Shadman Ahmed
Amir Zamli

Lab Assignment 3 – Spark Streaming, Kafka, and Cassandra

Introduction

The purpose of this lab is to implement an application that process live data stream. The general topology of such process includes producers which sends the input data to the brokers whose tasks are to partition the received data to the consumers accordingly. In this lab, the data is taken from the Kafka application. The Spark Stream provides the continuous processed stream of data to Cassandra where it will store the data in the desired format.

Implementation

To be able to store the results in Cassandra we begin by connecting to Cassandra via the cluster builder method and then proceed to use the session execute method with a query for creating a keyspace if it does not already exist, then creating the table avg_space with the data storage format text, float.

We then configure Kafka to send data to the Spark Streaming framework, here we use the receiver-less direct approach.

The data will be received by the generator with key values as null, and the values in “String, Int” string format, so we map the messages using functional programming method .map to first split the string, and then .map to cast the second element to Double, as that will be the input value to the mapping function that we define next.

The mapping function, that will have the task of calculating the average value for each key character takes the key character and the value that we cast previously and store this in a hashmap instance, the hashmap takes the key and stores the sum of all values corresponding to that key together with the number of occurrences of that key. The mapping function then returns the key and average value-pair, the average value is calculated by dividing the sum of all values with the number of occurrences of that key, both of which is stored in the hashmap for that corresponding key value.

When the program finishes, it saves the values to Cassandra in the avg table, in the avg_space keyspace. And then terminates appropriately.

```
cqlsh:avg_space> use avg_space;
cqlsh:avg_space> select * from avg;

 word | count
------+----------
    z | 12.59044
    a | 12.78085
    c | 12.65507
    m | 12.69243
    f | 12.40079
    o | 12.46548
    n | 12.53037
    q | 12.39685
    g | 12.34307
    p | 12.21033
    e | 12.28107
    r | 12.69009
    d | 12.59286
    h | 12.27803
    w | 12.78082
    l | 12.34309
    j | 12.38812
    v | 12.46731
    y | 12.26468
    u | 12.40615
    i | 12.32986
    k | 12.65245
    t | 12.76857
    x | 13.01465
    b | 12.34286
    s | 12.27967

(26 rows)
cqlsh:avg_space>
```

Fig 1: The Result, this shows the saved average occurrence of the characters from the generator stream, data processed and calculated by our kafka & Cassandra application.