FLIGHT DATA ANALYSIS

CS 644 – Introduction to Big Data

Professor: Chase Wu

TA: Songlin He

Rohan Parekh, Deep Patel

1. Find the 3 airlines with the highest and lowest probability, respectively, for being on schedule

(1) MapperOne

We can use them together as the key because each combination of airline carrier and flight number is unique. For each flight, MapperOne adds the ArrDelay and DepDelay (each row in the table).
When this amount exceeds the threshold time (15 minutes), the flight is considered to be behind schedule. Then put context (total, 1) and context (total, 1). (count , 1). In addition, the number of delayed flights and total flights for each airline carrier will increase by one. Otherwise, if the sum of ArrDelay and DepDelay is less than the threshold, indicating that the aircraft is on time, we must use context write to increase the on time flight by one (count , 1). This will not change the delayed fights of airline carriers and will only increase its total flight number by 1.

(2) ReducerOne

The total number of flights for each airline carrier, as well as the total number of flights that are not on schedule for this carrier, are required to calculate the on schedule probability. The number of total flights and not on schedule flights for each carrier are then counted. The delayed probability for each carrier can then be calculated by dividing the two count numbers, i.e. the number of not on schedule flights by the total number of flights. This reducer stores the delayed probabilities in a linkedlist. The airlines with the highest likelihood of being on time can be found by sorting this linkedlist in ascending order, while the airlines with the lowest probability of being on time can be found by sorting this linkedlist in desending order. Keeping two size 3 sorted lists to sort highest percentage of on schedule flights and lowest percentage of on schedule flights.

2. Find the 3 airports with the longest and shortest average taxi time per flight (both in and out), respectively

(1) MapperTwo

MapperTwo uses the airport code as the key and the taxi time as the value. It generates the arrival and departure taxi times (Origin, taxiTime) as well as the taxi out time (Dest, taxiTime).

(2) ReducerTwo

ReducerTwo adds the taxiIn and taxiOut values to get the total taxi time (in and out time) for each airport. ReducerTwo also keeps track of the overall number of flights for each airport. The average taxi time can then be computed by dividing total taxi time by the number of flights. The value will then be stored in a linkedlist. The airports with the shortest average taxi time can be found by sorting this linkedlist in ascending order, while the airports with the longest average taxi time can be found by sorting this linkedlist in desending order. Maintain two size 3 sorted lists to contain the highest and lowest average taxi time at the airport.

3. Find the most common reason for flight cancellations

(1) MapperThree

MapperThree count this cancellation reason by context write the cancellation code and 1 when a flight is cancelled.

(2) ReducerThree

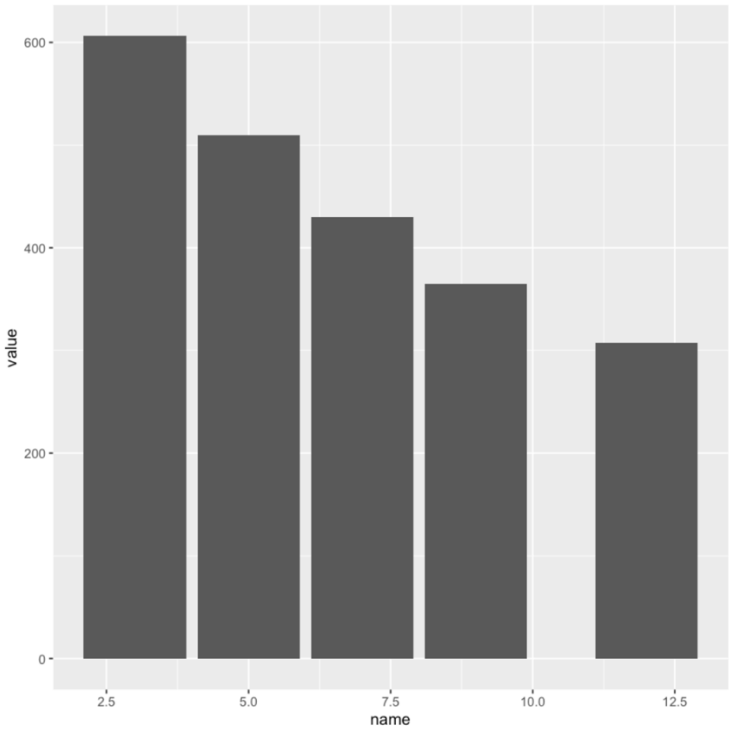
This Reducer is used to determine the most common cause of flight cancellations. All we have to do now is keep track of which cancellation code has the most cancelled flights. Count the number of flights that were cancelled for the same cause (same cancel code), and compare the number to the current code with the most cancelled flights. Replace the cancel code with this count number and make it the new max number if the count number is greater than the existing maximum number. The most prevalent reason for flight cancellations can be discovered this way.

4. Performance measurement

A performance measurement plot that compares the workflow execution time in response to an increasing number of VMs used for processing the entire data set (22 years)

Increase the cluster size from 3 to 12 nodes

Cluster Size	**3**	**5**	**7**	**9**	**12**
Execution Time	9 min 58 s	8 min 20 s	7 min 5 s	6 min 1 s	5 min 5 s

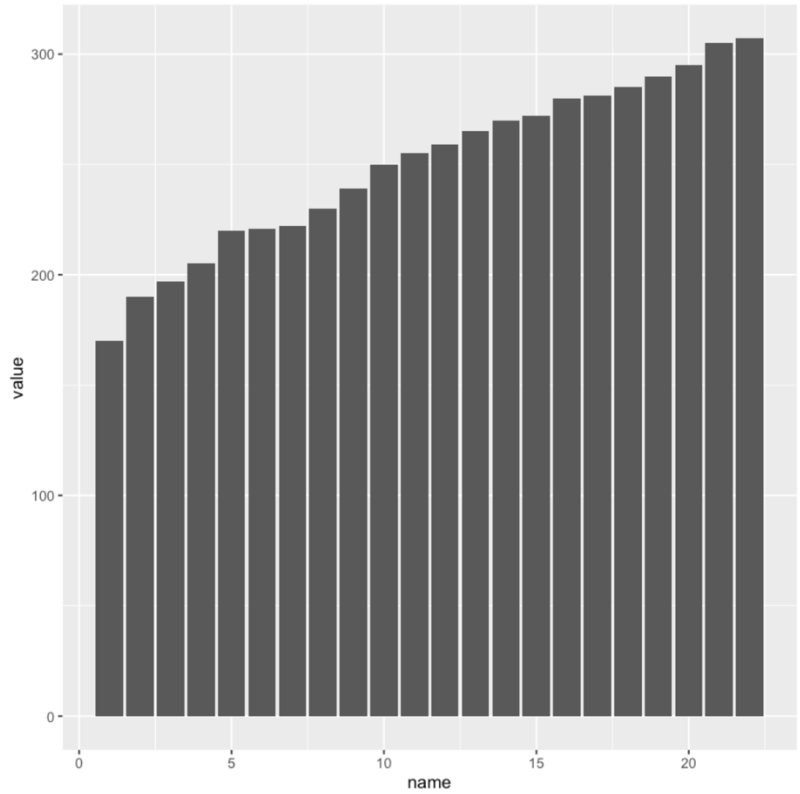


A performance measurement plot that compares the workflow execution time in response to an increasing data size (from 1 year to 22 years)

Changing dataset from 1 year to 22 years with 12 nodes

Years	Time
1	2 min 50 s
2	3 min 10 s
3	3 min 17 s
4	3 min 25 s
5	3 min 40 s
6	3 min 41 s
7	3 min 42 s
8	3 min 50 s
9	3 min 59 s
10	4 min 10 s
11	4 min 15 s
12	4 min 19 s
13	4 min 25 s
14	4 min 30 s
15	4 min 32 s
16	4 min 40 s
17	4 min 41 s
18	4 min 45 s
19	4 min 50 s
20	4 min 55 s
21	5 min 5 s

22	5 min 7 s



5. Oozie workflow graph: