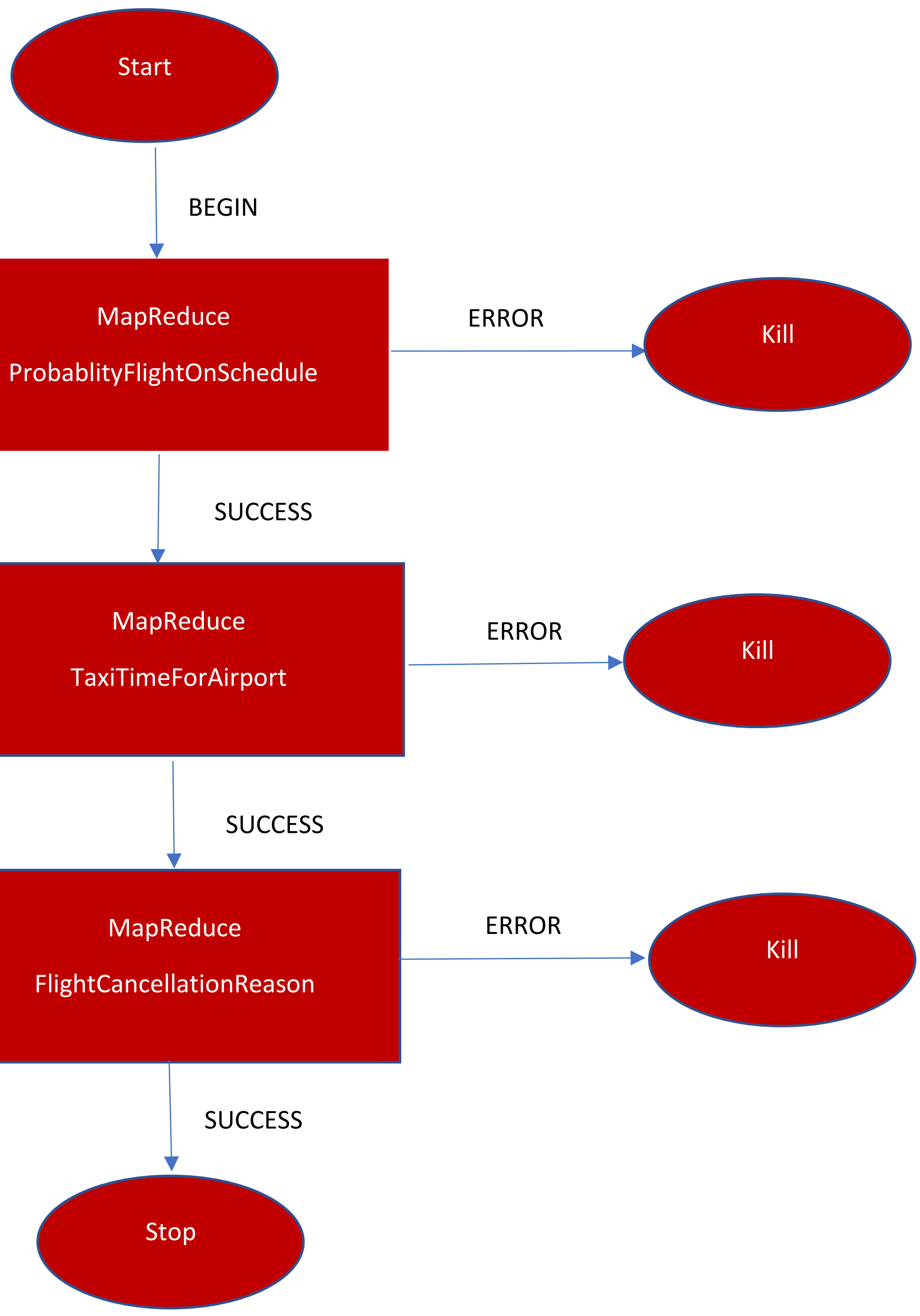
Course	:	CS 644 Introduction to Big Data
Document	:	Project Report
Topic	:	Flight Data Analysis
Submitted By	:	Chandini Ganesan (31454772)
		&
		Sakshi Pawar (31458871)
Batch	:	Spring 2019

OOZIE WORKFLOW



ALGORITHM

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

Mapper Phase:

1. Read input files line by line.
2. Split line based on comma and store all fields in an array
3. Fetch the values for fields corresponding to airlines, arrival delay and departure delay
4. We are considering two counts here. One for counting the airlines whose arrival delay and departure delay is 10 min and another for counting total number of airlines. These two counts are divided later to find the probability
5. Write to context<airlines, 1> and context<airlines_total, 1>

Combiner Phase:

1. Read context. Each Combiner will receive all the values corresponding to a key.
2. Calculate sum of all the airlines(filtered airlines which are delayed) and airlines_total(all airlines)

Reducer Phase:

1. Check whether the current key is airlines_total
    a. if yes, then get the total count for airlines and airlines_total. Then divide these values to find probability
    b. if no, then assign current key with airlines and get the total count of the airlines (filtered)
2. After finding the probability, the probability values are inserted into a tree map, where a custom comparator is written in order to sort the values based on probability
3. We have two tree maps. One for highest probability and another for lowest probability. We are removing the last and first values if the size of the tree map is greater than 3.
4. Write the output to context.

Calculate the 3 airports with longest and shortest average taxi time per flight

Mapper Phase:

1. Read input files line by line.
2. Split line based on comma and store all fields in an array
3. Fetch the values for fields corresponding to origin, destination, taxiIn and taxiOut
4. Write to context<origin, taxiOut> and context<destination, taxiIn>

Reducer Phase:

1. Read context. Each Combiner will receive all the values corresponding to a key.
2. Iterate over the context values which is list of taxiIn and taxiOut values for the airport.
3. Calculate sum of all the values and total number of values.
4. Calculate average taxi = sum / number of values.
5. Create a class AirportTaxiTime and airportname and average taxi time as instance variables and implement interface comparable.
6. Sort based on average taxi time in the compareTo method.
7. Create 2Treesets one for airports with highest taxi and anothr for airports withlowest taxi
8. We want just top3 and bottom 3 values so remove others using pollFirst and pollLast methods of Treeset.
9. Write the output to context.

Calculate the most common reason for cancellation of flight

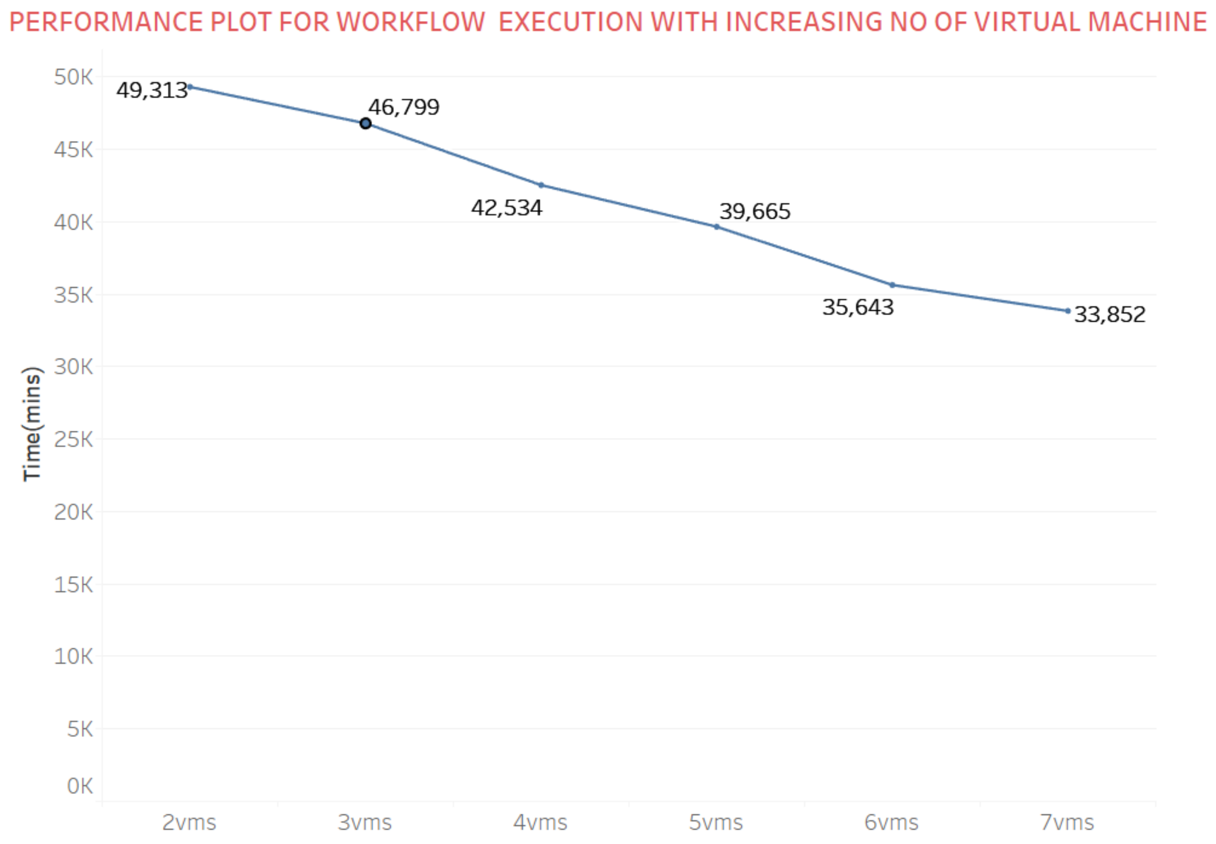
Mapper Phase:

1. Read input files line by line.
2. Split line based on comma and store all fields in an array
3. Fetch the values for fields corresponding to cancellationCode column
4. If cancellationCode corresponds to either A, B, C or D
5. Write to context<cancellationCode,1>

Reducer Phase:

1. Read context. Each Combiner will receive all the values corresponding to a key.
2. Iterate over the context values and Calculate sum of all the values.
3. Create a class with cancelCode and count as instance variables and implement interface Comparable.
4. Add the class objects to Treeset in order to maintain a sorted order.
5. Write the topmost reason to context

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)



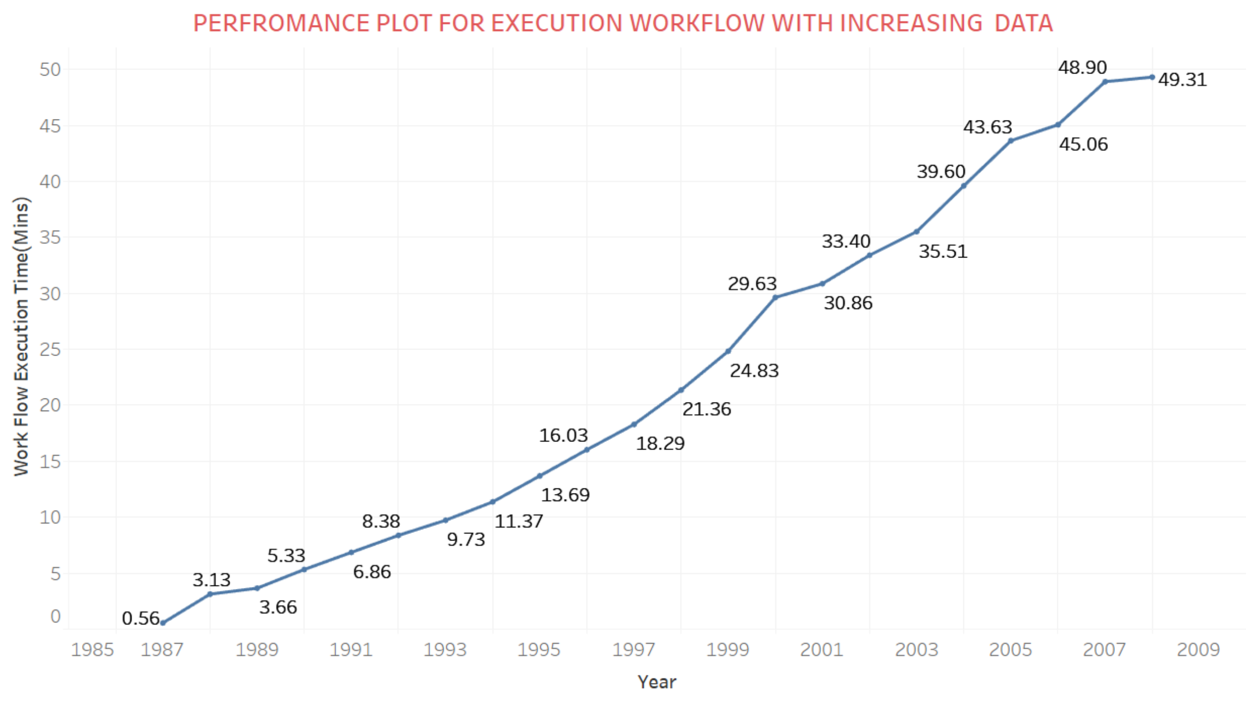
Here we are doing an experiment on performance of workflow having mapreduce jobs by varying the number of resources used. We are keeping data constant for all the runs i.e. flight data for all 22 years.

We are starting by using Hadoop on 2 Virtual machines. The total execution time taken is 49.3 mins. Next, we increases VM one at a time. We notice that as we increase the number of VMs there is a significant drop in time taken for execution.

CONCLUSION:

Therefore, we conclude that performance is directly proportional to number of virtual machine (resources) used to processing the big data.

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



In this experiment, we want to find out performance with respect to varying input data. We are using 2Virtual machines throughout this experiment.

First, we execute workflow on only one data file(1987.csv). We see that execution completes in very negligible time.

Next, we increase data by one year for every run and record the execution time. We observe that the execution time gradually increases as the input data increases.

CONCLUSION:
Hence, we can conclude that performance of our algorithm for the three given tasks is inversely proportional to input data.