Big Data Analytics - Practical Laboratory 1

Issam Falih

1 Programming Environment Setup

To get you ready for the programming portion of the assignment, you'll need to do a few things to get your programming environment setup and to get a version of hadoop setup. *You'll need to do this before you can fully complete either of the assignment parts*, though you can work on some portions of Part 1 in parallel if you get stuck.

1.1 Getting Hadoop running

Because setting up and maintaining a hadoop cluster can be challenging, instead, each of you is going to run your own mini-hadoop cluster as a virtual machine. You have two options for where to run hadoop, you can either run it on your laptop or you may use the cloud e.g. Hadoop google cloud.

- Install VirtualBox: If you're running on your laptop, install VirtualBox (`https://www.virtualbox.org/wiki/Downloads`). The machines in 105 already have it installed. VirtualBox is a virtual machine software that will allows you load an image of a machine and run it.
- Download VM image: Download the VM image from:

  `http://content.udacity-data.com/courses/ud617/Cloudera-Udacity-Training-VM-4.1.1.c.zip`

  and save it somewhere where you'll remember and unzip the file.

  These images are very large (4.2GB). If you are running this in the lab, when you are all done with the assignment, please delete the image file so that we don't use up extra space on the department file server.
- Setup new VM
  - Start up VirtualBox (if you're on a Mac, you can just press cmd+spacebar and search for it or it should be installed in your Applications folder).
  - Click the "New" button to create a new VM.
  - Give it a name (I called mine "hadoop"), select "Linux" and then "Ubuntu (64-bit)" then continue on.

- For memory size, enter 512 MB (though you could do larger if it makes you happy :), then continue on.
- Select "Use an existing virtual hard drive file", navigate to the location where you unzipped the Cloudera VM file and within that directory select the file "Cloudera-Training-VM-4.1.1.c.vmdk" (it should be highlighted).

  This should finish the installation and you should see a new VM in the VirtualBox manager window that says "Powered Off".

- Setup network communication for the VM

  1. Select the newly created VM in VirtualBox manage.
  2. Select the "Setting" button and then the "Network" tab.
  3. Under Adapter 1, change "Attached to:" to "Bridged Adapter", then select "OK".

You should now be able to select your VM image and click the "Start" button and it will boot the VM.

1.2 Transferring files to the VM

Once the VM boots, you should be able to interact with it. One thing you will want to do is to transfer files back and forth between your computer and the VM. When the VM is running, you can interact with it like any other linux server (i.e. `ssh` into it, `scp` files back and forth, etc.). The only catch is that the VM server doesn't have a name, so you'll need to get the VMs IP address.

To do this, in the VM terminal window, type `ifconfig` and you will get back networking info that looks something like:

```
eth1      Link encap:Ethernet  HWaddr 08:00:27:80:A7:A8
          inet addr:134.173.84.25  Bcast:134.173.85.255  Mask:255.255.254.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2000479 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12086 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:284526061 (271.3 MiB)  TX bytes:862768 (842.5 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:263313 errors:0 dropped:0 overruns:0 frame:0
          TX packets:263313 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:56949120 (54.3 MiB)  TX bytes:56949120 (54.3 MiB)
```

The IP address of the server is on the second line following `inet addr:`, in the case above, `134.173.84.25`, though yours will be different. The username and password for this machine are `training`. Given this you can copy files to the server using `scp`, e.g.

```
$ scp some_file.txt training@134.173.84.25:file_name_on_vm.txt
```

If you're rusty on `scp` lookup some examples online. Additionally, I recommend `ssh`ing into the machine to interact with it rather than try and do it via the VirtualBox window. Once the server is running, you can login to the server using `ssh`:

```
$ ssh training@134.173.84.25
```

This has the advantage of using your native terminal to interact with the server rather than through VirtualBox. The biggest benefit of this is that copy and paste will work, though I also find it to be more convenient in general.

1.3 Eclipse setup

For this assignment, we'll be using Eclipse. We'll need to do a few things to setup the workspace in Eclipse so that we can compile mapreduce programs.

- Download from myEfrei the hadoop libraries files and save somewhere, and then unpackage the directory.
- Open Eclipse and create a new Java project. During creation:
  - Under "Use an execution environment JRE:", select "JavaSE 1.6".
  - Check "Create separate folders for source and class files".
  - Click "Next" and in the second screen, select the "Libraries" tab.
  - Click the "Add External Jars" button and then navigate to the location of the hadoop libraries folder and add *all* of the `.jar` files.
  - Click "Finish".
- To test that this works:
  - Create a new package called "demos".
  - Create a new Java class called `WordCount`.
  - Go to the course webpage and find the `WordCount` demo from the class notes and copy and paste the contents into your `WordCount` class. If everything is setup correctly, this file should compile.

2 Hadoop Basics and MapReduce

2.1 Hadoop Basics

At this point, you should have a working hadoop VM setup, you should be able to copy files to that VM and you should have your Eclipse environment setup to write MapReduce programs.

To run your own code on the cluster you'll need to go through a few steps:

1. Write your code in Eclipse and make sure that it's compiling without any errors.
2. In the shell (e.g. Terminal) go to the `bin` directory of your Eclipse workspace.
3. Create a `.jar` file of the compiled version of your code:

   ```
   $ jar -cvf name_of_jar_file.jar packages_to_include
   ```

   For example, if I wanted to do it for the `WordCount` class in the `demos` package, I would type:

   ```
   $ jar -cvf wordcount.jar demos
   ```

   Note that you need to do the full package structure, so it should be at the base of the `bin` directory in your workspace.
4. Copy this `.jar` file over to your VM server.
5. Finally, run it:

   ```
   $ hadoop jar name_of_jar_file.jar name_of_main_class
   ```

   For example, for the `WordCount` class it would be:

   ```
   $ hadoop jar wordcount.jar demos.WordCount
   ```

   Assuming everything worked well, you should see printed out:

   ```
   WordCount <input_dir> <output_dir>
   ```

Question 1: What are the results of running the `WordCount` demo code on the following text:

```
I do not like them in a house .
I do not like them with a mouse .
I do not like them here or there .
I do not like them anywhere .
I do not like green eggs and ham .
I do not like them , Sam-I-am .
```

One route to doing this would be:

- Create a copy of this file on your computer.
- `scp` the file to the VM.
- Using `hdfs` create the input directory for your run (most hadoop programs operate on directories, even if it's just a single file).
- Using `hdfs` copy the file from the VM to hdfs.
- Run the hadoop `WordCount` program, specifying the output directory.
- Copy the output file(s) back to the VM using `hdfs` (either just `get` or `getmerge`) and then back to your computer using `scp`. (Of course, if the file is short, you could also just view it either in `hdfs` or on the VM).

Part 2: Sales analysis

For the last part of this assignment we're going to write another map/reduce Hadoop program.

We will work now on a sales record data. The data can be downloaded on myEfrei (Go to Lab1 materials then download sales data).

It is a CSV file. Check its format; it has many columns. Also note it has a header ligne, which may be problematic when loading the file from your Hadoop implementation.

We are interested in specifically five columns: the sales region (Region), the sales country (Country), the type of item bought (Item Type), the sales channel (Sales channel : online or offline), and finally the total sale profit (Total profit).

It is highly advised to allow your program to adapt its behaviour / pick an analysis task depending on a command line argument.

Alternatively and if you don't see how to perform this, you can implement a separate software for each of the required tasks.

Remark: the Configuration object that we create in the Driver class can also be obtained later on from the Map and Reduce classes; and it also allows passing values between those classes, which may prove useful for this program. You are advised to look for the way to obtain the Configuration instance from the context object in the map and reduce classes, as well as looking up the getter and setter methods for the Configuration class.

Your software should be able to perform the following tasks:

- Obtain the total profit for any given world region.
- Obtain the total profit for any given country.
- Obtain the total profit for any given item type.

- For each item type, provide:
  - How many sales were performed online.
  - How many sales were performed offline.

  ... and for each of those quantities, how much the combined total profit for those sales was.