Project Milestone-2

: Sai Dhiren Musaloji(sm3673)

Nithin Krishna Krishnappa(nk737)

Introduction:

In this project, an Oozie-based workflow has been developed which executes three MapReduce programs to analyze the flight data.

Each MapReduce program solves one of the following problems:

a. The airlines with the highest and lowest probability, respectively, for being on time.

b. The airports with the longest and shortest average taxi time per flight (both in and out), respectively.

c. The most common reason for flight cancellations.

A. On-Time Flights:

1. Start by setting a delay variable to a default value.
2. In the Mapper phase, add up the Arrival Delay and Departure Delay for each flight.
3. If the total delay is below the set threshold, label the flight as on schedule and emit a tuple indicating this.
4. If the total delay exceeds or equals the threshold, classify the flight as delayed and emit the corresponding tuple.
5. During the Reducer phase, count the total number of emitted values for each unique carrier.
6. Also in the Reducer, increment a count for on-time flights for each carrier.
7. Calculate the probability of flights being on schedule based on the counts.

8. Store these probabilities along with carrier information.

9. Repeat these steps for each unique carrier.

10. Arrange the stored data based on probability in descending order before writing it to storage.

B. Average Taxi Time:

1. For each flight in the Mapper phase, emit tuples containing Origin and Destination along with their respective taxi times.

2. In the Reducer phase, calculate the total count of flights for each airport.

3. Similarly, compute the total taxi time for each airport by summing up all taxi times.

4. Find the average taxi time for each airport by dividing the total taxi time by the total count of flights.

5. Store the calculated average taxi times along with airport information.

6. Sort the stored data based on average taxi times in descending order during cleanup.

7. Write the sorted airport information along with their average taxi times to storage.

C. Most Common Cancellation Cause:

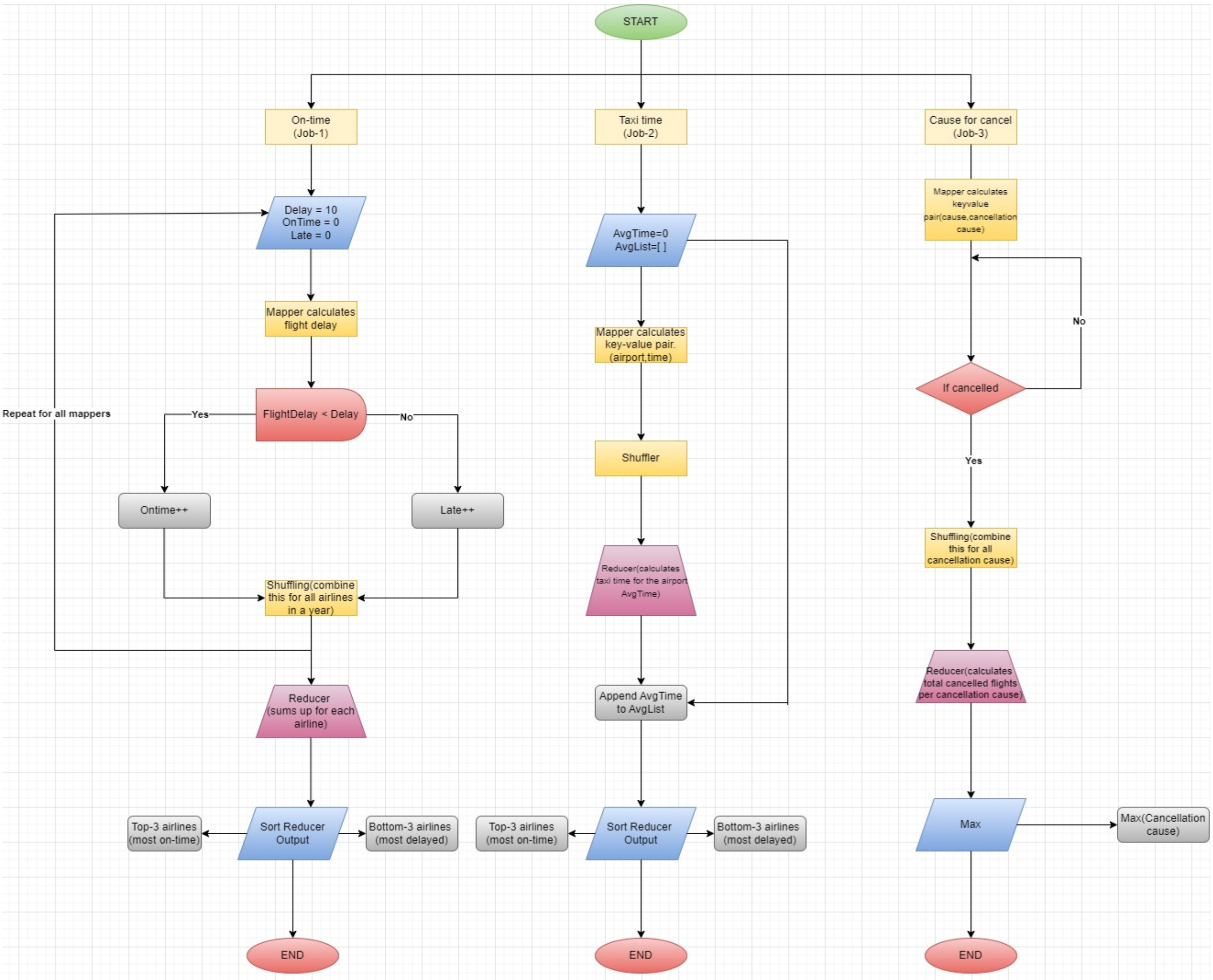
1. In the Mapper phase, emit a tuple indicating the cancellation cause if a flight is cancelled.

2. In the Reducer phase, aggregate the counts of each cancellation cause.

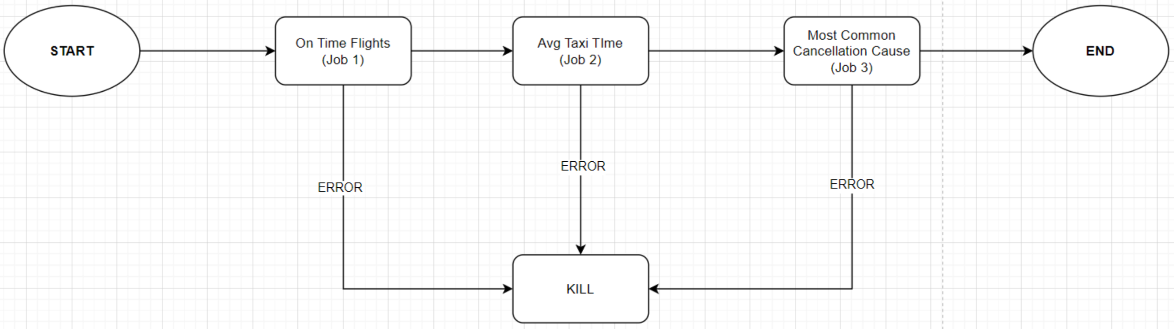
3. Write the total counts of each cancellation cause to storage.

4. Repeat these steps for all cancellation causes.

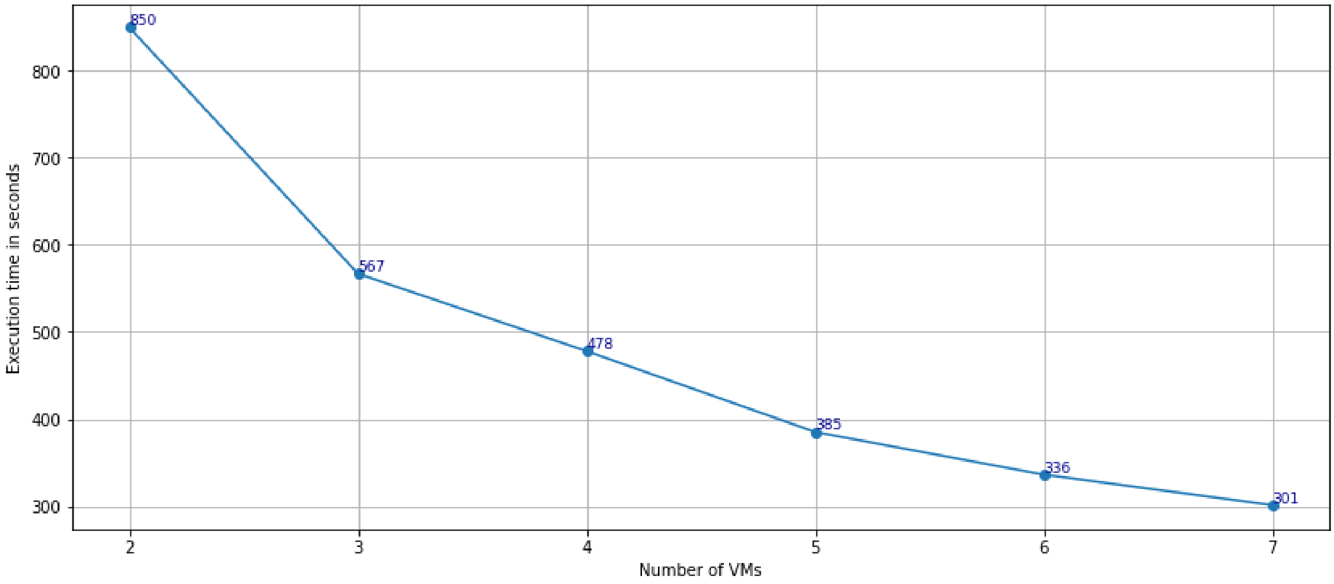
Flowchart


START
On-time
(Job-1)
Taxi time
(Job-2)
Cause for cancel
(Job-3)
Delay = 10
OnTime = 0
Late = 0
Mapper calculates flight delay
Repeat for all mappers
FlightDelay < Delay
Yes
No
Ontime++
Late++
Shuffling(combine this for all airlines in a year)
Reducer (sums up for each airline)
Sort Reducer Output
Top-3 airlines (most on-time)
Bottom-3 airlines (most delayed)
END
AvgTime=0
AvgList=[ ]
Mapper calculates key-value pair. (airport,time)
Shuffler
Reducer(calculates taxi time for the airport AvgTime)
Append AvgTime to AvgList
Top-3 airlines (most on-time)
Sort Reducer Output
Bottom-3 airlines (most delayed)
END
Mapper calculates keyvalue pair(cause,cancellation cause)
If cancelled
No
Yes
Shuffling(combine this for all cancellation cause)
Reducer(calculates total cancelled flights per cancellation cause)
Max
Max(Cancellation cause)
END

Simplified Flowchart



Performance Measurement Plot-1



The performance measurement plots illustrate the impact of increasing computational resources and data size on the workflow execution time for processing a large dataset using Hadoop/Oozie.

This graph depicts the relationship between the number of virtual machines (VMs) and the associated execution time in seconds. The x-axis represents the number of VMs, while the y-axis shows the execution time in seconds.

This plot compares the workflow execution time as the number of virtual machines (VMs) in the Hadoop/Oozie cluster is increased from 2 to 7. The graph shows a decreasing curve, indicating that as the number of VMs increases, the execution time decreases.

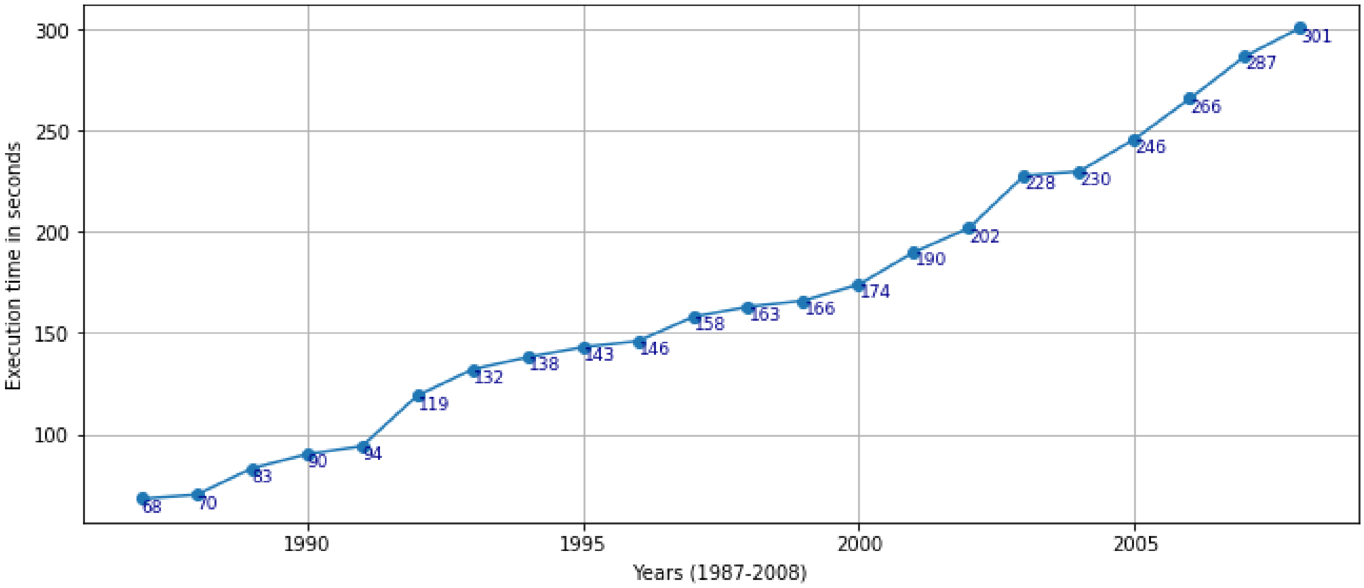
The key observations are:

- The execution time drops sharply when the number of VMs is increased from 2 to 3, indicating the benefits of parallel processing.
- After 3 VMs, adding more VMs leads to an almost linear decrease in execution time.

- With 7 VMs, the total execution time is 301 seconds, which is an improvement by a factor of around 2.83 compared to the initial 2 VM cluster (850 seconds).[1]

The decrease in execution time with more VMs is consistent with the theory of parallel processing, where additional computational resources enable faster processing of the data.

Performance Measurement Plot-2



This plot compares the workflow execution time as the data size is increased from year 1987 to 2008, using a maximum of 7 VMs.

The key observations are:

- As the data size increases, the processing time increases accordingly.
- The execution time increases by a factor of approximately 4.4 (from 68 seconds to 301 seconds) when the data size is increased from 1 year to 22 years.

With fixed computational resources, increasing the data size requires more processing to compute the final output, leading to longer execution times.

In summary, these plots demonstrate the trade-off between computational resources and data size in determining the overall workflow execution time for big data processing using Hadoop/Oozie.