Lab 3 Report

Anuj Sampat
asampat@bu.edu

1. Problem

To analyze the species data in specieswiki-latest-pages-articles.xml and apply the page rank formula to iteratively assign a popularity rank to all the listed species, resulting in converged values for the ranks.

2. System Configuration

- OS: Ubuntu 14.04 LTS 64-bit
- Home Directory: /home/anuj
- Hadoop Version: hadoop-0.20.2
- Hadoop Installation Directory: /home/anuj/hadoop-0.20.2
- Java Installation Directory: /home/anuj/jdk1.8.0_05
- Hadoop Distributed Cluster consisting of one master node and two slave nodes, all running Ubuntu 14.04.

3. Problems Encountered

- The data in the wiki species XML has certain properties (species titles) in which the "== Taxonavigation ==" marker is not demarcated from the names of zoologists by the "== Name ==" marker, which results in the names showing up as sub-species. There does not seem to be any way to reliably fix this in parsing. This should not affect the overall results though.
- The XML streaming library (hadoop-0.20.2-streaming.jar) provided by Hadoop 0.20.2 has a bug which causes duplicate XML properties show up when streaming XML input to the mapper. This was fixed by using the XML streaming library provided by hadoop 0.22.0.

4. Changes Made (Bugs Fixed?)

- The damping factor in SpeciesIterMapper2 was changed to 0.85 and 0.15 since it seemed to provide a better value for the rankings. Also, the total number of links N was calculated and added to the species rank as a measure of its own rank given by the formula (1 – d)/N where d is the damping factor.

  The output.collect() call in SpeciesIterMapper2.java where each outlink is output with its inlink was removed. This is not required because the mapper only needs to output the outlinks with their calculated page ranks. Furthermore, doing this will break the next output.collect() call where the inlinks are output with their outlinks.

An extra output.collect() call was added to the SpeciesIterMapper2 mapper to output the outlink with its current rank prepended by the string "oldrank=". This allows the reducer to know what the previous rank of a species was.

The combiner was removed from the job configuration in SpeciesIterDriver2.java since that was causing the reporter to return a null counter during the combine phase. Removing this simplified the process of incrementing a global counter in the reduce phase used to communicate the requirement of additional iterations.

A new job is created in SpeciesIterDriver2.java for each rank calculation iteration. This allows access to the API waitForJobToComplete() (JobClient returns RunningJob, which is not accessible in the Hadoop 0.20 API).

5. Method Adopted

The SpeciesIterDriver2 class is the main driver that calculates the ranks in three steps:

Step 1: SpeciesGraphBuilderMapper and SpeciesGraphBuilderReducer.

The Map function corresponding to this MapReduce job parses out the species name (<title>) and the sub-species identified by the string “== Taxonavigation ==” from the input XML. A DOM parser based approach is used for this. The Reduce function simply outputs the species with an initial rank of 1 followed by a list of its sub-species.

Step 2: Iteration in SpeciesIterMapper2 and SpeciesIterReducer2, leading to converged ranks.

The first iteration of MapReduce in this job simply counts the total number of links in the graph built in step (1) by using a counter called TOTAL_LINKS. This counter value is made available to subsequent job iterations through the job configuration structure.

The next part of this MapReduce step is to iteratively calculate the rank of a species. The mapper in SpeciesIterMapper2.java assigns ranks to a given species and its sub-species based on the page rank formula. These ranks are sent to the reducer, along with the current (old) rank of the species. The reducer sums up the ranks of the species to give a new total rank and compares it against the old rank of a species. If the first two decimal places of the new and old rank don't mtach, the rank calculation is repeated. The reducer communicates the requirement of an additional iteration through the global counter ITERATIONS_NEEDED. The reducer also outputs the species name, with its current rank and list of sub-species.

Step 3: SpeciesViewerMapper.

This phase is simple a Map function that outputs the species ranks in descending order. The reducer used is an identity reducer.

6. Hadoop Configuration

Hadoop was configured to run in a fully-distributed mode, with one master and two slaves, all running Ubuntu 14.04 via Virtual Box. The cluster was setup by adding the following to the Hadoop configuration files on all three machines:

In core-site.xml:

```
<property>
  <name>fs.default.name</name>
  <value>hdfs://master:54310</value>
</property>
```

In hdfs-site.xml:

```
<property>
  <name>dfs.replication</name>
  <value>2</value>
</property>
```

In mapred-site.xml:

```
<property>
  <name>mapred.job.tracker</name>
  <value>master:54311</value>
</property>
```

Additionally, on the master, the hostname of the master was added to conf/masters and the hostnames of the master and two slaves were added to conf/slaves.

The HADOOP_CLASSPATH environment variable in hadoop-env.sh was changed to point to the Hadoop Streaming jar file hadoop-0.22.0-streaming.jar.

The HTML pages master.html and nodes.html provided with this report give additional details about this configuration.

7. Execution And Testing

The following commands were used to compile and package the java code:

```
$~/jdk1.8.0_05/bin/javac -classpath "/home/anuj/hadoop-0.20.2/contrib/streaming/hadoop-
0.22.0-streaming.jar:/home/anuj/hadoop-0.20.2/hadoop-0.20.2-core.jar" -d classes/ <....java
files>
```

```
$~/jdk1.8.0_05/bin/jar cvf  speciesRank.jar -C classes/ .
```

The wiki species file was then copied to the Hadoop HDFS using the command:

```
$bin/hadoop dfs -copyFromLocal /home/anuj/<wikispeciesfile> /tmp/hadoop-anuj/input
```

The program was executed in Hadoop using:

```
$bin/hadoop jar speciesIter.jar BU.MET.CS755.SpeciesIterDriver2 /tmp/hadoop-anuj/input
/tmp/hadoop-anuj/output
```

7. Results

The Hadoop webpages masterTaskTracker.html, masterJobTracker.html, slave1TaskTracker.html, slave2TaskTracker.html and hdfsOutput.html have been provided with this report.

The output of the program was generated in the following directories:

output0/part-00000: The list of the species parsed from the XML, followed by an initial rank of 1 and its sub-species.

output2 to output<n>: This contains the output files of the 'n' iterations for the species rank calculation.

outputFinalRanks/part-00000: The final values (multiplied by 1000 and negated). of the species ranks. This is provided with the report as the file SpeciesFinalRanks.txt.

The total number of iterations required to obtain the converged ranks was 51. The species with the highest rank was Euphaedra. The Wikipedia page for this species states the following in the taxonomy section of the page:

"Euphaedra is a species rich genus. In the most recent monograph Jacques Hecq listed 180 species later adding 12."