Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación
IIC-3423 Big Data

Tarea 1: Hadoop

Fecha entrega: 23 Abril

Muchach@s en esta tarea tendrán la oportunidad de experimentar con Hadoop, en particular vamos a utilizar la version de Hadoop que es distribuida por Cloudera: `http://www.cloudera.com/`.

Instalación Máquina Virtual de Cloudera

En esta tarea comenzarán a tener contacto con la máquina virtual que utilizaremos durante esta unidad. La máquina ya viene con una instalación completa de la implementación de Cloudera de Hadoop, por lo que no es necesario instalar nada extra. Además, esta máquina virtual tiene todos los elementos que utilizaremos a lo largo de esta parte del curso, incluyendo las herramientas del ecosistema, lenguajes y diversas bases de datos.

Esta parte de la tarea consiste simplemente en descargar la máquina virtual, montarla y realizar una prueba básica de funcionamiento. Dependiendo de las capacidades de su computador pueden descargar 2 posibles versiones, ambas con funcionalidades similares en términos de los requerimientos del curso:

(a) Versión más recientes. Requiere 8GB Ram:
`https://www.cloudera.com/downloads/quickstart_vms/5-13.html.html`

(b) Versión menos reciente. Requiere 4GB Ram:
`https://drive.google.com/file/d/0Bzj3hS8Ko2fTQTU0UzlwVk9DSkU/view?usp=sharing`

Para montarla, pueden utilizar directamente la aplicación VMware Workstation Player[1]. Es posible también utilizar otras aplicaciones como VirtualBox[2], pero es primero necesario realizar un proceso de conversión de formato de los archivos descargados[3].

Una vez montada y ejecutada la máquina virtual, deberían ver la pantalla en figura 1. Luego en la terminal que aparece en el centro de la pantalla, deberán simplemente ejecutar el comando `hadoop`, que generará la salida en figura 2.

Algunos aspectos técnicos de la máquina virtual

La máquina virtual utiliza la distribución de Linux CentOS 6.3 y viene configurado con la distribución de Cloudera de Hadoop (CDH) instalado en modo pseudo-distribuido. Además del núcleo de Hadoop (HDFS, Common, MapReduce), están instaladas las herramientas del ecosistema necesarias para completar las tareas (Pig, Hive, Flume, etc.) y lenguajes relacionados (Perl, Python, PHP, Ruby, Java, etc). La instalación en modo pseudo-distribuido de Hadoop es un método de funcionamiento en el cual todos los *daemons* se ejecutan en la misma máquina, siendo en esencia un *cluster* de una sola máquina. Las funcionalidades de este modo

[1] http://www.vmware.com/products/player/playerpro-evaluation.html
[2] http://www.virtualbox.org/
[3] http://www.howtogeek.com/125640/how-to-convert-virtual-machines-between-virtualbox-and-vmware/

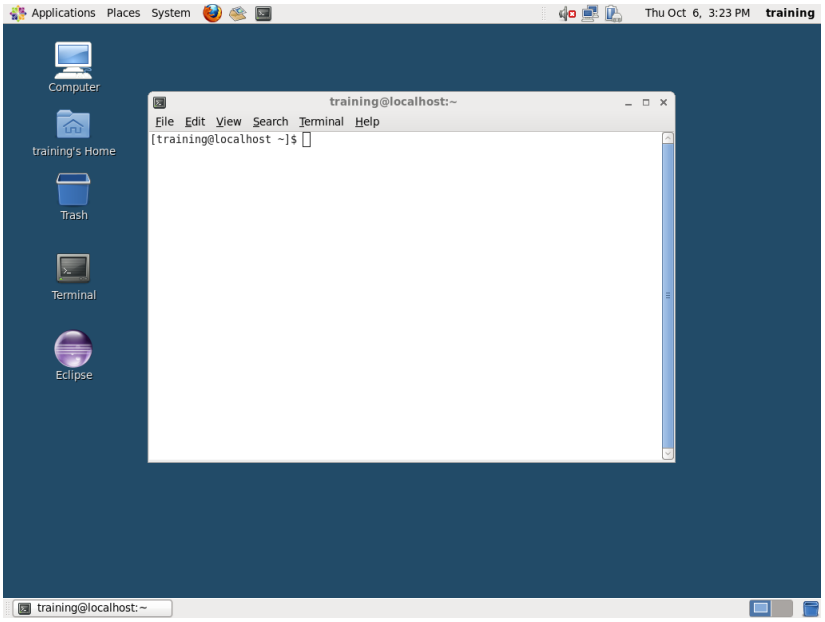

Figure 1: Pantalla durante instalación.

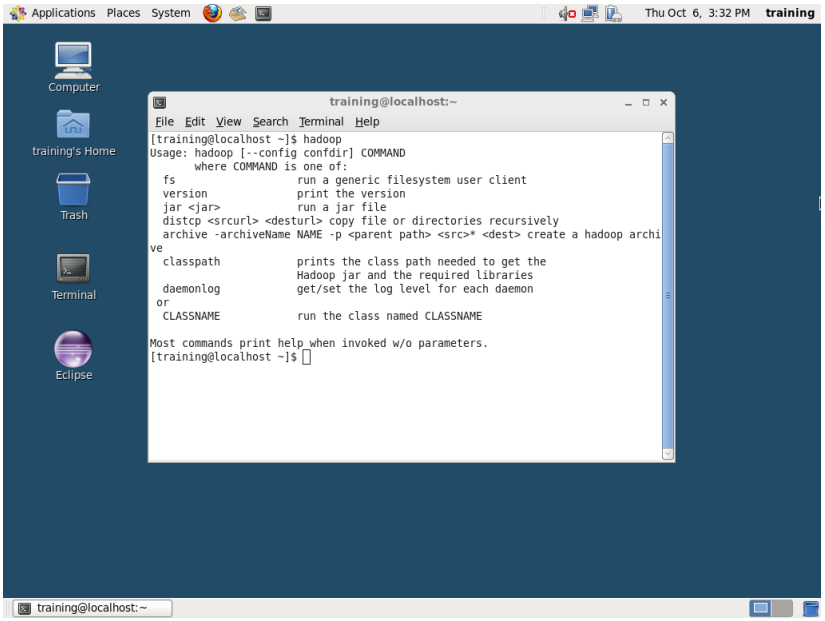

Figure 2: Pantalla durante instalación.

son las mismas que las de una instalación completamente distribuida, aparte de la velocidad. Además, dado que sólo existe un disco en la máquina virtual, el factor de replicación de bloque se encuentra fijo en 1. Finalmente, la máquina virtual está configurada para identificarse automáticamente con el usuario `training`, utilizando la contraseña `training`.

1 Interacción con Hadoop Distributed File System (20%).

En esta actividad, ejercitarán el uso del sistema de archivos de Hadoop, HDFS, mediante la máquina virtual montada en sus computadores. Utilizando los comandos descritos en la siguiente figura y a través de una terminal, deberán interactuar con distintas funcionalidades de HDFS. El modo de uso de estos comandos es el siguiente: `hdfs dfs comando`, donde `comando` es uno de los comandos que aparecen en la Figura 3.

Command	Description
-ls *path*	Lists contents of directory
-lsr *path*	Recursive display of contents
-du *path*	Shows disk usage in bytes
-dus *path*	Summary of disk usage
-mv *src dest*	Move files or directories within HDFS
-cp *src dest*	Copy files or directories within HDFS
-rm *path*	Removes the file or empty directory in HDFS
-rmr *path*	Recursively removes file or directory
-put *localSrc dest* (Also –copyFromLocal)	Copy file from local filesystem into HDFS

Command	Description
-get *src localDest*	Copy from HDFS to local filesystem
-cat *filename*	Display contents of HDFS file
-tail *file*	Shows the last 1KB of HDFS file on stdout
-chmod [-R]	Change file permissions in HDFS
-chown [-R]	Change ownership in HDFS
-help	Returns usage info

Figure 3: Ejemplo de comandos para manejar funcionalidades de HDFS.

Comenzaremos revisando los comandos principales, en particular:

- Crear un directorio en HDFS, dentro del *home* del usuario actual.
- Listar el contenido de un directorio en HDFS que no se encuentre vacío.
- Copiar un archivo, desde el sistema de archivos local, a HDFS.
- Mover y copiar archivos dentro de HDFS.
- Verificar información sobre el uso del disco por parte de HDFS.
- Borrar un archivo y un directorio en HDFS.

Actividades:

(a) Escribir y leer archivos de distinto tamaño, midiendo el tiempo de transferencia. ¿Es lineal el crecimiento de este tiempo?. En su análisis considere archivos que sean menores y mayores al tamaño de bloque.

(b) Repetir el ejercicio anterior, pero esta vez utilizando un tamaño de bloque distinto. ¿Tiene este cambio algún efecto en el tiempo de transferencia?. En su análisis considere tamaños de bloque menores y mayores al valor seteado en la máquina virtual.

Con el fin de verificar los efectos sobre el sistema de archivos, puede utilizar los comandos `hdfs fsck filename` y `hdfs dfsadmin report`, para revisar el estado de los archivos copiados y del sistema, respectivamente. Para modificar el tamaño de bloque, es necesario editar el archivo de configuración central de HDFS, ubicado en /etc/hadoop/conf/hdfs-site.xml, agregando la siguiente propiedad:

```
<property>
  <name>dfs.block.size<name>
  <value>134217728<value>
  <description>Block size<description>
<property>
```

Donde *value* indica el tamaño del bloque en bytes, que en este caso equivale a 128 MB. Recuerde que el password de superusuario es training.

2 Cuenta de Palabras (Word Count) (40%).

Como vimos en clases, uno de los ejemplos clásicos del uso del paradigma de programación Map Reduce es el contar la ocurrencia de cada una de las palabras que aparecen en un conjunto de documentos. En esta parte de la tarea tendrán la oportunidad de implementar esta aplicación usando Python. Para ello usarán HadoopStreaming, una aplicación incluida en la distribución de Hadoop que permite ejecutar funciones de Map y Reduce desde el shell de la máquina virtual. En términos de la tarea, lo importante es el comando de HadoopStreaming para ejecutar las funciones de Map y Reduce, la sintaxis es:

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
-input nombre_directorio_con_archivo_entrada \
-output nombre_directorio_con_archivo_entrada \
-mapper ubicacion_y_nombre_funcion_map.py \
-reducer ubicacion_y_nombre_funcion_reduce.py
```

(Obs: el símbolo \ significa que el comando sigue en la siguiente línea). Por defecto la función HadoopStreaming guarda su salida en el archivo part-00000.

Pueden ver opciones de la función HadoopStreaming con el comando:
`hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar --help`

Para una completa introducción a HadoopStreaming pueden consultar: `https://hadoop.apache.org/docs/r1.2.1/streaming.html`. Una guía más breve a HadoopStreaming está disponible en: `https://wiki.apache.org/hadoop/HadoopStreaming`.

En su informe de tarea, incluya el archivo de salida con el resultado del contador de palabras para el archivo *palabras.txt* disponible en el sitio web del curso. Además incluya su código para las funciones de Map y Reduce.

3 Union de datos parte 1 (joining data) (20%).

Una aplicación típica de SQL es combinar 2 set de datos de acuerdo a alguna llave. Por ejemplo, para el caso de unir tablas A y B según cierta llave (key), el pseudo código podría ser :

```
Select * from tableA, tableB, where
A.key=B.Key
```

En esta parte de la tarea utilizarán Hadoop para unir el contenido de 2 archivos: dataCuentaTotal.txt y dataCuentaDiaria.txt. El archivo dataCuentaTotal.txt contiene el resultado de la cuenta total de cierto grupo de palabras, utilizando el formato $< palabra, cuentaTotal >$. Por ejemplo:

alumno, 15
universidad, 18
profesor, 16
...

El archivos dataCuentaDiaria.txt contiene el resultado de la cuenta diaria de palabras según fecha, utilizando el formato $< fecha\ palabra, cuentaDiaria >$. Por ejemplo:

Ene-5 alumno, 1
Feb-10 universidad, 3
Jun-5 profesor, 2
...

El objetivo es unir (join) estos archivos, de tal manera que se genere un archivo de salida con la cuenta diaria y total de cada palabra, utilizando el formato $< fecha\ palabra, cuentaDiaria\ cuentaTotal >$. Por ejemplo:

Ene-5 alumno, 1 15
Feb-10 universidad, 3 18
Jun-5 profesor, 2 16
...

Una aplicación de este tipo podría servir para identificar palabras que en determinadas fechas aparecen más de lo usual, por ejemplo, *empanadas* en fechas cercanas al 18 de Septiembre, o *regalo* en fechas cercana a navidad.

- En términos de Map Reduce, ¿Cuál sería una buena elección del *key* para el mapper? ¿Cuál para el *value*?
- Teniendo en cuenta que el mapper deja los datos ordenados por key, cuál sería la operación del reducer?
- En su informe de tarea incluya su código para las funciones de Map y Reduce.

4 Union de datos parte 2 (20%).

- Descargue desde el sitio web del curso el archivo *TvChannelInfo.zip*. Este archivo contiene una serie de archivos que corresponden a 2 posibles tipos: i) Archivos con información de programas de televisión que son emitidos por distintos canales, y ii) Archivos con información del número de televidentes que observan distintas emisiones de cada programa.
- Implemente funciones de Map y Reduce que permitan calcular el total de televidentes que ve cada uno de los programas emitidos por cierto canal de televisión. En particular, en el reporte de su tarea entregue como resultado la cuenta de los televidentes que vió programas de CAB. Por ejemplo, si el total de televidentes que vió los programas Baked-Sports y Hourly-Talking emitidos por CAB es 10.500 y 11.900, respectivamente, su archivo de salida debería contener líneas indicando:
  Baked-Sports 10.500
  ...
  Hourly-Talking 11.9003
  ...

Al realizar sus funciones tenga en cuenta que un mismo programa de televisión puede ser emitido por varios canales. Además, cada combinación (TvShow, channel) puede aparecer múltiples veces, y cada combinación (TvShow, audiencia) también puede aparecer múltiples veces con distinta cuenta.

- En su informe de tarea incluya su código para las funciones de Map y Reduce.