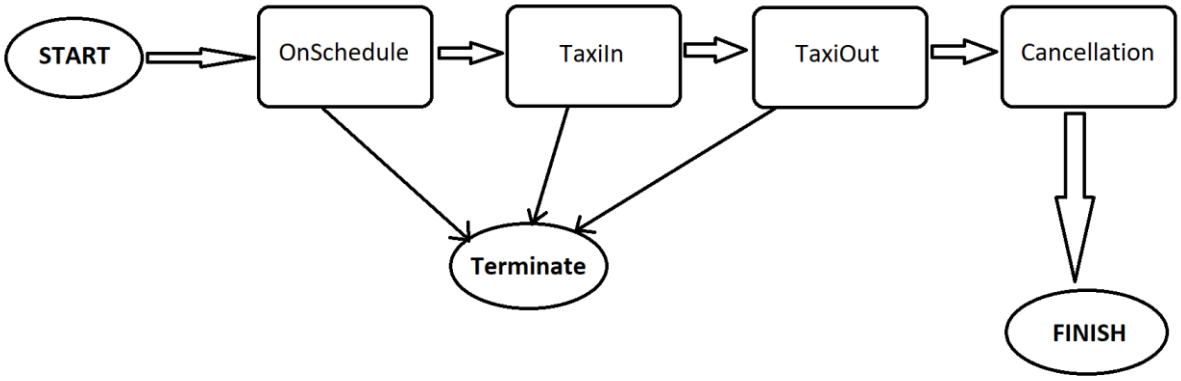
Oozie Workflow Diagram:

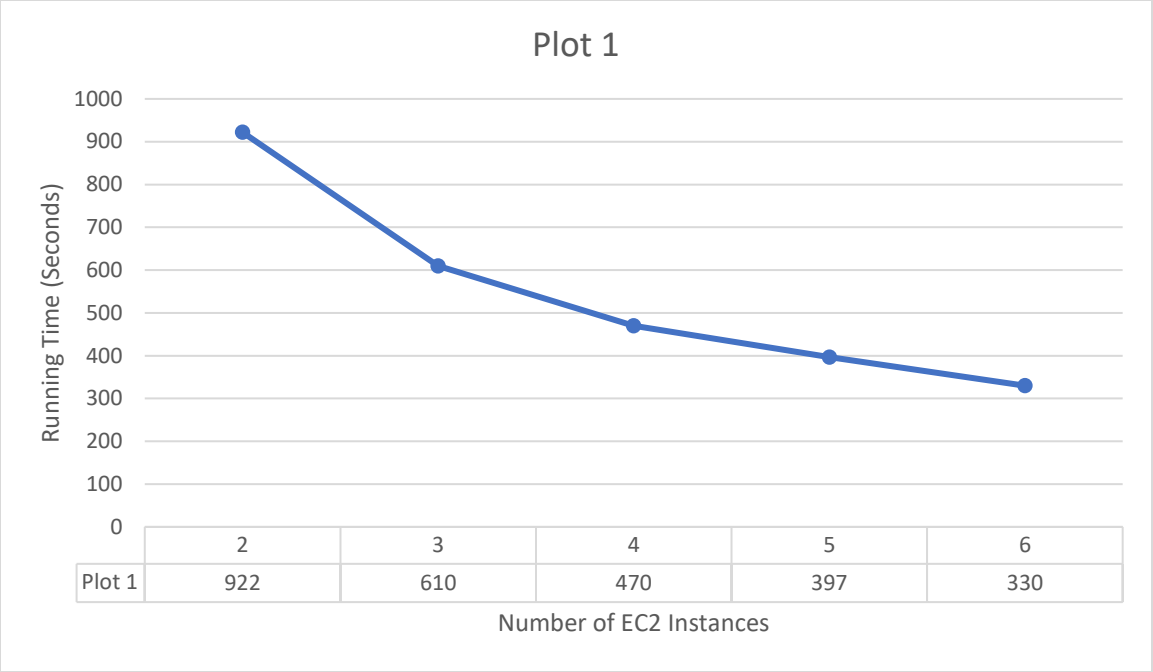


Algorithm:

1. The 3 Airlines with the highest and lowest probability, each, to be on schedule
    - For the flight to be on schedule, a benchmark of 10 has been set to determine if the flight is on schedule or not. If the Arrival Delay is greater than 10, it is classified as being late, i.e., not on schedule. Or else, the flight is on schedule.
    - The Mapper here takes the 'UniqueCarrier' column of the data as the key to write it to the context for the Reducer phase.
    - For each row, a small String "All: " is concatenated before the key and 1 is assigned as its value. And if the 'ArrivalDelay' column has a reading more than 10, a small String "Delayed: " is concatenated before the key. This is done simply to help the addition of each in the Reducer phase and calculate the probability.
    - The Reducer stores 6 instance variables for max & min (3 each) to keep record of the 3 maximum and 3 minimum flight carriers.
    - To get the total count, the Reducer checks if the key has the string "All: " in it and if it does, it calculates the total sum for that key. This is similarly done for the keys with the "Delayed: " string and the probability is taken out by dividing the two.
    - Now the 3 max and 3 min variables are updated according to the probability calculated for each row by a simple method call to the 6 methods.
    - Finally everything is written in to the cleanup function.

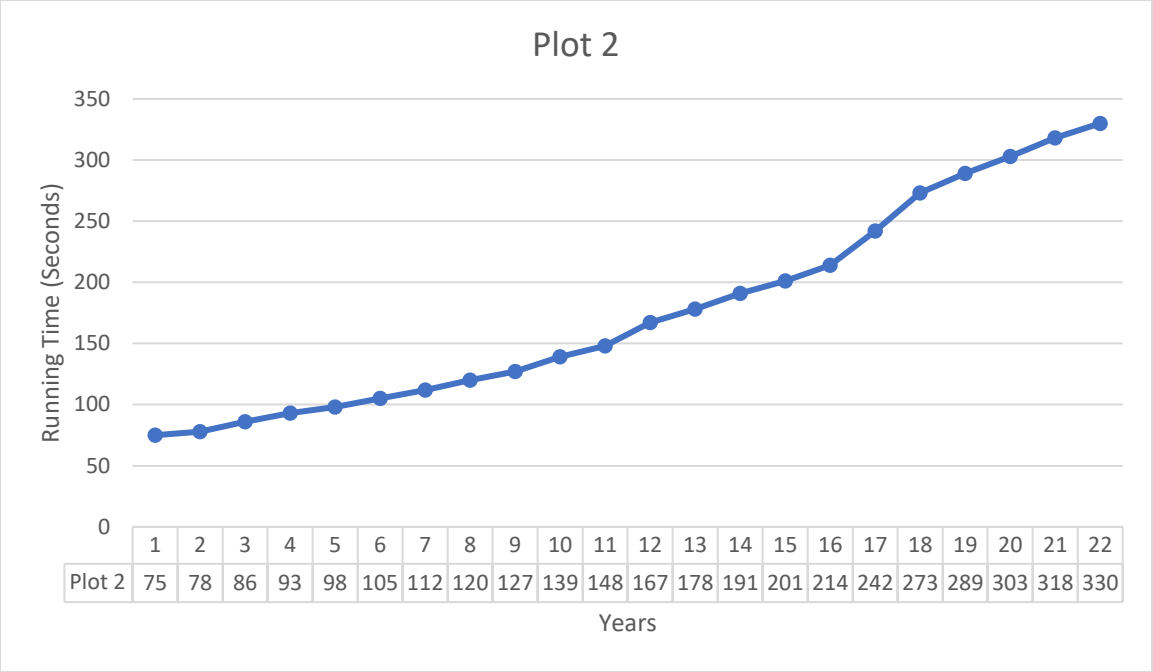
2. The 3 Airports with the longest and shortest taxi time per flight, each.
    - This program is done in 2 separate jobs, one each for Taxi In time and Taxi Out time. But the Reducer remains the same for both.
    - The Mapper function for Taxi In job does nothing but writes the “Dest” column of the data as the key to the context and “TaxiIn” column as the corresponding value.
    - Similarly, the Taxi Out job has “Origin” column as the key and “TaxiOut” column as the corresponding value.
    - The Reducer maintains the 3 max and 3 min values in the same way as the 1st Program, which is by maintaining 3 max and 3 min instance variables.
    - The Counter is run for each key and is updated accordingly.
    - Finally, the cleanup function is called, and all the data is written in to it for output.

3. The most common reason for flight cancellation.
    - This is a simple word count type of job.
    - The Mapper for this job writes into context the “CancellationCode” column of the data as the key and 1 as its value.
    - The Reducer does the counter for each key’s value and does the sum. The ‘maxCancellationCode’ instance variable maintains the maximum value as the code progresses.
    - The cleaup function in the end writes the maximum value in the context for the output.



The above graph shows the readings for the gradual increase in the EC2 instances beginning with 2 instances and increasing up to maximum of 6. The type of EC2 instance used here is t2.large. After close look at the graph, it can be deduced that there is a sharp decline in the running time when a 3rd instance is added. And then the decline continues as the number of instances are increased. The best running time that was calculated was with 6 instances of 330 seconds. This is about 1/3rd of the running time with 2 instances.

The graph will continue to keep showing decline as more and more resources such as number of VM instances and the types of instances, are available for the Hadoop.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Plot 2	75	78	86	93	98	105	112	120	127	139	148	167	178	191	201	214	242	273	289	303	318	330

The above graph shows the plot of the Running Time in seconds against the 22 Years. As the Year number is incremented and each file is added in the execution, naturally, the running time increases. It should be noted that the readings in this plot are in accordance with a maximum of 6 EC2 instances with t2.large type.

The graph shows a rise in the pattern when more and more files are added in the job. A single year file takes about 1 minute and 15 seconds to finish the job. Then there is a gradual increase as the number of files are incremented for the job.