Introducción a Hadoop

Luis Fernando Llana Díaz

Departamento de Sistemas Informáticos y Computación
Universidad Complutense de Madrid

February 7, 2017

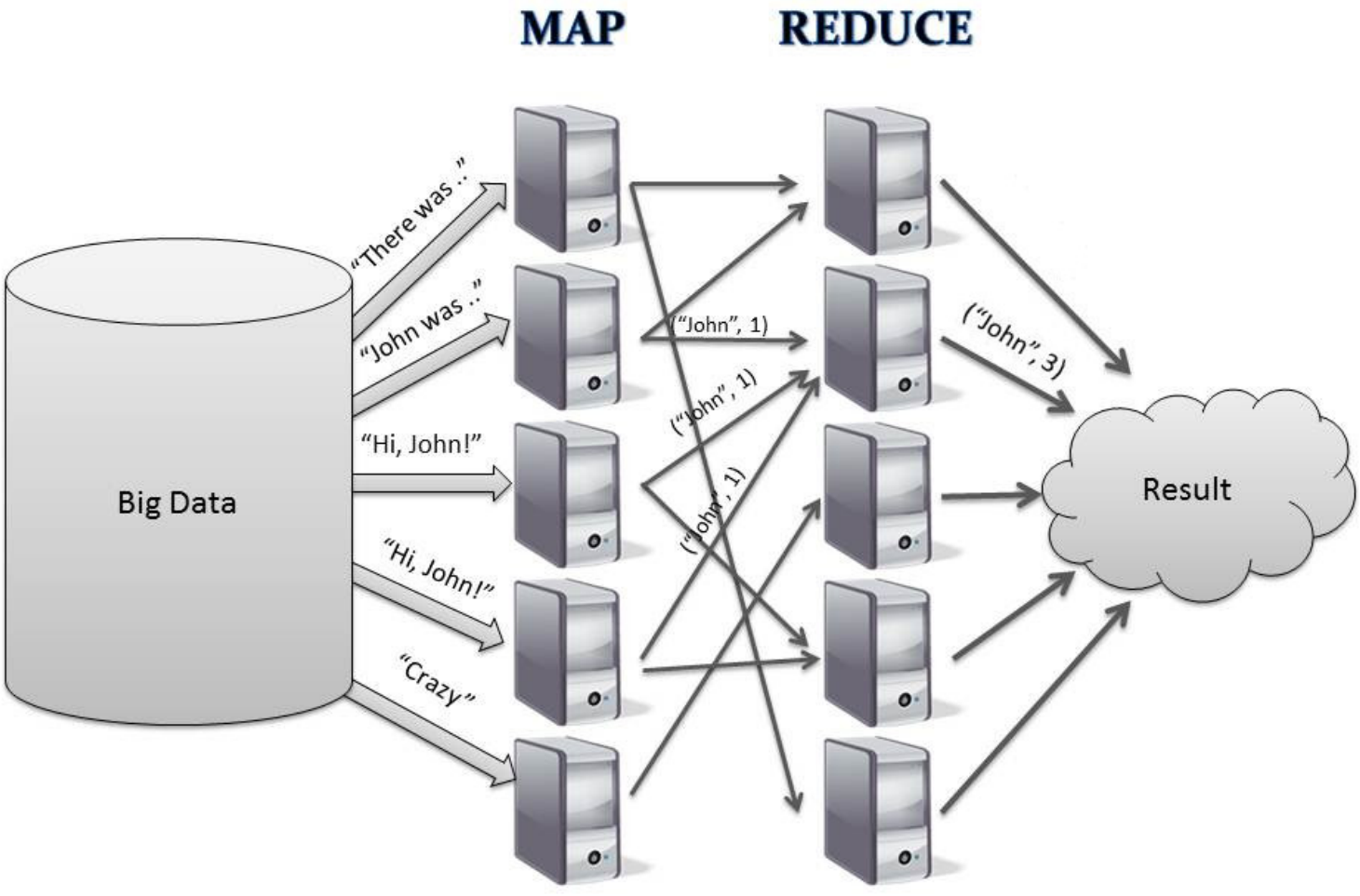
¿Qué es Hadoop?

- Sistema de ficheros hdfs.
  - distribuido
  - redundancia.
- Un Planificador de tareas (yarn).
- Implementación del esquema distribuido MapReduce
- Proyectos relacionados: HBase, Hive, Spark, Tez.

¿Qué no es MapReduce?

- Una base de datos.
- Una herramienta on-line.
- Un lenguaje de programación.

MapReduce


MAP
REDUCE
Big Data
"There was .."
"John was .."
"Hi, John!"
"Hi, John!"
"Crazy"
("John", 1)
("John", 1)
("John", 1)
("John", 3)
Result

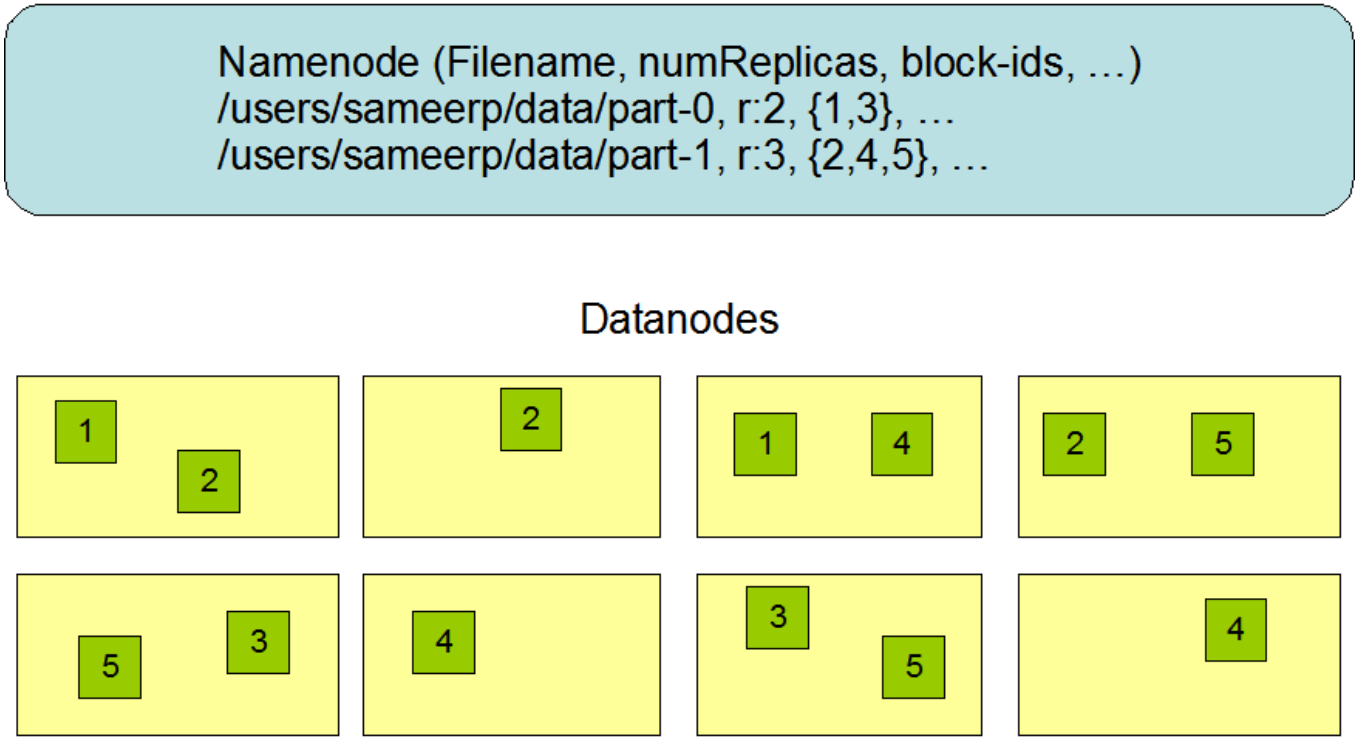
Hadoop Distributed File System

- Distribuido.
- Diseñado para ejecutarse en “commodity hardware”.
- Tolerante a fallos en hardware barato. El fallo es la norma en lugar de la excepción.
- Diseñado para el acceso en forma de “streaming”.
- Diseñado para datos grandes. (block size 64Mb o más)
- “write-once-read-many”. Un fichero que ha sido creado no va a ser modificado.
- Mover la computación es más barato que mover los datos.
- Portabilidad entre plataformas hardware y software.
- Se accede con comandos POSIX.

NameNode y DataNode

NameNode Almacena los metadatados de los ficheros. En hdfs hay 1 NameNode.

DataNode Son los nodos donde se almacenan los datos.



Comandos POSIX

Portable Operating System Interface, UNIX

- Norma escrita IEEE.
- Interfaz para el Sistema Operativo y su entorno.
- Intérprete de comandos.
- ADVERTENCIA: mejor olvidarse de las tildes y espacios en los nombres de los ficheros.

Secure SHell

- Conexiones seguras a máquinas remotas
- Acceso mediante clave privada pública. Las contraseñas no viajan por la red.
- Necesitamos cliente ssh. En linux y windows existe "de serie". En Windows hay que instalarlo (cygwin).
- Hay que generar el par de claves públicas/privadas.

```
$ ssh-keygen -b 2048
```

Genera dos ficheros `id_rsa` (clave privada, no se comparte) e `id_rsa.pub` (clave pública, se puede compartir) en el directorio `.ssh`.

Conexión con el servidor

```
$ ssh [usuario]@dana.estad.ucm.es
```

El usuario se puede omitir si el nombre de usuario es el mismo en el cliente y en el servidor.
Podemos usar comandos POSIX.
Comodines:

- `~` Directorio del usuario. `/home/<usuario>`
- `.` Directorio actual.
- `..` Directorio padre.
- `*` Cualquier secuencia de caracteres.

Copiar datos en el servidor

```
$ scp <fich orig> [usuario]@dana.estad.ucm.es:<fich. destino>
$ scp [usuario]@dana.estad.ucm.es:<fich orig> <fich. destino>
```

Copia ficheros al o desde el servidor.

Opciones:

-R Recursivo (copia subdirectorios).

Ejemplos

```
$ scp -R datos dana.estad.ucm.es:
```

ls

```
ls [opciones] <ficheros>
```

Normalmente se usa para mostrar el contenido de un directorio.
Pero también se puede usar para ver los detalles un fichero concreto
Opciones

- -l Formato largo. Muestra los detalles de los ficheros.
- -a Muestra los ficheros ocultos.
- -t Muestra los ficheros ordenados por tiempo.
- -r Invierte la ordenación.

```
ls ~
ls -l ~
ls -al ~
ls -altr ~
```

cd

```
cd <directorio>
```

Cambiar de directorio de trabajo.

Ejemplos

```
cd ..
cd ~/pruebas
cd ../tmp
```

cp

```
cp [opciones] <fich. origen> <fich. destino>
```

Opciones comunes:

- -R recursivo (copia subdirectorios).
- -v verboso (va indicando que fichero va copiando).

Ejemplos

```
cp patata.txt ../tmp/
cp ~/patata* .
cp patata.txt patata_v1.txt
cp -Rv pruebas/ /var/lib/
```

cat

```
cp <ficheros>
```

Muestra el contenido de ficheros.

Ejemplos

```
cat ~/.bashrc
cat datos/20150915.txt
cat datos/20150915.txt | less
```

mv

```
mv <ficheros orig> <fichero dest>
```

Se usa para dos cosas

- Cambiar de nombre un fichero.
- Mover ficheros a otro directorio

Ejemplos

```
mv * old/
mv patata.txt patata_v1.txt
```

rm

Borra ficheros

Opciones comunes

- -r Recursivo
- -f Fuerza el borrado
- -v Verboso

Ejemplos

```
rm -rfv /
rm -r datos
rm ~/patata.txt
```

df

Muestra el uso de los discos conectados al sistema
Opciones

-h Muestra el tamaño en Megas, Gigas..(human readable)

du

Muestra lo que ocupa un fichero o directorio
Opciones

- -s No muestra los subdirectorios.
- -h Muestra el tamaño en Megas, Gigas.. (human readable)

Comandos hdfs

```
hadoop fs -<comando> <argumentos>
```

Comandos

put Pone un fichero en hdfs

```
hadoop fs -put books /user/luis
```

rm Borra un fichero o directorio.

```
hadoop fs -rm -r books /user/luis/books
```

ls Muestra el contenido

```
hadoop fs -ls /user/master16/books
```

Lista completa: `https://hadoop.apache.org/docs/r1.0.4/file_system_shell.html`

Ejecución MapReduce con python

Programa count.py

```
from mrjob.job import MRJob

class MRCharCount(MRJob):

    def mapper(self, _, line):
        yield "chars", len(line)
        yield "words", len(line.split())
        yield "lines", 1

    def reducer(self, key, values):
        yield key, sum(values)


if __name__ == '__main__':
    MRCharCount.run()
```

```
python count.py quijote-sample.txt #óEjecucin local
python count.py -r hadoop hdfs:///user/luis/quijote.txt #óEjecucin
```

Parámetros importantes

Numéro de reducers Se establece por el tamaño de bloque. Se establece de forma automática.

Tamaño de bloque: 64Mb

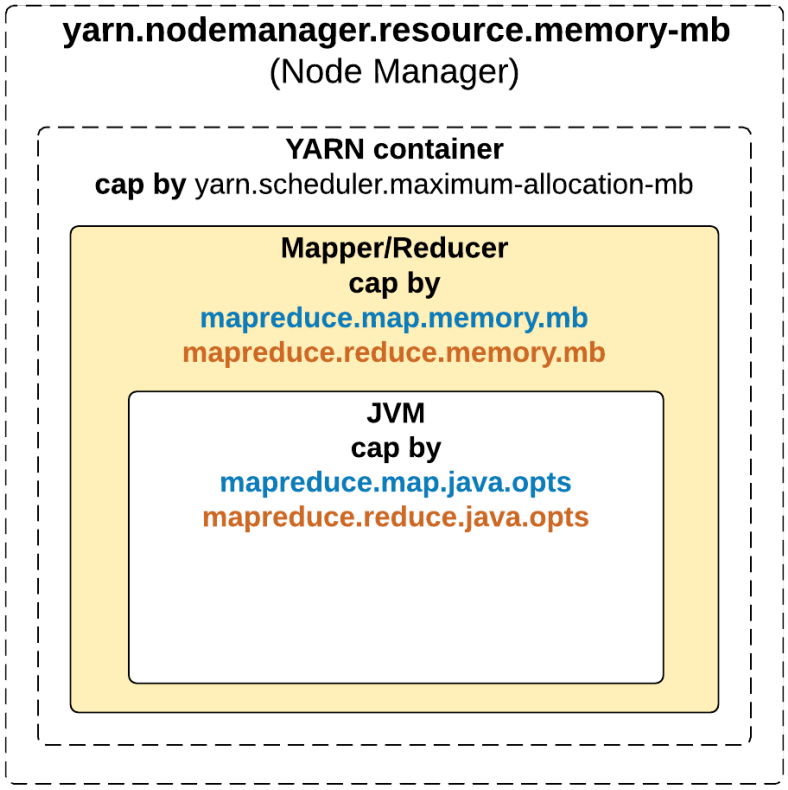
```
hadoop fs -ls /user/hadoop/cristobal/ikea/ventas4C.txt
-rw-r--r--   2 hadoop supergroup 3374349083 2016-05-09
```

El fichero son 3.3Gb: 51 tareas de reducers.

Número de reducers Se establece manualmente. Depende del problema.

```
python emparejarTiposMR_v5.py --jobconf mapreduce.job.
```

Control de memoria




```
--jobconf mapreduce.map.memory.mb=1700
--jobconf mapreduce.map.java.opts="-Xmx1600M␣␣-XX:+UseSerialGC"
--jobconf mapreduce.reduce.memory.mb=1700
--jobconf mapreduce.reduce.java.opts="-Xmx1600M␣␣-XX:+UseSerialGC"
```

Ejemplo Real

```
python emparejarTiposMR_v5.py --jobconf mapreduce.job.reduces=4 \
 --jobconf mapreduce.map.memory.mb=1700 \
 --jobconf mapreduce.reduce.memory.mb=1600  \
 --jobconf mapreduce.reduce.java.opts="-Xmx1500M␣␣-XX:+UseSerialGC
 --jobconf mapreduce.map.java.opts="-Xmx1600M␣␣-XX:+UseSerialGC" \
 --jobconf mapreduce.task.io.sort.mb=1024 --numMin=100 \
 -r hadoop \
 hdfs:///user/hadoop/cristobal/ikea/ventas4C.txt > tipos_100.txt
```

Número total de mappers 51

Número de reducers 4

Tiempo medio de los mappers 50m

Tiempo medio de shuffle 3h 24m

Tiempo medio reducers 1h 10m

Tiempo de ejecución 6h

Control de ejecución

Es necesario usar OpenVPN (`http://openvpn.net`) y un fichero de configuración proporcionado por nosotros.

En tiempo de ejecución : `http://192.168.8.1:8088`

Histórico : `http://192.168.8.1:19888`

Resumen de pasos

1. Copiar los datos y el programa al sistema de ficheros normal del servidor

```
$ scp quijote.txt count.py dana.estad.ucm.es:quijote/
```

2. Nos conecamos al servidor

```
$ ssh dana.estad.ucm.es
$ cd quijote
```

3. Probar el program en *local* (no hadopp).

```
$ python count.py quijote.txt
$ time python count.py quijote.txt
```

4. Subir los datos a hadoop

```
$ hadoop fs -mkdir quijote
$ hadoop fs -put quijote.txt quijote/
```

5. Ejecutar el programa en hadoop

```
$ python count.py quijote.txt -r hadoop
```

6. Monitorizar `http://192.168.8.1:8088` y `http://192.168.8.1:19888`