Project Big Data

Using the Hadoop Map/Reduce Framework

RDF Statistics Calculation

Submitted by
Vasantharaja Valaiyapathi
M.Tech –Database Systems
IIIT-Srirangam

TABLE OF CONTENTS

Contents

1 *Introduction*

The purpose of this project is to use ***Hadoop*** by creating jobs Map / Reduce to calculate statistics on data ***n-triple*** format. The different statistics will be calculated:

Number of occurrence of each URI or literal

- In the "Subject" RDF triples.
- In the "property" of RDF triples.
- In the "Object" field of RDF triples.

The 10 most common properties.

10 classes appearing most frequently.

10 subjects with the greatest number of distinct properties specified.

2 *Implementation*

To implement the different jobs, we chose to use ***Python 2*** and ***Hadoop*** Streaming to focus on implementing rather than wasting time doing properly run Java programs.

2.1 **Hadoop Streaming**

Hadoop Streaming is supplied with an extension that allows Hadoop to run any script or executable as job Map / Reduce. The scripts read data from standard input and write the results to standard output. So our python scripts are simplified because they only read and write to the standard input and standard output after completing treatment on the data. No need to run a file-by-file playback.

2.2 **Number of occurrence of URI**

To carry out this job, the mapper parses each line by applying a regular expression to retrieve the interesting part of the line, namely the subject, property or object. The mapper then writes to standard output couple (***URI, 1***) for each URI.

The alphabetic sorting allows the reducer to group according to URI and count the number of occurrences of these stages are identical except for the regular expression that will retrieve the right information.

We performed three different jobs in order to retrieve each result individually and easily.

https://hadoop.apache.org/docs/r2.6.0/hadoop-mapreduce-client/hadoop-mapreduce-client-core/HadoopStreaming.html

2.3 **The 10 most common properties**

To realize these requests, we execute two jobs. The first is the occurrence of each property account, as seen above.

The second job is as follows. The reverse mapping the URI and the occurrence account and write to standard output. Sorting in descending order is used between the mapper and reducer. The reducer plays the first 10 entries and written to standard output. The other entries are discarded.

2.4 **The 10 most frequent classes**

As before, we need two jobs to make this request. The first job parse each line and retrieves the subject, property and object only if it is between chevron, i.e. <***object***>. Items in quotes are not considered since they are not classes.

If ever the property contains ‘http://www.w3.org/1999/02/22-rdf-syntax-ns#type’, this means that the object is a class. The mapper then written on the standard peer output (***object,1***).

The second job map / reduce is the same as the previous query.

2.5 **The 10 subjects with the most different properties**

To obtain this information, the principle is similar to previous requests. The difference is that the first map provides the pairs (***subject, property***) to standard output. The reducer has distinct properties and provides the pair (***subject, number of properties***). This is done by adding each property *set* in a python, avoiding duplication, and recovering the size of the set.

The second job map / reduce is identical to previous requests.

3 ***Execution***

In order to execute the jobs, you must first extract the data using the ***extract.sh*** script in the ***data folder***. This will create the file ***.nt*** in the data file containing the data ***n-triples*** format. This data will be used for the jobs.

In the ***src*** folder, ***1-occurrence*** folder contains scripts to count the occurrences of the ***subjects, properties*** and ***objects***. To get the results, simply running the script provided in the folder, for example:

./execHadoop.sh compte_sujets / mapper.py compte_sujets / reducer.py

../../ data / files output

To run scripts in other folders, simply use the following command:

./execHadoop.sh ../../ data / files output

The script will automatically use ***mapper1.py*** and ***reducer1.py*** for the first job and ***mapper2.py*** and ***reducer2.py*** for the second job.

The ***demo.sh*** script in the ***src*** directory to automatically execute the different jobs and consolidates the results of each query in the result file. To run it, use the following command:

./demo.sh

4 *Conclusion*

This project has allowed us to discover how to use and implement Hadoop Map / Reduce jobs. It was interesting to discover this framework and successfully run jobs in order to extract information. It was an interesting first experience with Hadoop.