Faria Mardhani
12/12/17
W205_Storing
Exercise 2 Architecture

Faria W205 Twitter Application

Application Idea:

The Faria_W205 Twitter App has been developed to obtain a livestream of currently-posted tweets from twitter, parse all words from the given tweets, and append each word and its count of occurrences in a PostgreSQL database. The database can then be queried using the finalresults.py and histogram.py files to return the counts of all words in the database, to return the count of only words of interest, or to return all the words for which their counts fall within a given range.

Such an app can be used for many applications – such as to study trending topics on Twitter or to study public sentiment regarding any given issue. The documentation below describes the architecture of the application. Detailed step-by-step instructions on how to run the application can be found in the Readme.txt file.

Description of Architecture:

The entire application resides in the MIDS-INFO-W18/w205_2017_fall/Exercise_2 Github repository. Within the repository, we have created a directory and file structure to store each part of our application.

To begin with, let's discuss the topology of our application. As can be seen from Figure 1 of the Exercise prompt below, the Topology of the application consists of the following parts:

1. A 3-process tweet "spout" which obtains a livestream of data from Twitter
2. A 3-process parse-tweet "bolt" which parses the words from the livestream data
3. A 2-process count "bolt" which counts the occurrences of each word and writes them to our PostgreSQL database

4. In addition to this topology, we also have two files to run queries on our database, called finalresults.py and histogram.py

We have therefore built our directory/file structure to match the topology above in the Exercise_2 directory:

/Exercise_2:

In the /Exercise_2 directory, we have the following directories and files:

/extweetwordcount – a directory that contains the spouts, bolts, and topology files of our application

/screenshots – a directory that contains screenshots of end-to-end execution of the application

Architecture.pdf – this file

readme.txt – a file which provides step-by-step instructions regarding how to run the program

Plot.py – a program which creates a bar chart of the top 20 words in our Twitter stream when run. When this file is run, it will out put a file called "Plot.png" in the Exercise_2 directory, which will be a picture of the bar chart.

plot.png – output of Plot.py file above

Twittercredentials.py – a file that connects us to the twitter application with the necessary access tokens. It is used by the sample application hello-stream-twitter (below)

hello-stream-twitter.py – a sample file to test our livestream connection to Twitter through the application (not part of our application)

psycopg-sample.py – a sample file to test our ability to write words and counts to the PostgreSQL database (not part of our application)

psycopg-word-count.py – another test file we have written to be able to write to the PostgreSQL database (not part of our application)

finalresults.py – a file used to query our PostgreSQL database to determine the counts of a single word or of all the words in the database

histogram.py – a file used to query our PostgreSQL database to return the words and counts of all of the words which have counts that fall within a specified range of integers

---

/screenshots:

In this directory, we have the following files:

/extweetwordcount

In the /extweetwordcount directory, we have the following files and directories:

/topologies – a directory which contains our "tweetwordcount.clj" file which runs our entire application topology (1 spout and 2 bolts). Our spout and bolts files depend on this file

/src – a directory which contains our **/bolts** and **/spouts** directories

The **/virtualenvs** directory and the remaining files in the **/extweetwordcount** directory have been provided automatically through **/extweetwordcount** program in order to enable the application.

/spouts directory:

The **/spouts** directory contains our **tweets.py** file which obtains a livestream of data from Twitter and passes it on to our parse-tweet bolt. This file has a dependency on our **tweetwordcount.clj** file.

/bolts directory:

The **/bolts** directory contains our **parse.py** file which obtains data from the spout and parses tweets. It also contains the **wordcount.py** file which obtains data from the **parse.py** file and counts the number of occurrences of each word and writes both the word and counts to the PostgreSQL database. Both of these files depend on the **tweetwordcount.clj** file.

Figure 1 (Topology):

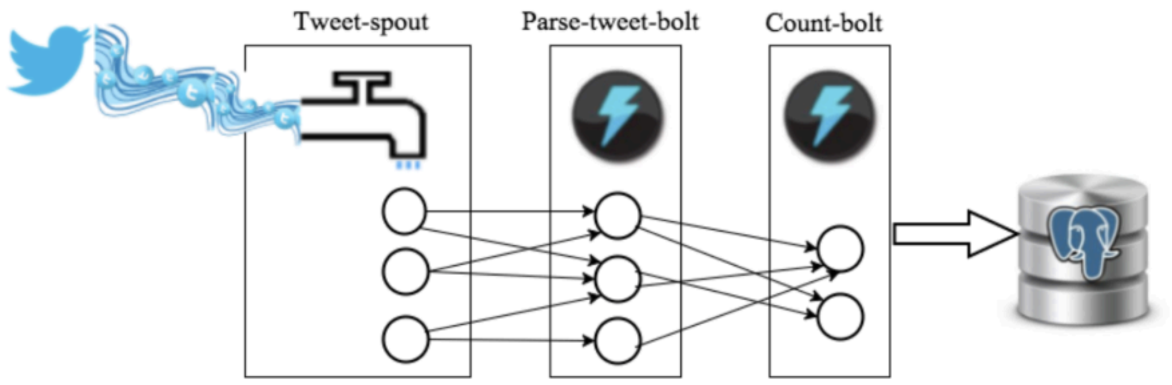

Figure 1: Application Topology