Big Data Management

Assignment 3

Description

In this assignment your task is to compute the real-time views. You can use the same dataset as in previous assignments. You are required to use Apache Spark's Streaming API to compute the real-time views. For storing these views you need to use the Apache Cassandra.

Dataset

Transport Infrastructure Ireland (TII) (`https://www.tii.ie`) operates and maintains a network of traffic counters on the motorway, national primary and secondary road networks in Ireland. These traffic counters capture data on different parameters. There are currently around 400 of these counters active across the network. For an interactive view of the data they capture, go to the TII Traffic Counter Data Website: (`https://www.nratrafficdata.ie`). On this website, green dots display the individual traffic counter locations around the country. Summary traffic data information can be obtained from each site by clicking on the green dot. Upon clicking a dot, a pop-up window will appear that summarises the traffic data and provides a link to more detailed data. You can click on the **list of sites** button to view description of each counter.

The traffic counter data set is a valuable source of information on vehicle movements across the national road network and is made available publicly in its raw form, in order to provide researchers, public bodies, engineering companies, as well as the general public, with the opportunity to analyse and query the data independently for their own specific purposes. The first row of each file contains headers which describe each field. However, the meaning of some of these may not be apparent to consumers. The following explains some of the less obvious column headers:

cosit: The unique identifier for the traffic counter device. In conjunction with the siteâĂŹs dataset, this can be used to determine the location and route of the counter, used to record the vehicle movement.

lane: The Id of the lane in which the movement was recorded, which is specific to each counter.

straddlelane: If a value is present, this indicates that the vehicle may have been changing lanes as it passed over the counter.

class/classname: This indicates the category of vehicle that was recorded e.g. car, bus, etc.

length: The approximate length of the vehicle recorded.

headway: The approximate distance between the front of the recorded vehicle and the vehicle behind.

gap: The approximate distance between the rear of the vehicle and the front of the vehicle behind.

weight: This is available on (Weigh-in-Motion) WIM sites only and indicates the approximate weight of the vehicle.

temperature: If available, this indicates the approximate surface temperature of the road at the location of the device.

numberofaxles: This is available on WIM sites only and indicates the number of axles detected for the vehicle.

axleweights: This is available on WIM sites only and expresses as an array of real numbers, the weight over each axel in order.

axlespacing: This is available on WIM sites only and expresses as an array of real numbers, the distance between each of the axles.

Setting Up

Follow the `Getting Started with Spark Streaming` document provided on Moodle. For help Spark Streaming API and various operations and transformations that you can apply on DStreams see the following links:

`http://spark.apache.org/docs/latest/api/python/pyspark.streaming.html`
`https://spark.apache.org/docs/latest/streaming-programming-guide.html`

Questions

To answer the questions below you must use Apache Spark's Streaming API. This time we are interested in real-time processing of the traffic counters dataset. The results should be persisted in the Cassandra storage (use the `append` option).

1. To emulate a live-stream of the traffic counter dataset, you are required to write a separate Python script that reads 10 records (of

each counter site - ignore the `test` site) every 5 seconds from traffic counter data file and stores them as separate files (`countdata1, countdata2, countdata3, etc.`) in the streaming directory on which your application is listening. `(20 marks)`

2. Prepare the streaming application to read the data streams from the streaming directory using a batch length of 5 seconds. `(10 marks)`

Define the following streaming computations (every 5 seconds):

1. Show total number of counts (on each site of M50) by vehicle class. `(10 marks)`
2. Compute the average speed (on each site on M50) by vehicle class. `(10 marks)`
3. Find the top 3 busiest counter sites on M50. `(10 marks)`
4. Find total number of counts for HGVs on M50. `(10 marks)`

Store the results of streaming computations defined above:

1. Prepare Cassandra data structures to store the results. `(10 marks)`
2. Prepare code for writing the results into the Cassandra tables. `(20 marks)`

Submission

- Create a PDF document that contains the code you used to answer queries + some screenshots of the results stored in Cassandra (after running your streaming application for 5 minutes for each question).
- Acceptable code file format: Python notebook - name it `assignment3.ipynb`. The notebook should be exported as iPython Notebook with *.ipynb extension. If the code in your notebook does not run, it will result in 20% penalty.
- Zip both files: the PDF and the Python notebook. Submit the zip file on Moodle before the deadline.
- Do not submit work thatffs not your own and do not let others copy work that is your own. Both Copyier and Copyee will get ZERO marks.