Máster en Big Data y Data Science:
Ciencia e Ingeniería de Datos

UNIVERSIDAD AUTONOMA DE MADRID

Infraestructura para Big Data: Benchmarking de un clúster Hadoop

Utilización de un clúster Big Data Cloudera

Resultados de la práctica

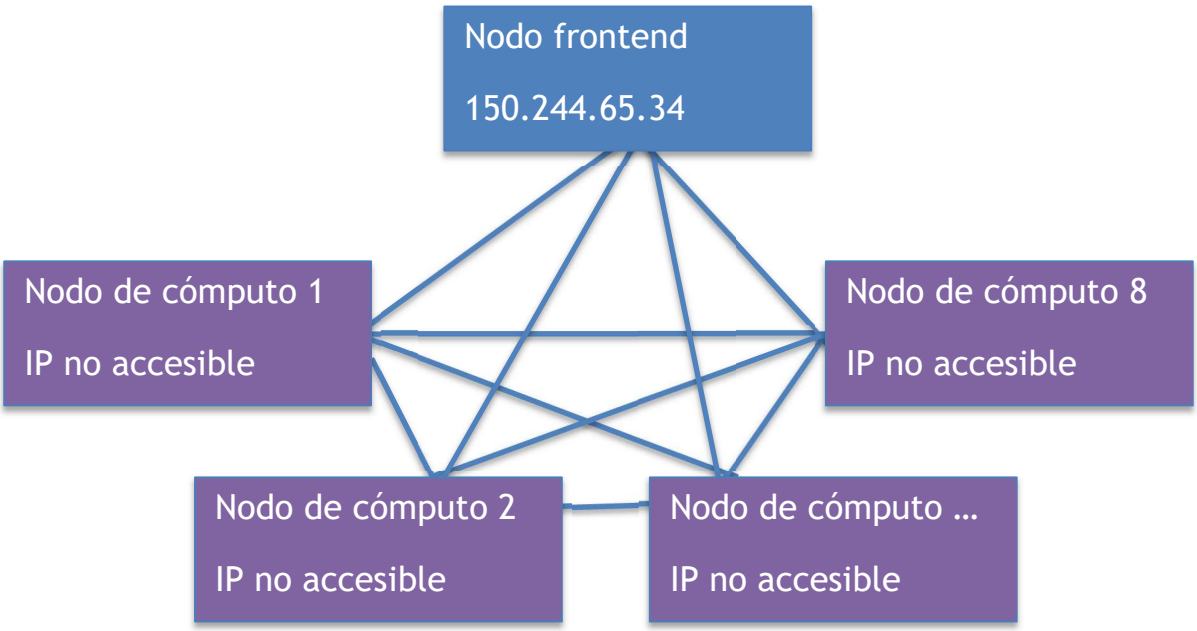
El desarrollo de esta práctica se llevará a cabo en horario de clase de la asignatura. Al terminar, los alumnos deberán entregar un archivo pdf a través de Moodle, con las respuestas a las preguntas y ejercicios que se plantean a lo largo del enunciado de la práctica.

Si algún alumno no puede asistir a la clase, o no le diera tiempo a completar las tareas, podrá terminar el trabajo desde casa, ya que el clúster utilizado permite el acceso desde fuera de la red de la Universidad Autónoma de Madrid.

Introducción: uso del clúster Hadoop de la asignatura

Todos los alumnos utilizarán un clúster Hadoop que se ha instalado y hecho accesible al menos durante la duración de la asignatura. A diferencia de los sistemas utilizados en curso hasta ahora, el clúster Hadoop será compartido por múltiples usuarios.

La configuración hardware de nuestro clúster es la siguiente:



Cada uno de los 8 nodos de cómputo cuenta con las siguientes características:

- 8 GB de memoria RAM
- Procesador Intel Core i7 3770, a 3,4GHz
- 100GB de disco duro para SO
- 360 GB de disco añadido al HDFS
- SO CentOS 7.3 64 bit

Para el lanzamiento de tareas, **se han creado usuarios para cada uno de los alumnos en la máquina frontend del clúster**. Para acceder a dicha máquina, bastará con ejecutar desde una terminal de Linux el comando:

```
>ssh uambdXX@150.244.65.34
```

Donde XX es el número de usuario asignado a cada uno. El reparto de usuarios se habrá realizado en clase durante las horas asignadas a esta

parte de la asignatura.

Nota: el clúster de la asignatura es accesible desde direcciones externas a la Universidad Autónoma, de modo que los alumnos podrán lanzar tareas desde casa.

A diferencia de en los sistemas Big Data de pruebas utilizados hasta el momento, una vez conectados al nodo del clúster no es necesario arrancar los distintos servicios, ya que deberían arrancarse al iniciar los equipos que conforman el clúster.

Ejecutar una tarea de prueba: WordCount

En primer lugar, hemos de copiar el fichero que utilizaremos como entrada a nuestro clúster. Para ello, utilizaremos los ficheros que se proveen con la práctica:

```
>./creaFichero.bash 40
```

Este script nos genera un fichero de texto "veryBig.txt" que es el fichero "big.txt" replicado tantas veces como le indiquemos.

```
>scp veryBig.txt uambdXX@150.244.65.34:
```

Con este comando **copiamos el fichero de prueba** (accesible en el moodle de la asignatura) al directorio "home" de nuestro usuario del clúster /home/uambdXX).

Una vez copiado el fichero al clúster, *subimos* su contenido al sistema de ficheros distribuido (HDFS):

```
>hdfs dfs -put veryBig.txt

>hdfs dfs -ls
```

En este punto, ya tenemos nuestro fichero de prueba copiado en nuestro sistema de ficheros distribuido. Ahora procederemos a ejecutar la aplicación WordCount sobre nuestro fichero de ejemplo:

```
>hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-examples.jar wordcount veryBig.txt salida-veryBig/
```

Tras la ejecución, comprobamos la salida generada:

```
>hdfs dfs -ls salida-veryBig

Found 25 items
```

```
-rw-r--r--   3 uambd34 uambd34          0 2017-10-13 20:42 salida-veryBig/_SUCCESS
-rw-r--r--   3 uambd34 uambd34      42260 2017-10-13 20:42 salida-veryBig/part-r-00000
-rw-r--r--   3 uambd34 uambd34      43141 2017-10-13 20:42 salida-veryBig/part-r-00001
-rw-r--r--   3 uambd34 uambd34      42561 2017-10-13 20:42 salida-veryBig/part-r-00002
-rw-r--r--   3 uambd34 uambd34      42745 2017-10-13 20:42 salida-veryBig/part-r-00003
-rw-r--r--   3 uambd34 uambd34      43587 2017-10-13 20:42 salida-veryBig/part-r-00004
-rw-r--r--   3 uambd34 uambd34      42803 2017-10-13 20:42 salida-veryBig/part-r-00005
-rw-r--r--   3 uambd34 uambd34      43396 2017-10-13 20:42 salida-veryBig/part-r-00006
-rw-r--r--   3 uambd34 uambd34      42785 2017-10-13 20:42 salida-veryBig/part-r-00007
-rw-r--r--   3 uambd34 uambd34      44007 2017-10-13 20:42 salida-veryBig/part-r-00008
-rw-r--r--   3 uambd34 uambd34      42875 2017-10-13 20:42 salida-veryBig/part-r-00009
-rw-r--r--   3 uambd34 uambd34      43225 2017-10-13 20:42 salida-veryBig/part-r-00010
-rw-r--r--   3 uambd34 uambd34      43144 2017-10-13 20:42 salida-veryBig/part-r-00011
-rw-r--r--   3 uambd34 uambd34      44493 2017-10-13 20:42 salida-veryBig/part-r-00012
-rw-r--r--   3 uambd34 uambd34      42967 2017-10-13 20:42 salida-veryBig/part-r-00013
-rw-r--r--   3 uambd34 uambd34      44323 2017-10-13 20:42 salida-veryBig/part-r-00014
-rw-r--r--   3 uambd34 uambd34      42161 2017-10-13 20:42 salida-veryBig/part-r-00015
-rw-r--r--   3 uambd34 uambd34      44410 2017-10-13 20:42 salida-veryBig/part-r-00016
```

```
-rw-r--r--   3 uambd34 uambd34      43817 2017-10-13 20:42 salida-veryBig/part-r-00017
-rw-r--r--   3 uambd34 uambd34      43437 2017-10-13 20:42 salida-veryBig/part-r-00018
-rw-r--r--   3 uambd34 uambd34      44819 2017-10-13 20:42 salida-veryBig/part-r-00019
-rw-r--r--   3 uambd34 uambd34      43878 2017-10-13 20:42 salida-veryBig/part-r-00020
-rw-r--r--   3 uambd34 uambd34      43697 2017-10-13 20:42 salida-veryBig/part-r-00021
-rw-r--r--   3 uambd34 uambd34      43229 2017-10-13 20:42 salida-veryBig/part-r-00022
-rw-r--r--   3 uambd34 uambd34      41099 2017-10-13 20:42 salida-veryBig/part-r-00023
```

Pregunta: ¿Cómo justificarías que el fichero de salida esté partido en varias partes (24 en el ejemplo)?

En un rato veremos como monitorizar de forma sencilla con HUE el estado de nuestra tarea MapReduce en el clúster.

¿Qué aplicaciones de ejemplo existen disponibles?

Para comprobar qué aplicaciones soporta el *jar* con los ejemplos de nuestra distribución de Hadoop, simplemente hay que ejecutarlo sin argumentos:

```
>hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-examples.jar
```

Esto nos retornará un listado de las aplicaciones implementadas:

```
An example program must be given as the first argument.
Valid program names are:
  aggregatewordcount: An Aggregate based map/reduce program that counts the words in the input files.
  aggregatewordhist: An Aggregate based map/reduce program that computes the histogram of the words in the input files.
  ...
```

grep: A map/reduce program that counts the matches of a regex in the input.

join: A job that effects a join over sorted, equally partitioned datasets

multifilewc: A job that counts words from several files.

…

pi: A map/reduce program that estimates Pi using a quasi-Monte Carlo method.

…

sort: A map/reduce program that sorts the data written by the random writer.

sudoku: A sudoku solver.

teragen: Generate data for the terasort

terasort: Run the terasort

teravalidate: Checking results of terasort

wordcount: A map/reduce program that counts the words in the input files.

…

Si queremos más información sobre los parámetros de E/S de una aplicación, la ejecutamos sin argumentos:

```
>hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-examples.jar wordcount

Usage: wordcount <in> [<in>...] <out>
```

a) Usar la línea de comandos e invocar la *shell* usando el formato:
   >**hdfs dfs <args>**
   >hdfs dfs –ls /

   >hdfs dfs –mkdir myTestDir

b) También se puede manipular HDFS usando HUE.

Importante recordar:

- Rutas relativas (por defecto) y absolutas: /user/uambdXX.
- HDFS **no** es un sistema de fichero POSIX: **no** podemos hacer todo lo que haríamos sobre un FS normal. Sí podemos hacer ciertas cosas aplicando *pipes* a la salida del comando sobre HDFS, pero no se ejecutan de forma distribuida.
- Los ficheros en HDFS se almacenan de forma distribuida: partición en bloques, replicación, …
- Los comandos **hadoop fs** y **hdfs dfs** son equivalentes (los encontraréis de forma indistinta en la documentación), aunque se está "generalizando" la utilización del segundo.

Acceso a la consola de Cloudera Manager de nuestro clúster

Hasta ahora hemos podido ejecutar una tarea en nuestro clúster Hadoop utilizando la línea de comandos, tal y como podríamos hacer en cualquier clúster Hadoop independientemente de la distribución sobre la que se ha montado (BigInsights, Cloudera, …).

Ahora, vamos a aprovechar una de las funcionalidades que la distribución de Cloudera nos ofrece: la interfaz web de *Cloudera Manager*. Para acceder a dicha interfaz web, tecleamos en la barra de direcciones de nuestro navegador:

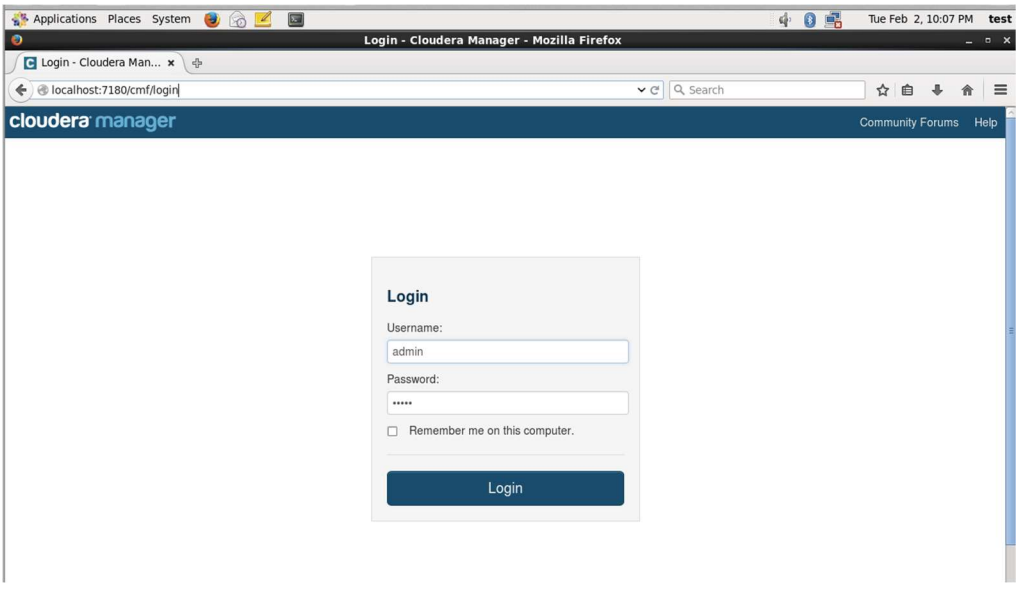
http://<direccion-IP-frontend>:7180

Que en nuestro caso se traduce en:

http://150.244.65.34:7180

Una vez cargue la interfaz web, se nos solicitará para el acceso un usuario y una contraseña. Los usuarios y contraseñas son los mismos que para el acceso al nodo frontend del clúster.

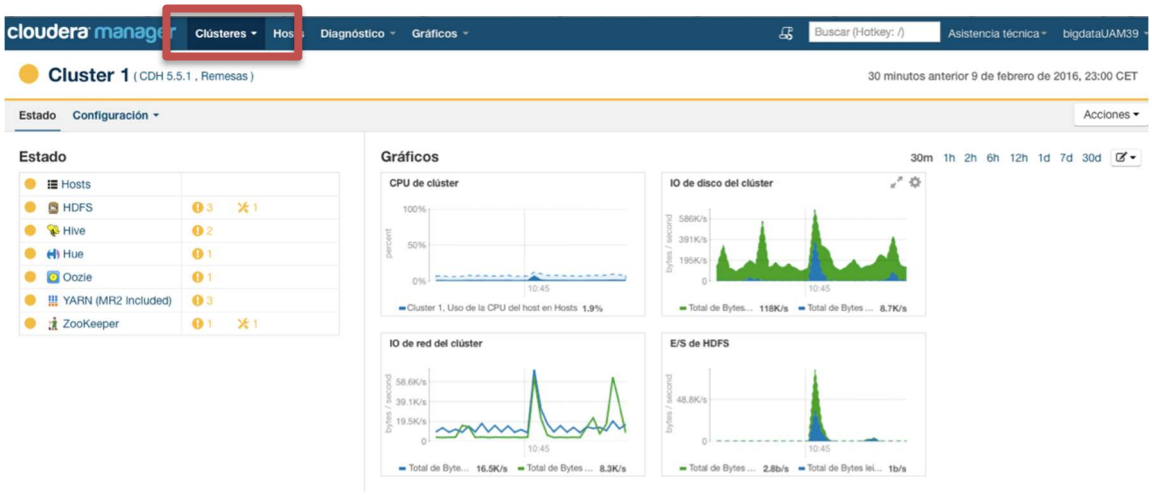
Nota: en este caso los usuarios y contraseñas para frontend y Cloudera Manager (así como para HUE) coinciden entre sí. Esto es debido a que se han creado adrede de esta forma, pero realmente son sesiones independientes. Por tanto, **un cambio en la contraseña de usuario en uno de estos tres entornos no supondrá un cambio los otros dos entornos.**



Una vez hemos logrado iniciar sesión en la interfaz de Cloudera Manager, podemos acceder a la configuración/consulta del estado tanto del hardware como del software del clúster.

Los usuarios creados para los alumnos no tienen permisos de administración, por lo que sólo podrán consultar el estado del clúster, y no modificar su configuración. Este tipo de usuario también permite al dueño de un trabajo su parada y eliminación, si así lo deseara.

Por ejemplo, vamos a consultar el estado de nuestro clúster, pinchando en el desplegable "Clústers" situado a la izquierda en la barra superior de la interfaz:



Por ejemplo, es esta captura de la interfaz observamos que se ha producido un pico de actividad en el clúster debido a la tarea "WordCount" que hemos lanzado previamente.

Ejercicio: utilizando la interfaz de web de Cloudera Manager, acceda a la siguiente información de configuración del clúster Hadoop:

- ¿Cuál es el tamaño de bloque configurado para el HDFS del clúster?
- ¿Cuál es el factor de replicación por defecto del HDFS?
- ¿Cuál es el número de tareas MapReduce que se lanzarán por defecto al crear un nuevo trabajo? ¿Y para una tarea lanzada desde Hive?

Utilizando HUE para gestionar nuestra actividad en el clúster

HUE (Hadoop User Experience) es un servicio que aglomera en una única interfaz las acciones más comunes realizadas al trabajar con un clúster Hadoop:

- Editor de Queries: Hive, Pig, Impala (si instalado), …
- Creación y gestión de flujos de trabajo: Oozie
- Interfaz amigable (navegador de ficheros) para el HDFS
- Consulta y gestión de los trabajos activos

Para acceder a la interfaz web de HUE, simplemente tecleamos en la barra de direcciones de nuestro navegador:

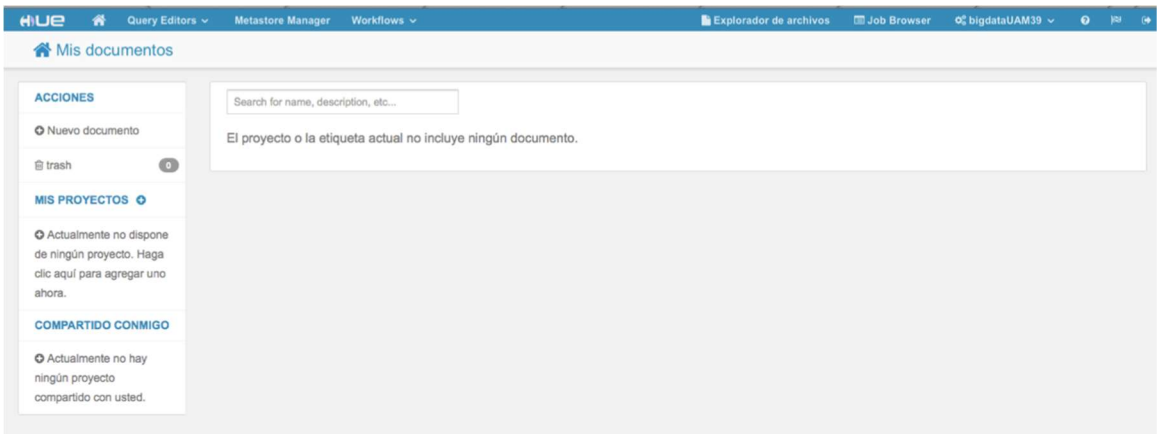
http://<dirección -IP-nodo-HUE>:8888

Que en nuestro caso se traduce en:

http://150.244.65.34:8888

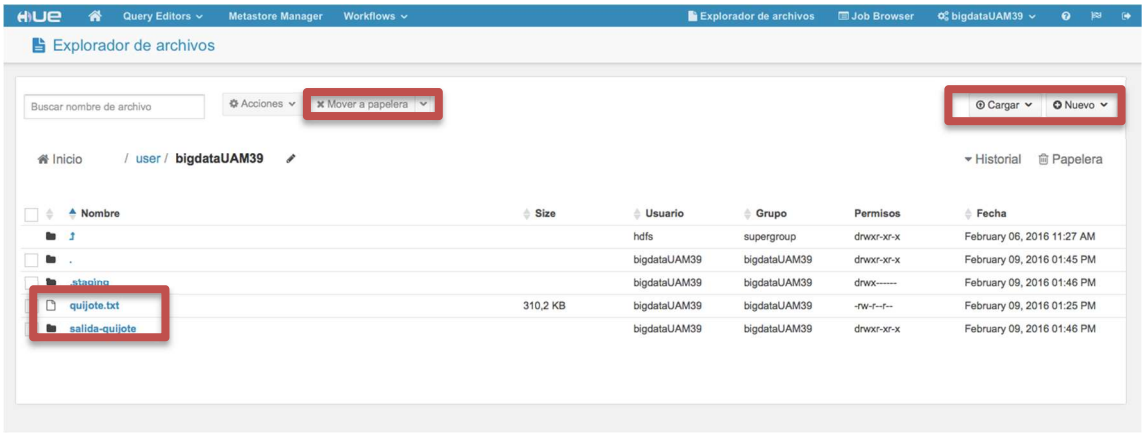
Nota: aunque en nuestra configuración coincidan, el nodo que albergue el servicio no tiene por qué ser el nodo frontend del clúster, sino cualquiera de los nodos del mismo. El nodo que aloje el servicio HUE es elegido durante la fase de instalación del servicio. En nuestro caso, se decidió que la configuración fuera de esta manera debido a que el nodo frontend es el único que tiene una dirección IP accesible desde el exterior de la red de la Universidad Autónoma de Madrid.

Accedamos pues a la interfaz HUE de nuestro clúster:



El explorador de archivos de HUE

HUE ofrece una interfaz sencilla desde la que el usuario del clúster puede consultar y modificar la parte del sistema de ficheros sobre la que tiene permisos de consulta/edición.

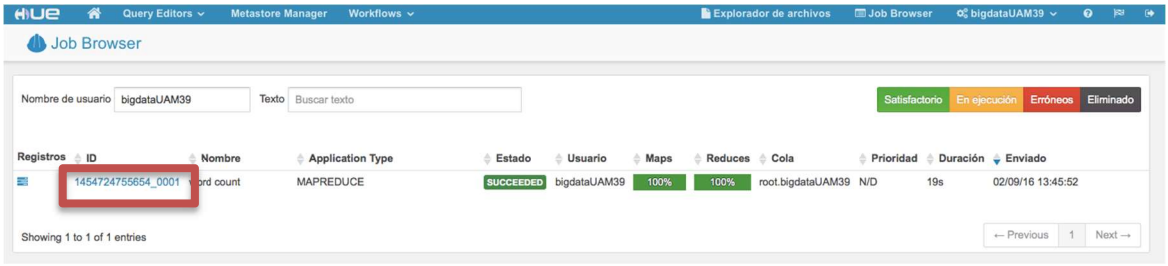


Ejercicio: utilizar el interfaz de HUE para cargar un fichero en tu directorio del HDFS. Cópialo cambiándole el nombre, y después elimina la copia original.

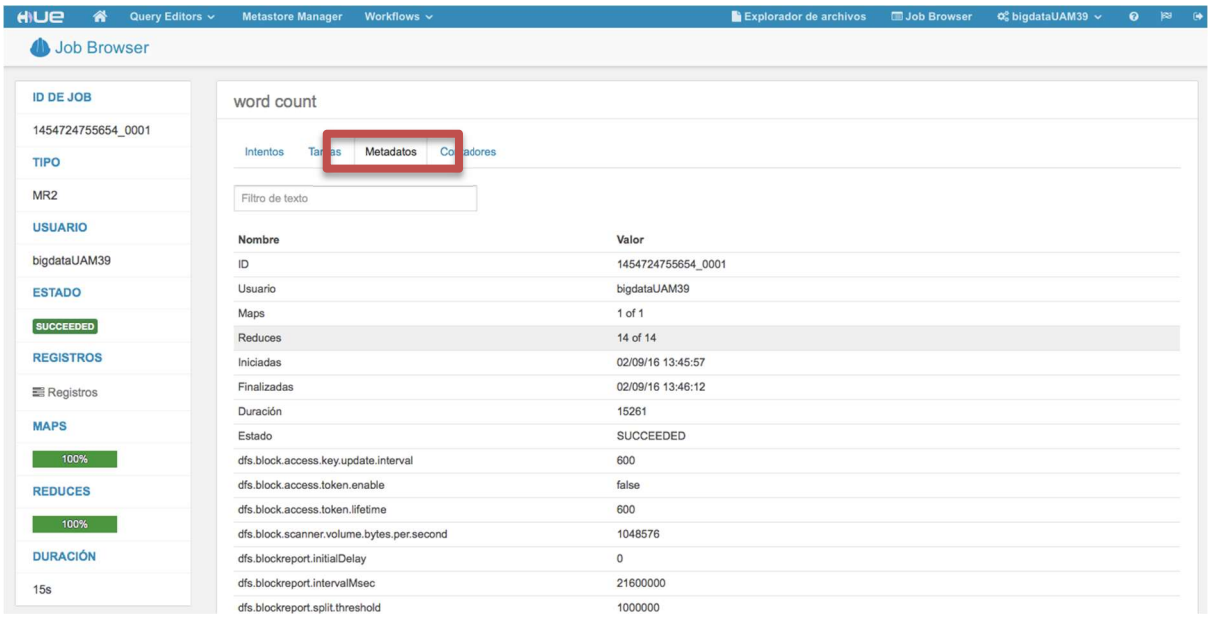
Nota: cuando desde HUE se manda un archivo a la "papelera", el espacio que el archivo ocupaba no es liberado, ya que pasa a formar parte de un directorio "/user/uambdXX/.Trash/Current/user/uambdXX", en el que permanecerá hasta que se borre automáticamente tras transcurrir un periodo de tiempo establecido, o que alguien lo borre de forma explícita.

Navegador de trabajos en HUE

HUE proporciona también una interfaz sencilla desde la que acceder a la información de los trabajos lanzados (tanto terminados como en ejecución) por el usuario.



Desde esta interfaz, el usuario puede consultar de una manera visual el estado de sus trabajos, así como acceder a detalles de más bajo nivel sobre la configuración de los mismos.



Al hacer click en el trabajo deseado, HUE nos muestra una interfaz en la que podemos acceder a los detalles de ejecución de cada uno de los intentos de ejecución de nuestro trabajo (recordemos que Hadoop reintenta el trabajo en caso de un fallo en algún nodo del clúster). En particular, en la pestaña de metadatos, podemos acceder a los detalles de configuración de nuestra tarea.

Ejercicio: consultar los siguientes datos de configuración de la tarea "WordCount" de ejemplo lanzada al inicio de la práctica:

- Número de tareas Map lanzadas
- Número de tareas Reduce lanzadas
- Duración de la ejecución de la tarea
- Estado de terminación

Ejericio: utilice el script "creaFichero.bash" que se provee para crear un fichero de texto más grande:

```
./creaFichero.bash 80
```

Esta ejecución creará un fichero de texto "veryBig.txt".

Ahora, ejecute el ejemplo WordCount de nuevo, y obtenga utilizando la interfaz de HUE los datos de configuración de la nueva tarea que se pedían anteriormente. ¿Ha cambiado algo? ¿A qué se debe?

Ejercicio: vuelva a la lanzar la ejecución de una tarea "WordCount" sobre el fichero "veryBig.txt". Acceda durante su ejecución al *Job Browser* de HUE y, utilizando la interfaz gráfica, termina (mata) la tarea. ¿Qué mensaje obtenemos en la consola desde la que lanzamos la tarea? ¿Y qué vemos en la configuración del trabajo matado en HUE?

Benchmarking de un clúster Hadoop

A continuación, vamos a ejecutar una serie de pruebas de rendimiento sobre nuestro clúster Hadoop.

TeraSort

TeraSort es uno de los ejemplos de referencia a la hora de evaluar el rendimiento de un clúster Big Data:

http://sortbenchmark.org/YahooHadoop.pdf

El ejemplo TeraSort consiste en realizar la ordenación de 1TB de datos rellenos con contenido aleatorio. El enlace superior incluye una explicación exhaustiva del algoritmo, cuyas características principales explicamos a continuación:

- Hay una etapa de particionado hecha a medida que toma N-1 muestras de los datos a ordenar:

  $$muestra_1, muestra_2, \ldots, muestra_N$$

- Los datos entrantes se distribuyen entre las N tareas reduce de manera que todas las claves cumpliendo

  $$muestra_{i-1} <= clave < muestra_i$$

  van a parar a la tarea i.

- Para acelerar el particionado, se utiliza una estructura de árbol que permita indexar rápidamente los datos de entrada.
- Una vez los datos han sido distribuidos, se ordenan los bloques utilizando el algoritmo QuickSort (con complejidad NlogN).

TeraGen

Puesto que vamos a proceder a realizar pruebas de rendimiento en nuestro clúster utilizando TeraSort, primero necesitamos un fichero "con muchos datos que ordenar". Para ello, podríamos subir a nuestro HDFS un fichero de datos que hayamos obtenido de una fuente externa. No obstante, por limitaciones de almacenamiento y ancho de banda, vamos a utilizar otro de los ejemplos de uso de Hadoop: Teragen. Esta aplicación nos permite generar de forma sencilla un bloque grande de datos aleatorios, sobre los que luego podremos realizar nuestras pruebas de rendimiento.

Para generar los datos, una vez dentro de nuestro clúster ejecutamos:

```
>hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-examples.jar teragen 40000000 terasort-input
```

Esta ejecución nos generará un fichero de tamaño 4GB en la carpeta "terasort-input" de nuestro directorio de usuario en el HDFS. El tamaño que se le pasa como argumento es el número de líneas de 100 Bytes que se desean generar (en nuestro caso $100x4x10^7$ bytes = $4x10^9$ bytes = 4 Gbytes).

Ejercicio: comprobar que el fichero se ha generado correctamente utilizando la línea de comandos de Hadoop, o la interfaz gráfica HUE. ¿Cuántas tareas Map y Reduce se han lanzado para crear nuestro fichero?

Ejecución de Terasort

Una vez tenemos unos datos de entrada con los que trabajar, es el momento de ejecutar nuestra aplicación de ordenación:

```
>hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-examples.jar terasort terasort-input/ terasort-output
```

Esta ejecución nos producirá, en el directorio terasort-output una copia de los datos de entrada, pero esta vez ordenados.

Pregunta: compruebe el tamaño de los archivos de entrada y de salida de TeraSort. ¿Cómo están distribuidos los datos? ¿Sabrías explicar a qué se debe?

Pregunta: compruebe cuántas tareas Map y Reduce se han lanzado para su tarea de ordenación.

Validación de resultados con TeraValidate

Además de las herramientas para generación de datos y ordenación, los ejemplos de Hadoop contienen una herramienta para verificar que la ordenación de ha realizado de forma satisfactoria. Para ejecutarla:

```
>hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-examples.jar teravalidate terasort-ouput/ teasort-validate/
```

Evaluando el impacto de la configuración de la tarea de ordenación en su rendimiento

El ejemplo de ordenación que hemos ejecutado ha tomado un determinado número de tareas Map y Reduce. Queremos ver cómo afecta la configuración de nuestra tarea a su rendimiento.

Ejercicio: utilizando el siguiente parámetro durante la ejecución de la tarea:

```
>hadoop jar /…/hadoop-examples.jar terasort
-Dmapred.reduce.tasks=XX terasort-input/ terasort-output-XX/
```

variar el número de reducers lanzados, y tomar nota del tiempo de ejecución requerido.

Si la tarea inicialmente se lanzó con N tareas reduce, probar a lanzarla con N/4, N/2, 3*N/4, 5*N/4, 3*N/2, 7*N/4 y 2*N tareas. Tomar nota de los resultados, e incluirlos en la entrega de la práctica.

¿Cómo justificarías los resultados obtenidos?

Nota: para evitar una excesiva ocupación del clúster, recordar borrar el directorio resultado de la ejecución de terasort tras la ejecución.

Evaluando el rendimiento del clúster con los tests de Hadoop

Además de los ejemplos de aplicaciones que ya hemos visto y probado, las distribuciones Hadoop incluyen una serie de aplicaciones para llevar a cabo pruebas de rendimiento sobre los distintos componentes del clúster. Para ver el listado de tests disponibles ejecutamos el siguiente comando:

```
>hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-test-mr1.jar

Valid program names are:

  …

  TestDFSIO: Distributed i/o benchmark.

  …

  mrbench: A map/reduce benchmark that can create many small jobs

  nnbench: A benchmark that stresses the namenode.

  …
```

Rendimiento del sistema de ficheros distribuido

Esta aplicación nos permite realizar pruebas tanto de lectura como de escritura en nuestro clúster. No obstante, para poder llevar a cabo las pruebas de lectura, es necesario llevar a cabo primero las de escritura (puesto que se utilizarán los ficheros generados por un test en el siguiente). Para llevar a cabo las pruebas de escritura podemos ejecutar el siguiente comando:

```
>hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-test-mr1.jar TestDFSIO
        -Dtest.build.data=benchmarks/ -write
        -nrFiles 10 -fileSize 1000
```

Los argumentos del programa son el directorio de salida en el que se generarán los ficheros generados en el test, la cantidad de ficheros generados, y el tamaño de los mismos. Un ejemplo de salida es el siguiente:

```
----- TestDFSIO ----- : write
            Date & time: Thu Feb 11 21:14:33 CET 2016
        Number of files: 10
Total MBytes processed: 10000.0
     Throughput mb/sec: 17.36264842894076
Average IO rate mb/sec: 17.441394805908203
 IO rate std deviation: 1.2378632113069292
    Test exec time sec: 78.787
```

La aplicación genera tantas tareas Map como ficheros se hayan solicitado, de modo que cada tarea Map se encarga de escribir/leer un único fichero.

De forma similar, podemos ejecutar los tests de lectura cambiando el parámetro "-write" por "-read". Si queremos limpiar los ficheros generados, simplemente tenemos que ejecutar la aplicación con la opción "-clean".

Ejercicio: obtener datos de rendimiento en el sistema de ficheros distribuido del clúster. Obtener datos de rendimiento al pedir la escritura de 5,10,15 y 20 ficheros, y anótelos. ¿Qué tendencia se observa?

Otros tests de rendimiento

Como se ha mencionado, Hadoop incluye varias aplicaciones de testeo de rendimiento. Más información sobre estas aplicaciones puede encontrarse en el siguiente enlace:

http://www.michael-noll.com/blog/2011/04/09/benchmarking-and-stress-testing-an-hadoop-clúster-with-terasort-testdfsio-nnbench-mrbench/