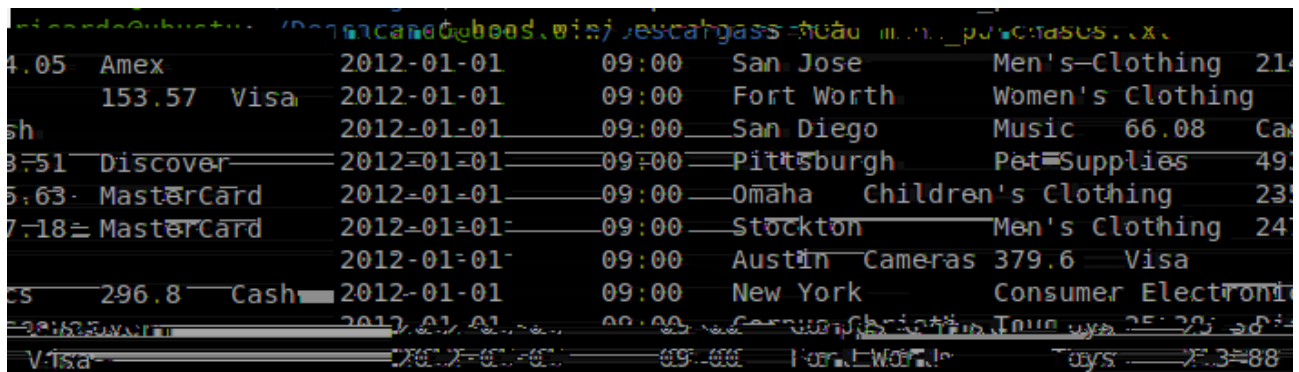
Exercicios MapReduce con Hadoop Streaming

Pautas e recomendacións para realiza-los exercicios

verifica-lo teu código

na túa máquina local

```
$ head -n100 purchases.txt > mini_purchases.txt
```



Amount	Card Type	Date	Time	Location	Category	Amount
4.05	Amex	2012-01-01	09:00	San Jose	Men's Clothing	21.4
153.57	Visa	2012-01-01	09:00	Fort Worth	Women's Clothing	
		2012-01-01	09:00	San Diego	Music	66.08
8.51	Discover	2012-01-01	09:00	Pittsburgh	Pet Supplies	49.1
5.63	MasterCard	2012-01-01	09:00	Omaha	Children's Clothing	23.1
7.18	MasterCard	2012-01-01	09:00	Stockton	Men's Clothing	24.1
		2012-01-01	09:00	Austin	Cameras	379.6
296.8	Cash	2012-01-01	09:00	New York	Consumer Electronics	
		2012-01-01	09:00	New York	Consumer Electronics	25.29
	Visa	2012-01-01	09:00	Fort Worth	Toys	2.38

```
$ cat mini_purchases.txt
```

mapper

pipe

```
$ cat mini_purchases.txt | python mapper.py
```

sort

```
$ cat mini_purchases.txt | python mapper.py | sort
```

reducer

```
$ cat mini_purchases.txt | python mapper.py | sort | python reducer.py
```

python

```
$ chmod +x mapper.py reducer.py
```

```
$ cat mini_purchases.txt | ./mapper.py | sort | ./reducer.py
```

executalo no Clúster Hadoop

```
$ mapred streaming \  
-files mapper.py, reducer.py \  
-input meuCartafoIEntradaHDFS \  
-output meuCartafoISaídaHDFS \  
-mapper scriptMapper \  
-reducer scriptReducer
```

purchases.txt.gz

```
$ ls -lh purchases.txt.gz
$ mkdir vendas
$ ls -lh vendas
$ gzip -l purchases.txt.gz
      compressed      uncompressed   ratio uncompressed_name
      38454568         211312924   81.8% purchases.txt
$ gunzip -k -c purchases.txt.gz > ./vendas/purchases.txt
```

-k

gunzip

-c

```
hduser@hadoop-master:~/Descargas$ ls -lh vendas/
total 0
hduser@hadoop-master:~/Descargas$ ls -lh purchases.txt.gz
-rw-rw-r-- 1 hduser hduser 37M Nov 10 17:20 purchases.txt.gz
hduser@hadoop-master:~/Descargas$ gzip -l purchases.txt.gz
      compressed      uncompressed   ratio uncompressed_name
      38454568         211312924   81.8% purchases.txt
hduser@hadoop-master:~/Descargas$ gunzip -k -c purchases.txt.gz > ./vendas/purchases.txt
hduser@hadoop-master:~/Descargas$ ls -lh vendas/
total 202M
-rw-rw-r-- 1 hduser hduser 202M Nov 10 17:41 purchases.txt
hduser@hadoop-master:~/Descargas$
```

vendas,

purchases.txt

```
$ hdfs dfs -put vendas
```

```
hduser@hadoop-master:~/Descargas$ hdfs dfs -put vendas
hduser@hadoop-master:~/Descargas$ hdfs dfs -ls vendas
Found 1 items
-rw-r--r-- 2 hduser supergroup 211312924 2023-11-10 17:54 vendas/purchases.txt
hduser@hadoop-master:~/Descargas$
```

MapReduce

mapper.py *reducer.py*

```
$ mapred streaming -files mapper.py,reducer.py -input vendas/purchases.txt -
output resultado_consulta -mapper "python mapper.py" -reducer "python
reducer.py"
```

Exercicio 1. Cálculo das vendas totais de cada tenda

purchases.txt

mapper.py reducer.py

1ª parte

mapper.py

```
#!/usr/bin/python
# Format of each line is:
# date\ttime\tstore name\titem description\tcost\tmethod of payment
#
# We want elements 2 (store name) and 4 (cost)
# We need to write them out to standard output, separated by a tab
import sys
for line in sys.stdin:
    data = line.strip().split("\t")
    date, time, store, item, cost, payment = data
    print(store+"\t"+cost)
```

reducer.py

```
#!/usr/bin/python
import sys
salesTotal = 0
oldKey = None
# Loop around the data
# It will be in the format key\tval
# Where key is the store name, val is the sale amount
#
# All the sales for a particular store will be presented,
# then the key will change and we'll be dealing with the next store
for line in sys.stdin:
    data_mapped = line.strip().split("\t")
    if len(data_mapped) != 2:
        # Something has gone wrong. Skip this line.
        continue

    thisKey, thisSale = data_mapped
    # Escribe un par key:value ante un cambio na key
    # Reinicia o total
    if oldKey and oldKey != thisKey:
        print(oldKey+"\t"+str(salesTotal))
        oldKey = thisKey;
        salesTotal = 0
    oldKey = thisKey
    salesTotal += float(thisSale)
# Escribe o ultimo par, unha vez rematado o bucle
if oldKey != None:
    print(oldKey+"\t"+str(salesTotal))
```

mapper reducer

parte2 parte3

mapreduce-vendas

2ª parte

mapper
mapper



mapper

```
for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 6:
        date, time, store, item, cost, payment = data
        print(f'{store}\t{cost}')
```

3ª parte

mapper reducer

purchases.txt

store:cost *mapper* *reducer* *item:cost*
item

```
for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 6:
        date, time, store, item, cost, payment = data
        print(f'{item}\t{cost}')
```

4ª parte

mapper reducer

purchases.txt

mappe payment:cost

```
for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 6:
        date, time, store, item, cost, payment = data
        print(f'{payment}\t{cost}')
```

reducer

payment

```
oldKey = thisKey
if thisSale >= salesMax:
    salesMax = float(thisSale)
```

5ª parte

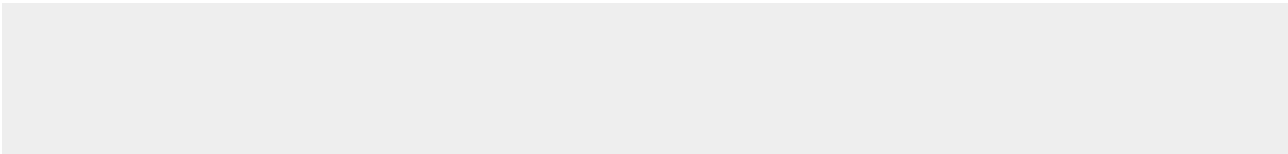
purchases.txt

payment mapper reducer all

```
for line in sys.stdin:
    data = line.strip().split("\t")
    if len(data) == 6:
        date, time, store, item, cost, payment = data
        print(f'all\t{cost}')
```


6ª parte

mapper reducer



Exercicio 2. Cálculo do día máis chuvioso de cada ano

choiva

valores diarios da

- **choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv**

Histórico da estación Vigo-Campus. Vigo. Pontevedra

REALIZAR NOVA CONSULTA

Variables diarias. Periodo de consulta: 13-11-2018 a 13-11-2023

* Datos en horario UTC

Códigos de validación

Resultados en formato CSV

Resultados en formato CSV en columnas

Resultados en formato JSON

Resultados en formato PDF

Código validación	Data*	Código parámetro	Parámetro	Valor	Unidades
1	13-11-2018	PP_SUM_1.5m	Chuvia	0.0	L/m2
1	14-11-2018	PP_SUM_1.5m	Chuvia	0.0	L/m2
1	15-11-2018	PP_SUM_1.5m	Chuvia	0.0	L/m2
1	16-11-2018	PP_SUM_1.5m	Chuvia	0.0	L/m2
1	17-11-2018	PP_SUM_1.5m	Chuvia	0.0	L/m2
1	18-11-2018	PP_SUM_1.5m	Chuvia	3.6	L/m2

-

coma

entre comiñas

dobres ringleira de encabezado

```
"Código validación","Data","Código parámetro","Parámetro","Valor","Unidades"
"1","2018-11-13 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
"1","2018-11-14 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
"1","2018-11-15 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
"1","2018-11-16 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
"1","2018-11-17 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
...
```

Códigos de validación

- 0 - Dato sen validar
- 1 - Dato válido orixinal
- 2 - Dato sospeitoso
- 3 - Dato erróneo
- 5 - Dato válido interpolado
- 9 - Dato non rexistrado

Polo tanto só teremos en conta rexistros co código de validación 1 e 5

Das restantes columnas só nos interesan o trozo de ano da data e a do valor da choiva caída.

map: xerar pares ano-choiva, co ano e o valor de choiva

- `strip()`.
- `,`
- `1 5`
- `print()`

mapperMaxChoiva.py

```
#!/usr/bin/python3
# -*- coding: iso-8859-15 -*-

'''
O formato de cada liña do ficheiro de entrada é:
"Código validación","Data","Código parámetro","Parámetro","Valor","Unidades"

Obxectivo: Obter, a partir do rexistro histórico, a cantidade de choiva diaria
para cada lectura e devolvelo precedido de só o número de ano
Para cada lectura obtemo-los pares <ano, choiva_recollida>

p.e.: 2023 4.2
'''

import sys

# Iterar sobre as liñas de entrada dende sys.stdin
for linha in sys.stdin:

    # Eliminar espazos en branco ó principio e ó final da liña
    linha = linha.strip()

    # Descompo-la liña en campos separados por comas
    codigo, instante_lectura, lixo1, lixo2, choiva, lixo3 = linha.split(",")

    # Verificar se o string de código (eliminando as comiñas) está na lista ["1",
"5"]
    if codigo.strip('"') in ["1", "5"]:

        # Imprimi-lo ano e a cantidade de choiva
        print(instante_lectura[1:5],choiva.strip('"'))
```

```
$ chmod u+x mapperMaxChoiva.py
```

```
$ cat choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv | ./mapperMaxChoiva.py
```

```
2022 0.0  
2022 0.0  
2022 48.3  
2022 30.4  
2022 8.7  
2022 34.1  
2023 54.7  
2023 4.7  
2023 0.0  
2023 0.0  
2023 0.0  
2023 0.0  
2023 38.4  
2023 31.5
```

reducer: calcula o valor máximo das choivas rexistradas en cada ano

-
-
-
-
- *print()*

reducerMaxChoiva.py

```
#!/usr/bin/python3
# -*- coding: iso-8859-15 -*-

'''
O formato de cada liña do ficheiro de entrada é:
- un número enteiro de 4 díxitos correspondente ó ano,
- un espazo en branco e
- un número real correspondente ás choivas en litros/metro cadrado
p.e: 2023 24.6

Obxectivo: calcula-la cantidade de choiva máxima diaria de cada ano a partir do
rexistro histórico
p.e.: 2018 39.0
'''

import sys

# Inicializar variables
ano_actual = None
choiva_maxima_actual = None

# Ler a primeira liña fóra do bucle para evitar comprobar sempre se é
# a primeira liña para cada novo rexistro lido
primeira_linha = sys.stdin.readline()

# Descompo-la primeira liña
ano_actual, choiva_maxima_actual = primeira_linha.strip().split(" ", 1)
choiva_maxima_actual = float(choiva_maxima_actual)

# Iterar sobre as demais liñas de entrada
for linha in sys.stdin:
    ano, choiva_str = linha.strip().split(" ", 1)

    # Converte-la cantidade de choiva a float
```

```

choiva = float(choiva_str)

# Se é o mesmo ano, comprobar se a cantidade de choiva é a máxima
if ano_actual == ano:
    choiva_maxima_actual = max(choiva_maxima_actual, choiva)
else:
    # Se cambia o ano, emitir resultado e actualizar variables
    print("%s\t%s" % (ano_actual, choiva_maxima_actual))
    ano_actual = ano
    choiva_maxima_actual = choiva

# Emiti-lo resultado do último ano
if ano_actual is not None:
    print("%s\t%s" % (ano_actual, choiva_maxima_actual))

```

```
$ chmod u+x reducerMaxChoiva.py
```

```
$ cat choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv | ./mapperMaxChoiva.py
| ./reducerMaxChoiva.py
```

```

r-clima-cuvi-python$ cat choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv | ./mapperMaxChoiva.py | ./reducerMaxChoiva.py
2018    39.0
2019    89.2
2020    70.8
2021    59.5
2022    71.9
2023    63.5

```

reducer

mapper

map-reduce

map

reduce

```
$ cat choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv | ./mapperMaxChoiva.py
| sort -k1,1 | ./reducerMaxChoiva.py
```

Execución en hadoop

```
$ hdfs dfs -mkdir -p mr-clima-cuvi-python/input
$ hdfs dfs -put choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv mr-clima-cuvi-python/input
```

```
$ mapred streaming \
  -files mapperMaxChoiva.py, reducerMaxChoiva.py \
  -input mr-clima-cuvi-python/input \
  -output mr-clima-cuvi-python/output \
  -mapper mapperMaxChoiva.py \
  -reducer reducerMaxChoiva.py
```

```
$ hdfs dfs -cat mr-clima-cuvi-python/output/*
```

Exercicio 3. Cálculo do día máis chuvioso de cada mes, para cada ano

- *choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv*

coma

entre comiñas

dobres

ringleira de encabezado

```
"Código validación","Data","Código parámetro","Parámetro","Valor","Unidades"
"1","2018-11-13 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
"1","2018-11-14 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
"1","2018-11-15 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
"1","2018-11-16 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
"1","2018-11-17 00:00:00.0","PP_SUM_1.5m","Chuvia","0.0","L/m2"
...
```

map: xerar pares periodo-choiva, co ano e mes como chave e a cantidade de choiva rexistrada como valor

mapper

mapperMaxChoivaMensual.py

```
# Imprimi-lo ano-mes e a cantidade de choiva
print(instante_lectura[1:8],choiva.strip(''))
```

```
#!/usr/bin/python3
# -*- coding: iso-8859-15 -*-

'''
O formato de cada liña do ficheiro de entrada é:
"Código validación","Data","Código parámetro","Parámetro","Valor","Unidades"

Obxectivo: Obter, a partir do rexistro histórico, a cantidade de choiva máxima
caída nun día dun mes e devolvelo precedido de só o número de ano - número de
mes
Para cada lectura obtemo-los pares <ano, choiva_recollida>
```



```

p.e: 2023-10 49.7

'''
import sys

# Iterar sobre as liñas de entrada dende sys.stdin
for linha in sys.stdin:

    # Eliminar espazos en branco ó principio e ó final da liña
    linha = linha.strip()

    # Descompo-la liña en campos separados por comas
    codigo, instante_lectura, lixo1, lixo2, choiva, lixo3 = linha.split(",")

    # Verificar se o string de código (eliminando as comiñas) está na lista ["1",
"5"]
    if codigo.strip('') in ["1", "5"]:

        # Imprimi-lo ano-mes e a cantidade de choiva
        print(instante_lectura[1:8],choiva.strip(''))

```

```
$ chmod u+x mapperMaxChoivaMensual.py
```

```
$ cat choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv |
./mapperMaxChoivaMensual.py
```

```

2022-12 0.0
2022-12 0.0
2022-12 48.3
2022-12 30.4
2022-12 8.7
2022-12 34.1
2023-01 54.7
2023-01 4.7
2023-01 0.0
2023-01 0.0
2023-01 0.0
2023-01 0.0
2023-01 38.4
2023-01 31.5

```

reducer: calcula o valor máximo das choivas rexistradas en cada ano

reducerMaxChoivaMensual.py

```
#!/usr/bin/python3
# -*- coding: iso-8859-15 -*-

'''
O formato de cada liña do ficheiro de entrada é:
- unha cadea composta dun número enteiro de 4 díxitos correspondente ó ano_mes,
- un guión,
- un número enteiro de 2 díxitos correspondente ó mes,
- un espazo en branco e
- un número real correspondente ás choivas en litros/metro cadrado
p.e: 2023-11 50.4

Obxectivo: calcula-la cantidade de choiva total mensual de cada ano_mes-mes a
partir do rexistro histórico
'''

import sys

# Inicializar variables
ano_mes_actual = None
choiva_acumulada_actual = None

# Ler a primeira liña fóra do bucle para evitar comprobar sempre se é
# a primeira liña para cada novo rexistro lido
primeira_linha = sys.stdin.readline()

# Descompo-la primeira liña
ano_mes_actual, choiva_acumulada_actual = primeira_linha.strip().split(" ", 1)
choiva_acumulada_actual = float(choiva_acumulada_actual)

# Iterar sobre as demais liñas de entrada
for linha in sys.stdin:
    ano_mes, choiva_str = linha.strip().split(" ", 1)

    # Converte-la cantidade de choiva a float
    choiva = float(choiva_str)

    # Se é o mesmo ano_mes, comprobar se a cantidade de choiva é a máxima
    if ano_mes_actual == ano_mes:
        choiva_acumulada_actual = max(choiva_acumulada_actual, choiva)
    else:
        # Se cambia o ano_mes, emitir resultado e actualizar variables
        print("%s\t%s" % (ano_mes_actual, choiva_acumulada_actual))
```

```
        ano_mes_actual = ano_mes
        choiva_acumulada_actual = choiva

# Emiti-lo resultado do último ano_mes
if ano_mes_actual is not None:
    print("%s\t%s" % (ano_mes_actual, choiva_acumulada_actual))
```

```
$ chmod u+x reducerMaxChoivaMensual.py
```

```
$ cat choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv |  
./mapperMaxChoivaMensual.py | ./reducerMaxChoivaMensual.py
```

```
r-clima-cuvi-python$ cat choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv | ./mapperMaxChoivaMensual.py | ./reducerMaxChoivaMensual.py  
2018-11 33.5  
2018-12 39.0  
2019-01 89.2  
2019-02 19.5  
2019-03 49.3  
2019-04 21.0
```

Execución en hadoop

```
$ hdfs dfs -mkdir -p mr-clima-cuvi-python/input
$ hdfs dfs -put choiva-diaria-CUVI-5-anos-13-11-2018-13-11-2023.csv mr-clima-cuvi-python/input
```

```
$ mapred streaming \
  -files mapperMaxChoivaMensual.py, reducerMaxChoivaMensual.py \
  -input mr-clima-cuvi-python/input \
  -output mr-clima-cuvi-python/output2 \
  -mapper mapperMaxChoivaMensual.py \
  -reducer reducerMaxChoivaMensual.py
```

mapper

```
$ mapred streaming \
  -Dstream.num.map.key.fields=2 \
  -Dmap.output.key.field.separator="-" \
  -files mapperMaxChoivaMensual.py, reducerMaxChoivaMensual.py \
  -input mr-clima-cuvi-python/input \
  -output mr-clima-cuvi-python/output3 \
  -mapper mapperMaxChoivaMensual.py \
  -reducer reducerMaxChoivaMensual.py
```

```
$ hdfs dfs -cat mr-clima-cuvi-python/output3/*
```

Exercicio 4. Cálculo do día máis chuvioso de cada mes, para cada ano (nos últimos 20 anos)

choiva-temperatura-mensual-CUVI-20-anos-11-2006-11-2023.csv

```
"Instante lectura","Chuvia","Chuvia diaria máxima","Temperatura máxima a 1.5m","Temperatura media a 1.5m","Temperatura mínima a 1.5m"
"2006-11-01 00:00:00.0","-9999.0","-9999.0","22.2","13.5","6.9"
"2006-12-01 00:00:00.0","289.2","-9999.0","14.4","9.3","3.2"
"2007-01-01 00:00:00.0","64.1","9.8","16.2","9.1","1.2"
"2007-02-01 00:00:00.0","216.0","37.2","14.1","9.5","4.0"
```

Notas:

-
-

Exercicio 5. Cálculo do día máis caluroso de cada mes, para cada ano (nos últimos 20 anos)

choiva-temperatura-mensual-CUVI-20-anos-11-

2006-11-2023.csv

Exercicio 6

Exercicios propostos

u.item

Movielens

Exemplos código map e código reduce

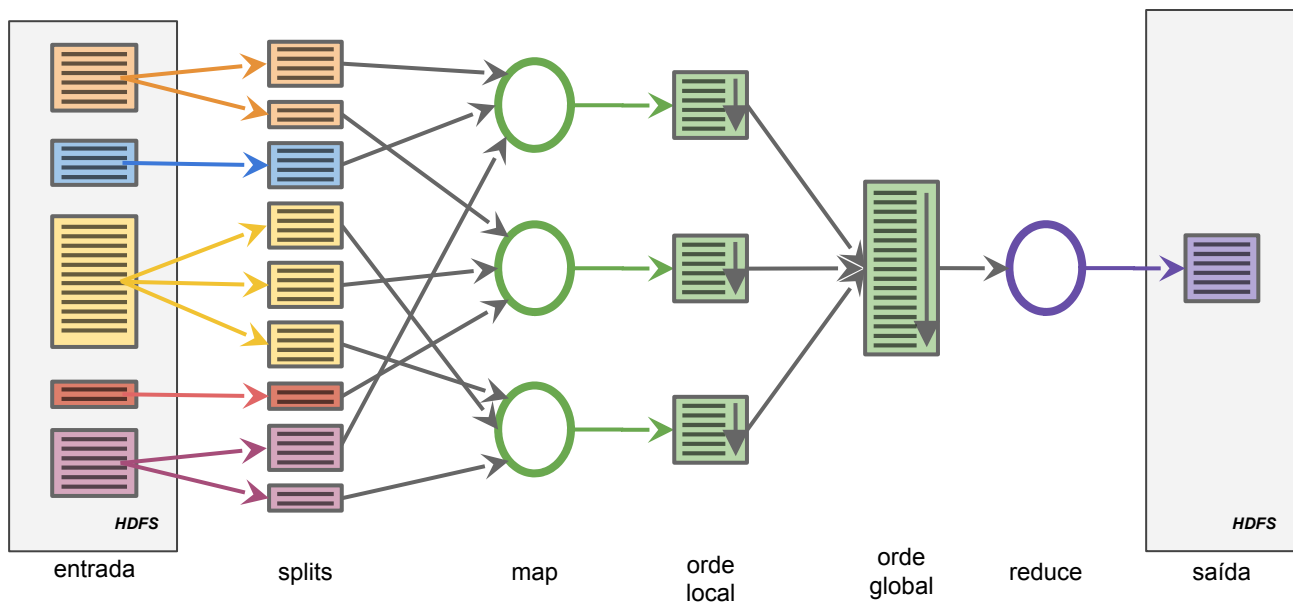
-

-

-

Intro to Hadoop and MapReduce

Execución de trabajos mapreduce



Opciones de ejecución

Número de reduces

-

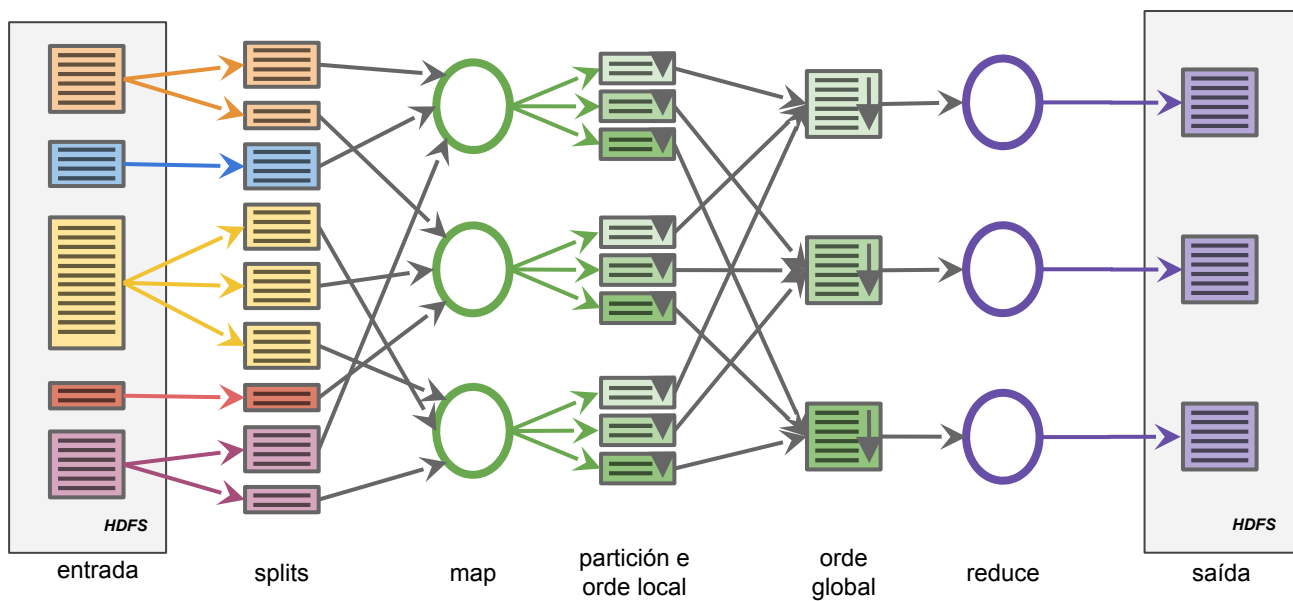
-

-

-

partición

-



-

-

-

-

-

-

-

-

map

-

```
job.setNumReduceTasks(4)
```

-

mapred-site.xml

```
<configuration>
  <property>
    <name>mapreduce.job.reduces</name>
    <value>4</value>
  </property>
</configuration>
```

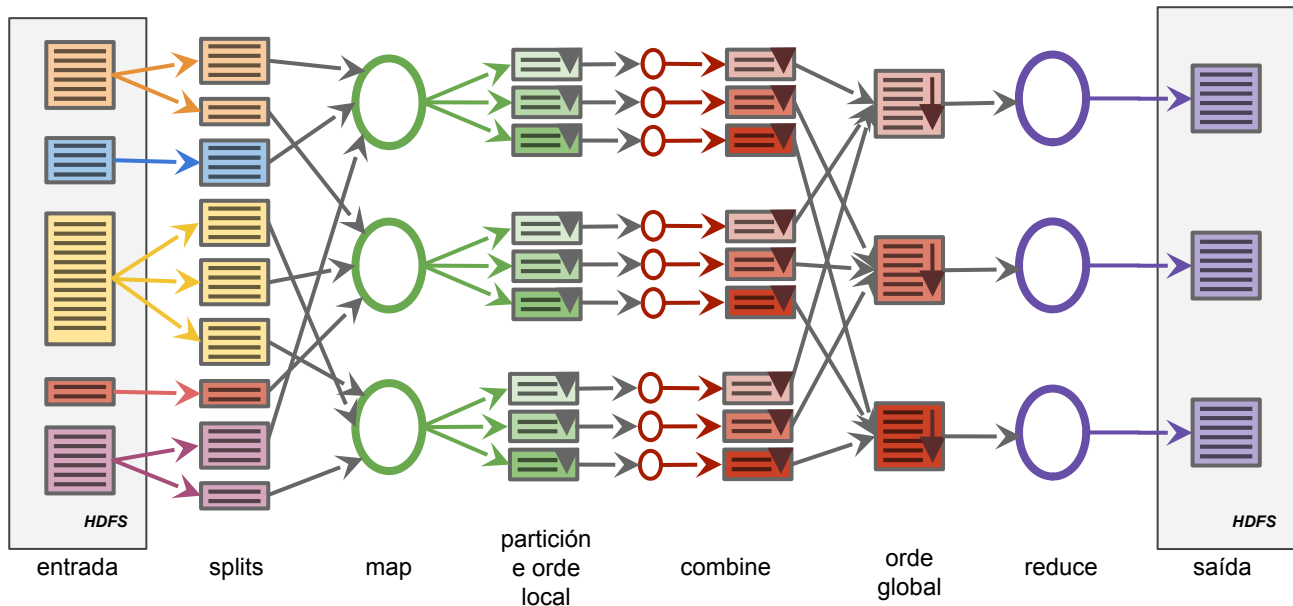
Combiner

-
-
-

map

reducer

```
job.setCombinerClass(WordCountReducer.class);
```



```
$ mapred streaming \
-Dstream.num.map.key.fields=2 \
-Dmap.output.key.field.separator="-" \
-files mapperMaxChoivaMensual.py,reducerMaxChoivaMensual.py \
-input mr-clima-cuvi-python/input \
-output mr-clima-cuvi-python/output2 \
-mapper mapperMaxChoivaMensual.py \
-reducer reducerMaxChoivaMensual.py \
-combiner reducerMaxChoivaMensual.py
```

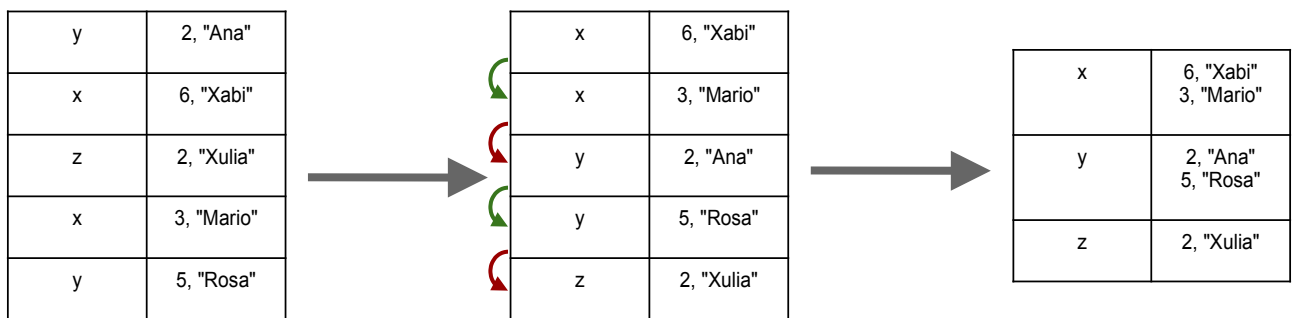
Manexo de datos

-
-
-
-
-
-
-
-

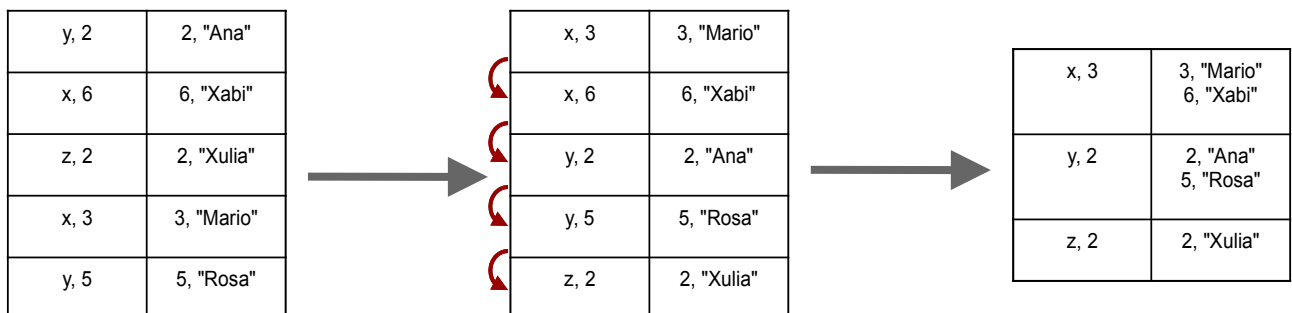
- -
 -
 -
 -
 -
 -
- -
 -
- -

Orde secundario

- -
 -



- -



Contadores

- -

- -
 -
 -
 -
- map*

Funcionalidades comunes

- -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
- org.apache.hadoop.mapreduce.lib*