Cloud Computing Capstone Task 2 Report

Roney Duílio Stein

Video: https://youtu.be/

Source code: <https://github.com/roneystein/aviation-spark-processing>

# General Considerations

* The datasets used to answer the questions were:
  + **Airline Origin and Destination Survey**: used only for question 1.1 (it’s a bigger dataset)
  + **Airline On-Time Performance Data**: remaining questions.
* On-time **performance** is defined as the percentage of flights arrived with **15 or more minutes** of delay, as defined by fields ArrDel15 and DepDel15.
* Questions selected: 1.1, 1.3, 2.2, 2.3, 2.4, 3.1, 3.2.
* Video doesn’t cover loading all the scenarios because of the time constraint of 5 minutes.

# Data extraction and cleaning

During Task 1: used a shell script to verify the archive’s ZIP files integrity, the existence and content of CSV files inside each. Two ZIP files had no valid content. Extraction was automated using two shell scripts that, when run in the directory containing the data files/directories, processed the content using pipes, not temporary files, as follows:

1. Extracted the CSV files from ZIP files;
2. Processed through the Perl script (one for Origin Destination and another for On-time tables) filtering the desired data and only required columns;
3. Compressed using BZip2;
4. Piped to proper HDFS location.

In the end, for task 1, we had two pre-processed file sets, one for Origin-Destination (Q1.1) and another for On-time statistics.

These task 1 input file sets were transferred for one of the Kafka nodes.

# System construction and integration

The Spark cluster was setup using 5 EC2 m4.xlarge nodes. Cassandra cluster was built using the same cluster nodes. Kafka cluster consisting of 3 nodes using EC2 c4.large instances.

Having the data in one the the Kafka nodes the data was input into the topics using “bz2cat <files> | kafka-console-producer”, one file at a time.

The Spark programs stored the data to be presented in Cassandra using the Datastax connector.

# Optimizations

Some optimizations made in the process or system:

* Partitioning the Kafka topics for better computing parallelization;
* Used Kafka’s Direct Stream for better and easier parallelization.
* Kafka compression

# Algorithms

The algorithms were coded using Scala.

**Question 1.1:** A word counting.

1. Reading the Origin-Destination messages from Kafka into a stream;
2. For each line tokenize words (separate origin from destination);
3. Group by airport and count the grouped lines;
4. Sort by number of lines (arrivals and departures) in descending order;
5. Crop the top 10 and display them.

**Question 1.3:** Calculate average value on a group of words.

1. Load all on-time input files;
2. Keep only the day of week and arrival delay flag (set when >15 min. delayed) fields;
3. Group by day of week;
4. For each day of week calculate the percentage of delayed arrivals;
5. Sort by the percentage in ascending order;
6. Output day of weed and percentage of delayed arrivals.

**Question 2.2:** Calculate average for a group then reorder and crop the results.

1. Load all on-time input files;
2. Keep origin, destination and departure delayed flag (set when >15 min. delayed) fields;
3. Group all records by origin-destination tuples;
4. For each origin-destination group calculate the percentage of delayed departures and generate: origin, destination and percentage;
5. Group by origin;
6. For each origin:
   1. Order by percentage of delayed departures in ascending order;
   2. Crop the top 10;
7. Generates: origin and list of destinations, comma-separated, in order of least delayed (left) to most delayed (right).

**Question 2.3:** Calculate average for a group, sort the carriers and list the top 10 as a comma separated list.

1. Load all on-time input files;
2. Keep origin, destination, carrier and arrival delayed flag (set when >15 min. delayed) fields;
3. Group by tuples of origin-destination-carrier;
4. For each tuple calculate the percentage of delayed arrivals (arrival performance);
5. Group by origin-destination pairs;
6. For each pair:
   1. Order by percentage of delayed arrivals in ascending order (or arrival performance in descending order);
   2. Crop the top 10;
7. Generates: origin, destinations and top 10 least delayed carriers, in a comma-separated list, in order of least delayed (left) to most delayed (right).

**Question 2.4:** Calculate average value for a group.

1. Load all on-time input files;
2. Keep origin, destination and arrival delayed minutes (0 or positive numbers);
3. Group all records by origin-destination pairs;
4. For each pair calculate the average of delayed minutes;
5. Outputs origin, destination and the calculated average.

**Question 3.2:** Split the data into two different groups of records; calculate a time difference and merge then based on the time difference and origin/destination.

1. Load all year 2008 on-time input files;
2. For each line keep/generate: date, “carrier + flight number”, origin, destination, departure time, arrival delay minutes (>0), date +2 days
3. Filter the lines with departure time < 12:00h to a group “X”, the first leg;
4. Filter the lines with departure time > 12:00h to a group “Y”, the second leg;
5. For X and Y, each: for each date – origin – destination we keep only the most on-time flight;
6. Merge X and Y generating all the possible combinations for each date;
7. Output: X departure date, X departure time, X flight, X origin, X destination or intermediate airport, Y destination of final destination, Y departure date, Y departure time, Y flight, total delay in minutes.

# Results

**Question 1.1:** Top 10 most popular airports.

|  |  |
| --- | --- |
| **Popularity** | **Airport** |
| 58187766 | ATL |
| 49596357 | ORD |
| 44373231 | DFW |
| 31219499 | DEN |
| 25452018 | LAX |
| 24668293 | MSP |
| 24276984 | CLT |
| 22927673 | DTW |
| 21864094 | PHX |
| 21501371 | IAH |

**Question 1.3:** Rank of the days of weed by on-time arrival performance.

|  |  |
| --- | --- |
| **Day of Week** | **% of delayed arrivals** |
| Saturday | 17.170948184159514 |
| Tuesday | 18.837742475105646 |
| Monday | 19.60122803398963 |
| Sunday | 19.787507762028206 |
| Wednesday | 20.43301101641691 |
| Thursday | 22.98366271134333 |
| Friday | 23.879051559476984 |

**Question 2.2:** For each X airport rank the top 10 airports in decreasing order of on-time departure performance from X.

|  |  |
| --- | --- |
| **Origin** | **Least Delayed (left to right)** |
| CMI | PIT, DAY, STL, PIA, DFW, CVG, ATL, ORD |
| BWI | MLB, IAD, DAB, CHO, UCA, SRQ, SJU, OAJ, BGM, GSP |
| MIA | BUF, SAN, HOU, ISP, PSE, SLC, TLH, MEM, GNV, SJC |
| LAX | SDF, LAX, BZN, PMD, MAF, VIS, MEM, IYK, HDN, SNA |
| IAH | MSN, MLI, HOU, AGS, EFD, JAC, RNO, MDW, VCT, CLL |
| SFO | SCK, MSO, SDF, FAR, LGA, PIE, BNA, OAK, MKE, MEM |

**Question 2.3:** For each source-destination rank the top 10 carriers in decreasing order of on-time arrival performance.

|  |  |  |
| --- | --- | --- |
| **Origin** | **Destination** | **Least Delayed Carriers (left to right)** |
| CMI | ORD | MQ |
| IND | CMH | AA, CO, HP, US, NW, DL, EA |
| DFW | IAH | CO, OO, UA, XE, EV, DL, AA, MQ |
| LAX | SFO | TZ, F9, PS, EV, AA, US, MQ, CO, WN, UA |
| JFK | LAX | UA, AA, HP, DL, TW |
| ATL | PHX | FL, US, HP, EA, DL |

**Question 2.4:** For each source-destination determine the mean arrival delay in minutes.

|  |  |  |
| --- | --- | --- |
| **Origin** | **Destination** | **Mean Delay (minutes)** |
| CMI | ORD | 15 |
| IND | CMH | 6 |
| DFW | IAH | 11 |
| LAX | SFO | 13 |
| JFK | LAX | 14 |
| ATL | PHX | 13 |

**Question 3.1:** Does the popularity distribution of airports follow a Zipf distribution? If not, what distribution does it follow?

The popularity is **not** a Zipf distribution as it doesn’t make a straight line in a logarithmic chart. Without extensive research we could conclude that **it follows an exponential distribution** which is both drawn below for comparison using a linear chart using Lambda=0,1.

**Question 3.2:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1st leg departure date** | **Origin Airp.** | **Interm. Airport** | **Dest.**  **Airport** | **1st leg flight** | **1st leg dep. time** | **2nd leg flight** | **2nd leg dep. date** | **2nd leg dep. time** |
| 04/03/2008 | CMI | ORD | LAX | MQ 4401 | 0810 | AA 607 | 06/03/2008 | 1950 |
| 09/09/2008 | JAX | DFW | CRP | AA 845 | 0725 | MQ 3627 | 11/09/2008 | 1645 |
| 01/04/2008 | SLC | BFL | LAX | OO 3755 | 1100 | OO 5429 | 03/04/2008 | 1455 |
| 12/07/2008 | LAX | SFO | PHX | UA 889 | 0759 | WN 2645 | 14/07/2008 | 2030 |
| 10/06/2008 | DFW | ORD | DFW | AA 2332 | 0915 | AA 2333 | 12/06/2008 | 1515 |
| 01/01/2008 | LAX | ORD | JFK | UA 944 | 0705 | B6 918 | 03/01/2008 | 1900 |

# Between Apache Spark and Hadoop