

Práctica 5 - Bases de Datos 2

Hayk Kocharyan
757715@unizar.es

Juan José Tambo Tambo
755742@unizar.es

Pedro Tamargo Allué
758267@unizar.es

Jesús Villacampa Sagaste
755739@unizar.es

7 de junio de 2020

Índice

1. Esfuerzos invertidos	1
2. Instalación de Hadoop y HBase	1
2.1. Apache Hadoop	1
2.2. Apache HBase	3
3. Pseudo-código de las funciones map y reduce utilizadas	4
4. Más cosas necesarias	4
5. Conclusiones sobre Hadoop y HBase	4
6. Anexo 1: Figuras	5

Índice de figuras

1. Pseudocódigo proporcionado en el enunciado de la práctica	5
--	---

1. Esfuerzos invertidos

- Hayk:
 - Instalación de Hadoop y HBase: 4 horas
 - Implementación del código de Logistica.java: 3 horas
 - Implementación del código de Validacion.java: 1 hora
 - Memoria: 2 horas
- Juan José:
 - Implementación del código de Logistica.java: 3 horas
 - Implementación del código de Validacion.java: 1 hora
- Pedro:
 - Instalación de Hadoop y HBase: 4 horas
 - Implementación del código de Logistica.java: 3 horas
 - Depuración de código de Logistica.java: 3 horas
 - Implementación del código de Validacion.java: 1 hora
 - Memoria: 3 horas
- Jesús:
 - Implementación del código de Logistica.java: 3 horas
 - Implementación del código de Validacion.java: 1 hora
 - Memoria: 1 hora

2. Instalación de Hadoop y HBase

Para la instalación de *Hadoop* y *HBase* se han seguido los tutoriales disponibles en sus respectivas páginas oficiales. Se va a utilizar una máquina con *SO Ubuntu 18.04*.

En el caso de *Apache Hadoop* se ha procedido a configurar un servidor ssh para acceder a la propia máquina sin contraseña, utilizando una clave pública. Se puede instalar el servidor utilizando la orden:

```
sudo apt update && sudo apt -y install openssh-server
```

Se va a proceder a crear e instalar una clave *RSA* pública en nuestra máquina con el comando:

```
cd ~/.ssh          # Nos situamos en el directorio .ssh
ssh-keygen          # Creación de los ficheros de clave pública y privada
cat <id_rsa.pub >>authorized_keys
```

Tras esta configuración podremos acceder vía *SSH* a la máquina sin contraseña mediante el comando:

```
ssh localhost
```

2.1. Apache Hadoop

Después de la configuración técnica de la máquina se va a proceder a instalar *Hadoop 3.2.1*. Se ejecutarán los siguientes comandos:

```
wget http://apache.uvigo.es/hadoop/common/hadoop-3.2.1/ \
  hadoop-3.2.1.tar.gz
tar -xzf hadoop-3.2.1.tar.gz
# Exportamos al PATH
export PATH="$PATH:$PWD/hadoop-3.2.1/bin:$PWD/hadoop-3.2.1/sbin"
cd hadoop-3.2.1
```

Una vez se han descomprimido los ficheros se va a proceder a modificar los ficheros de configuración. Estos ficheros se encuentran en el directorio *./etc/hadoop/*. En el fichero *hadoop-env.sh* se va a establecer la variable de entorno *JAVA_HOME*, para ello se insertarán las siguientes líneas:

```
# set to the root of your Java installation
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
# Establecer el directorio raíz de Java
```

Se van a proceder a modificar los ficheros *core-site.xml* y *hdfs-site.xml*. En el primero, se va a modificar el contenido por el siguiente:

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

En el segundo fichero se va a modificar su contenido por el siguiente:

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

Tras la configuración de los ficheros se va a proceder a formatear el sistema de ficheros, para ello se va a proceder a ejecutar el siguiente comando:

```
hdfs namenode -format
```

Tras esto, se habrá instalado *Apache Hadoop*, no obstante faltará crear al usuario que interactuará con el sistema de ficheros distribuidos. Ejecutaremos los siguientes comandos:

```
start-dfs.sh
hdfs dfs -mkdir -p /user/$LOGNAME
```

Para la configuración de *YARN*, se va a proceder a parar todos los servicios (*stop-all.sh*) y a modificar los ficheros de configuración situados en *./etc/hadoop/*. En el fichero *mapred-site.xml* se establecerá el siguiente contenido:

```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
  <property>
    <name>mapreduce.application.classpath</name>
    <value>${HADOOP_MAPRED_HOME}/share/hadoop/mapreduce/*: \
```

```
        $HADOOP_MAPRED_HOME/share/hadoop/mapreduce/lib/*</value>
    </property>
</configuration>
```

Y en el fichero *yarn-site.xml* se establecerá el siguiente contenido:

```
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.env-whitelist</name>
    <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME, \
        HADOOP_CONF_DIR, CLASSPATH_PREPEND_DISTCACHE, \
        HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
  </property>
</configuration>
```

Se va a proceder a iniciar *Hadoop* de nuevo, para ello se va a utilizar el siguiente comando:

```
start-all.sh
```

2.2. Apache HBase

Para la instalación de *Apache HBase 1.4.13* se va a proceder a descargar los binarios desde su página oficial utilizando los comandos:

```
wget http://apache.uvigo.es/hbase/1.4.13/hbase-1.4.13-bin.tar.gz
tar -xzvf hbase-1.4.13-bin.tar.gz
# Exportamos la carpeta al path
export PATH="$PATH:$PWD/hbase-1.4.13/bin"
cd hbase-1.4.13
```

Lo primero se va a proceder a modificar los ficheros de configuración situados en *./conf*. En primer lugar modificaremos el fichero *hbase-env.sh*, añadiremos las siguientes líneas:

```
# Set environment variables here.
# The java implementation to use.
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
# Establecer el directorio raíz de Java
```

Ahora se va a proceder a modificar el fichero *hbase-site.xml* modificando su contenido y añadiendo lo siguiente:

```
<property>
  <name>hbase.cluster.distributed</name>
  <value>true</value>
</property>
<property>
  <name>hbase.rootdir</name>
  <value>hdfs://localhost:9000/hbase</value>
</property>
```

Tras esto, se va a proceder a ejecutar *HBase*, para ello (con *Hadoop* iniciado) se utilizará el comando:

```
start-hbase.sh
```

Si se quiere acceder a hbase utilizando un prompt interactivo se puede utilizar el comando:

```
hbase shell
```

3. Pseudo-código de las funciones map y reduce utilizadas

Para la adaptación del algoritmo de descenso de gradiente en *Hadoop* se ha dividido el algoritmo proporcionado en el enunciado (Figura 1) en uno equivalente utilizando las funcionalidades de *Map-Reduce* disponibles en *Hadoop*.

La función *map* se encargaría de emitir para cada clave j el valor de cada uno de sus n sumandos. De esta manera la función *reduce* se encargará de realizar ese sumatorio de n sumandos y será una vez se haya completado la tarea de cálculo del vector gradiente cuando se hará el ajuste de los parámetros θ_{act} .

De esta manera el pseudocódigo de las funciones mencionadas anteriormente quedará de la siguiente manera:

```
function map(key:clave1, value:valor1) {
    // clave1 se corresponde con el identificador del cliente a evaluar.
    // valor1 contiene toda la información asociada al cliente.
    // Hallamos el producto escalar
    var prod_Esc = thetasAct * value[caracteristicas]
    // Calculamos yi en función de value[cliclass] es A o B
    var yi = if (value[cliclass] == A) then 0 else 1
    // Para cada característica (índice, valor) de
    // value[caracteristicas] emitimos
    foreach (j,valor) in value[caracteristicas] do
        emit(j, (yi - g(prod_Esc))*valor )
    end
}
```

```
function reduce(key:clave2, value:valor2) {
    // clave2 se corresponde con el identificador de
    // la característica a evaluar.
    // valor2 contiene la lista de sumandos de la característica j.
    // Hallamos el sumatorio
    emit(key, sum(value))
}
```

4. Más cosas necesarias

5. Conclusiones sobre Hadoop y HBase

6. Anexo 1: Figuras

```
1  $\theta_{act} = (0, 0, \dots, 0)$ 
2 while (no se cumpla criterio de convergencia) {
3   //vector de 1601 posiciones,  $grad_j$  es la posición  $j$  de  $grad$ 
4   grad =  $(0, 0, \dots, 0)$ 
5   for  $j \leftarrow 0$  to 1600 {
6      $grad_j = \sum_{i=1}^n (y^i - g(\theta_{act} \cdot \mathbf{x}^i)) * x_j^i$ 
7   }
8    $\theta_{act} = \theta_{act} + \alpha * \mathbf{grad}$ 
9 }
```

Figura 1: Pseudocódigo proporcionado en el enunciado de la práctica