Yueqi Su & Justin Zheng

OMIS 107

Twitter Project

Functionalities

1. Retrieves all of the tweets relative to a certain hashtag (a word given by the user) and stops after a certain number of seconds (given by the user, optional)
2. Collects how many times per 10 seconds a certain hashtag has been tweeted
3. Perform statistical analysis on hashtag data
4. Displays hashtag data in histogram form
5. Writes warning messages on the screen when a spike in tweets/10 second is observed
6. Kills program if needed with only one call

Note: Source Codes are listed below and color-coded to indicate different functionalities in same code file.

Output Files

Log.txt → all incoming tweets

Tweet.txt → count of tweets per 10 second

Function 1, 2 & 5 // script.sh

Run with: $ ./script.sh twitter-hashtag [time-to-die(in seconds)]

```
#!/bin/bash

# remove tweet.txt if exists before starting
if test -r tweet.txt
  then rm tweet.txt
fi

# running twitter hashtag
python tweetering.py $1  > log.txt &

# if user inputs "Time to Die", wait and kill
if [ $# -eq 2 ]
then
(sleep $(( $2 + 1 )); echo Program Terminated!; ./kill.sh;) &
fi

# checking new tweets per 10 second
count=1

while [ 1 -eq 1 ]
do
  # total duration of sleep could be changed here by changing the
number to desired time in seconds
  sleep 10
  temp=$(cat log.txt | grep -Ev "^RT" | wc -l)
  new=$(($temp-$count))
  count=$temp
  echo $new >> tweet.txt

  # Warning message if high traffic
  if [ $new -ge 100 -a $new -lt 200 ]
    then
    echo Warning: High number of tweets being observed
  fi
  if [ $new -ge 200 ]
    then
    echo Warning: Very high number of tweets being observed!!!
  fi
done
```

Function 3 & 4 // graph.cpp

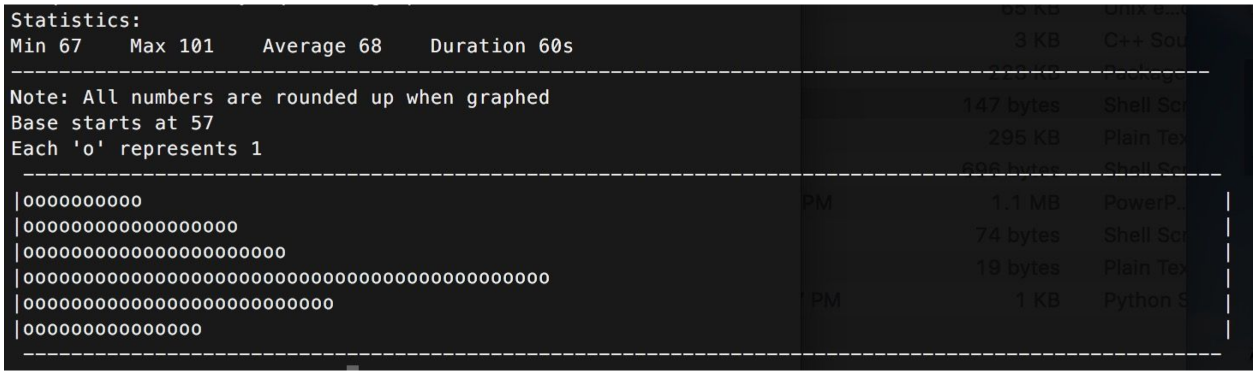
Compile with $ g++ -g -o graph graph.cpp
Run with $ ./graph
(after execution of script.sh)

```
#include <iostream>
#include <fstream>
#include <vector>
#include <map>
#include <cmath>

using namespace std;

/*
 * Scale number on base and/or range of input values
 * (Max character allowed in one line is 100)
 * if number is scaled, we want at least 1 "o" for each row (except
when number is 0)
 *
 * return new base and start value in pair
 * default value base=0, val=1
 *
 */
pair<int, int> scaleNum(int min, int max){
     int base=0;
     int val=1;
     if (max>100){ // large base
          base=min-10;
     }

     if ((max-min) > 90){ // large range
          //cout<<max-base<<endl;

          // find the smallest x (> 1) where
          // max/x - base/x < 100
          val = ceil((max-base)/double (100));
     }

     //cout<<"Base: "<<base<<" val: "<<val<<endl;
     return ( pair<int, int> (base, val) );
}
```

```
/* Original bash code for graph

cat tweet.txt | while read number
do
  result=""
  if [ $number = '0' ]
  then
    echo $result
  else
    for n in $(seq 1 $number)
    do
      result+="o"
    done
    echo $result
  fi
done
*/

/*
 * Draw histogram based on base and val provided
 *
 */
void histogram(vector<int> num, int base, int val){
     // formatting
     for (int k=0; k<100; k++){
          cout<<'-';
     }
     cout<<endl;

     cout<<"Note: All numbers are rounded up when graphed"<<endl;
     cout<<"Base starts at "<<base<<endl;
     cout<<"Each 'o' represents "<<val<<endl;

     // formatting
     cout<<' ';
     for (int k=0; k<100; k++){
          cout<<'-';
     }
     cout<<' ';
     cout<<endl;

     //graph
	for (int i=0; i<num.size(); i++){
		cout<<'|';
		int j;
		for (j=0; j<ceil((num[i]-base)/double (val)); j++){
			cout<<'o';
		}
		//fill rest with space
		for (j; j<100; j++){
			cout<<' ';
		}
		cout<<'|'<<endl;
	}

	//formatting
	cout<<' ';
	for (int k=0; k<100; k++){
		cout<<'-';
	}
	cout<<' ';
	cout<<endl;
}


int main(){
	vector<int> num;
	// open tweet.txt
	ifstream ifs("tweet.txt");
	if (ifs.fail()){
		cout<<"Can't open tweet.txt!"<<endl;
		exit(1);
	}
	// read tweet.txt into vector
	string n;
	while (getline(ifs, n)){
	num.push_back(stoi(n));
	}
	// close connection
	ifs.close();

	int total=0;
	int min = num[0];
	int max = num[0];
	// find min and max of data
```

```
        for (int i = 1; i<num.size(); i++){
        if (num[i]<min){
                min = num[i];
        }
        else if (num[i]>max){
                max = num[i];
        }
        total += num[i];
        }
        int avg = total/num.size();

        // scale graph
        pair<int, int> temp = scaleNum (min, max);
        int base = temp.first;
        int val = temp.second;
        //cout<<"Range: "<<base<<" val: "<<val<<endl;

        cout<<"Statistics:"<<endl;
        cout<<"Min "<<min<<"    Max "<<max<<"    Average "<<avg<<"
Duration "<<((num.size())*10)<<"s"<<endl;

        // print histogram
        histogram(num, base, val);

        return 0;
}
```

Sample Output:



```
Statistics:
Min 67    Max 101    Average 68    Duration 60s
----------------------------------------------------------------------------------------------------
Note: All numbers are rounded up when graphed
Base starts at 57
Each 'o' represents 1
 ----------------------------------------------------------------------------------------------------
|oooooooooo                                                                                          |
|oooooooooooooooooo                                                                                  |
|oooooooooooooooooooooo                                                                              |
|oooooooooooooooooooooooooooooooooooooooooooo                                                        |
|oooooooooooooooooooooooooo                                                                          |
|ooooooooooooooo                                                                                     |
 ----------------------------------------------------------------------------------------------------
```

Function 6 // kill.sh

Run with $ ./kill.sh

```
#!/bin/bash

# kill script.sh & tweetering.py

for pid in $(ps | awk '{if(($4~/\/bin\/bash/) || ($4~/python/)) print
$1}')
do
  kill -9 $pid
done
```