Exercise 2 Architecture

Real Time Data Processing Using Apache Storm

MIDS W205

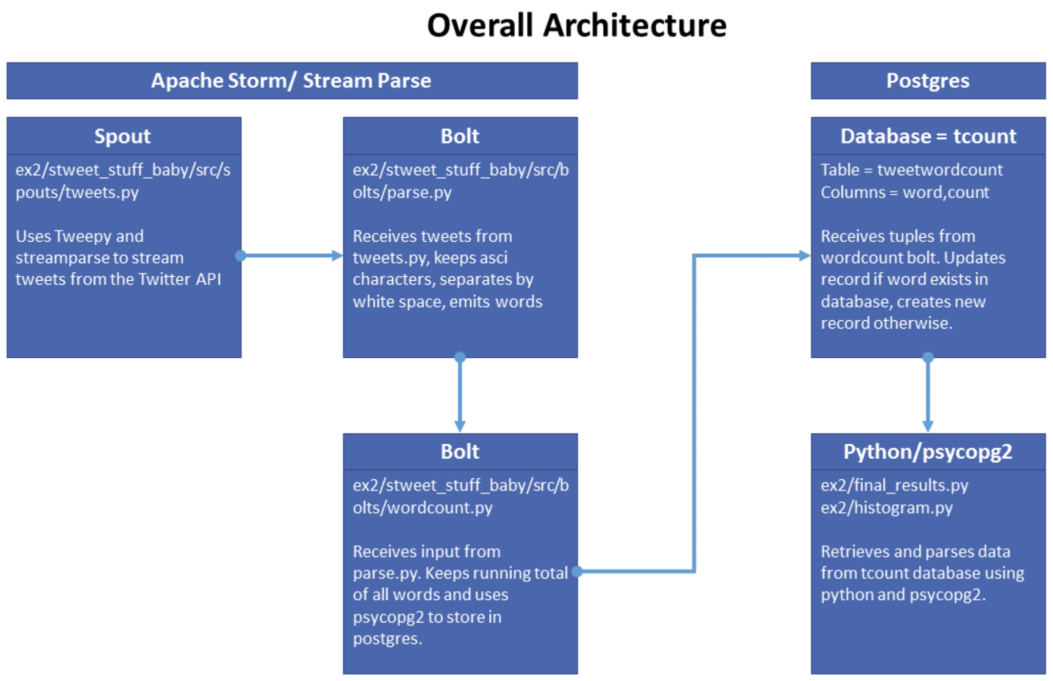
Steve Pelkey

Executive Summary

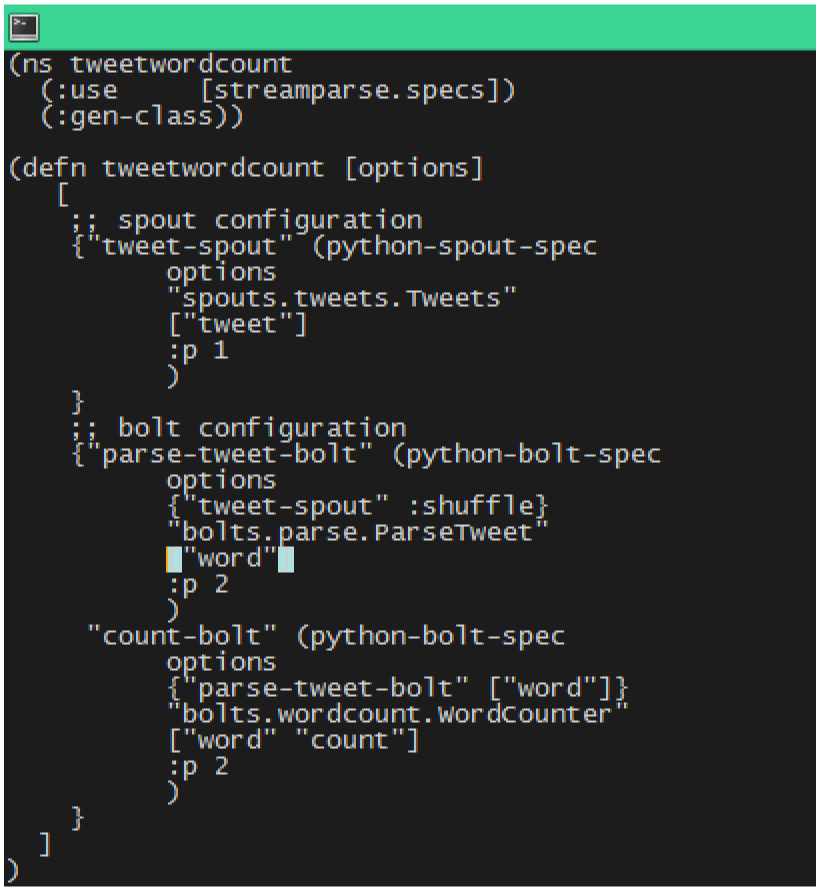
This application captures live data from the Twitter API, and uses StreamParse to create an Apache Storm processing pipeline that parses and maintains a real-time cumulative count of unique word usage. It uses the python package psycopg2 to store, update, and retrieve records from a postgres database. Two different functions, final_results.py and histogram.py were created to quickly retrieve records from the postgres database.

Architecture

The overall architecture of the application is displayed below:


Overall Architecture
Apache Storm/ Stream Parse
Postgres
Spout
ex2/stweet_stuff_baby/src/spouts/tweets.py
Uses Tweepy and streamparse to stream tweets from the Twitter API
Bolt
ex2/stweet_stuff_baby/src/bolts/parse.py
Receives tweets from tweets.py, keeps asci characters, separates by white space, emits words
Database = tcount
Table = tweetwordcount
Columns = word,count
Receives tuples from wordcount bolt. Updates record if word exists in database, creates new record otherwise.
Bolt
ex2/stweet_stuff_baby/src/bolts/wordcount.py
Receives input from parse.py. Keeps running total of all words and uses psycopg2 to store in postgres.
Python/psycopg2
ex2/final_results.py
ex2/histogram.py
Retrieves and parses data from tcount database using python and psycopg2.

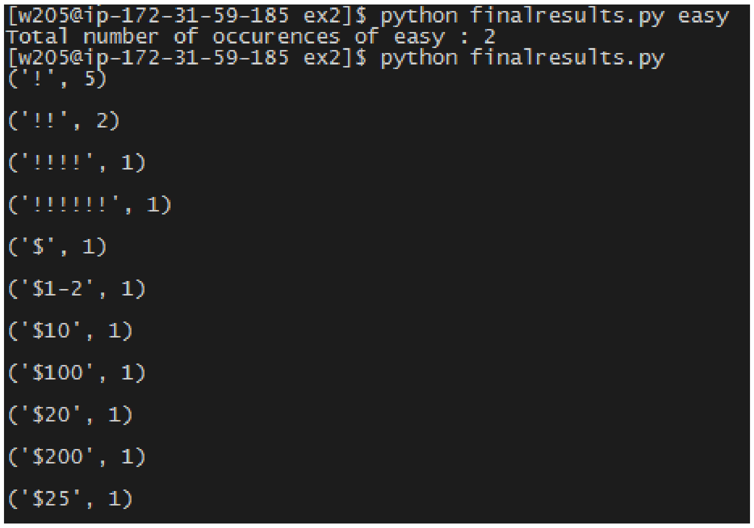
Storm topology is displayed below and can be found in stweet_stuff_baby/topologies/tweetwordcount.clj



```
(ns tweetwordcount
  (:use     [streamparse.specs])
  (:gen-class))

(defn tweetwordcount [options]
   [
    ;; spout configuration
    {"tweet-spout" (python-spout-spec
          options
          "spouts.tweets.Tweets"
          ["tweet"]
          :p 1
          )
    }
    ;; bolt configuration
    {"parse-tweet-bolt" (python-bolt-spec
          options
          {"tweet-spout" :shuffle}
          "bolts.parse.ParseTweet"
          ["word"]
          :p 2
          )
     "count-bolt" (python-bolt-spec
          options
          {"parse-tweet-bolt" ["word"]}
          "bolts.wordcount.WordCounter"
          ["word" "count"]
          :p 2
          )
    }
  ]
)
```

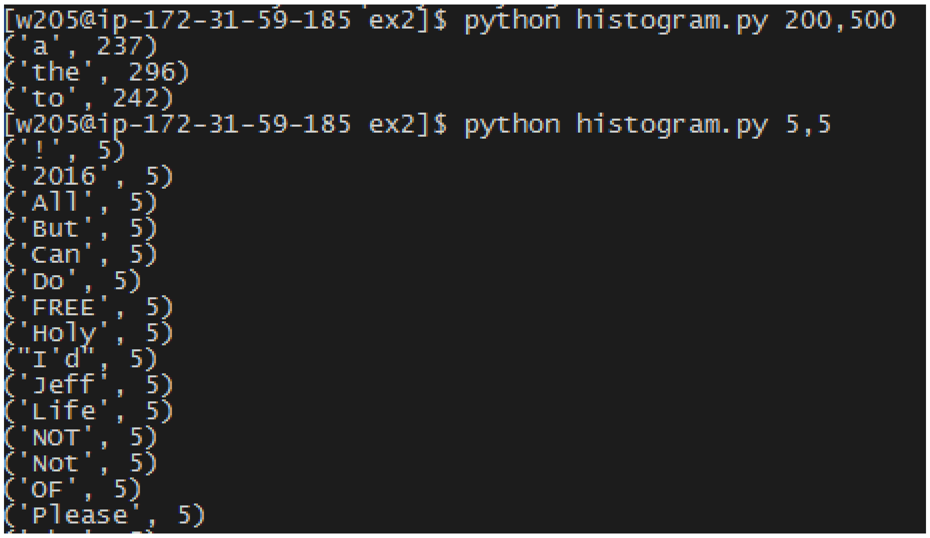
Running the Program

1. Fire up an Amazon EC2 instance using the UCB MIDS W205 EX2-FULL AMI and install postgres
2. Download this git repository
3. Create a postgres database called 'tcount'. It will be accessed through port 5432.
4. Create a table titled 'tweetwordcount' with two columns (word, count) to store Twitter data from Storm
5. Ensure psycopg2 and tweepy are installed for python
6. Navigate to the Storm streamparse project folder called 'stweet_stuff_baby' and run.
7. Let application run for as long as you please. It is populating the postgres ‘tcount’ database
8. Two serving scripts are available to query the postgres database. Call them from your command line.
    a. finalresults.py
       Takes a word as an argument and returns the total number of word occurrences in stream. If you don't provide an argument, all words in stream will be returned.



```
[w205@ip-172-31-59-185 ex2]$ python finalresults.py easy
Total number of occurences of easy : 2
[w205@ip-172-31-59-185 ex2]$ python finalresults.py
('!', 5)

('!!', 2)

('!!!!', 1)

('!!!!!!', 1)

('$', 1)

('$1-2', 1)

('$10', 1)

('$100', 1)

('$20', 1)

('$200', 1)

('$25', 1)
```

b. histogram.py
   Returns the words that have counts in an inclusive range of integers. Takes one argument in the form of two positive integers separated by a colon.



```
[w205@ip-172-31-59-185 ex2]$ python histogram.py 200,500
('a', 237)
('the', 296)
('to', 242)
[w205@ip-172-31-59-185 ex2]$ python histogram.py 5,5
('!', 5)
('2016', 5)
('All', 5)
('But', 5)
('Can', 5)
('Do', 5)
('FREE', 5)
('Holy', 5)
("I'd", 5)
('Jeff', 5)
('Life', 5)
('NOT', 5)
('Not', 5)
('OF', 5)
('Please', 5)
```