**CS 644 Introduction to Big Data**

**Project: Flight Data Analysis**

**Fall 2023**

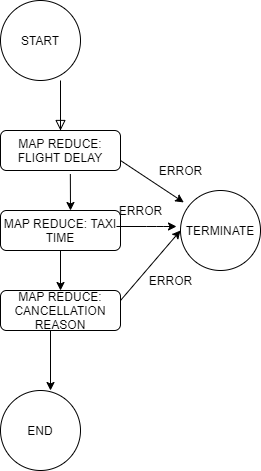
Submitted by: Craig Bernier UCID: #####

David Cancel

Ranjan Kumar

Pavel Zhinhel

# Oozie Workflow Diagram



# Algorithm Description

## On Schedule Flights

* Establishing delay threshold as delay\_threshold = 5.
* For each flight, the Mapper program tags flights with a delay time above 5 minutes threshold as delayed flights.
* The Reducer algorithm counts on time and delayed flights. Calculate total number of 0 and 1. Find probability of on schedule or not.
* The reducer sorts "Highest 3 Probability" and "Lowest 3 Probability" and sends this record to output.
* At cleanup, sort the arraylist in decreasing order of probabilities.

## Average Taxi Time

* For each flight, the Mapper groups origin airport and taxi in time; destination airport and taxi out time (origin, taxiOutTime) & (dest, taxiInTime).
* Reducer counts the total number of values as total\_count for each airport entry key.
* Reducer computes the total taxi time by adding all the values as taxiTimeVal.
* Compute airport average taxi time by dividing the total taxi time obtained in steps above by the total\_count.
* Write the arraylist of FlightTaxiTime to HDFS.

## Most Common Cancelation Reason

* For each flight, the Mapper emits CancellationCode. If the flight was canceled then Cancelled =1.
* Reducer looks at each CancellationCode key and adds all the values to get the cancel\_count.
* Writing CancellationCode and total\_count to HDFS.

# Performance Assessment I

Compares the workflow execution time in response to an increasing number of VMs used for processing the entire data set (20 years).

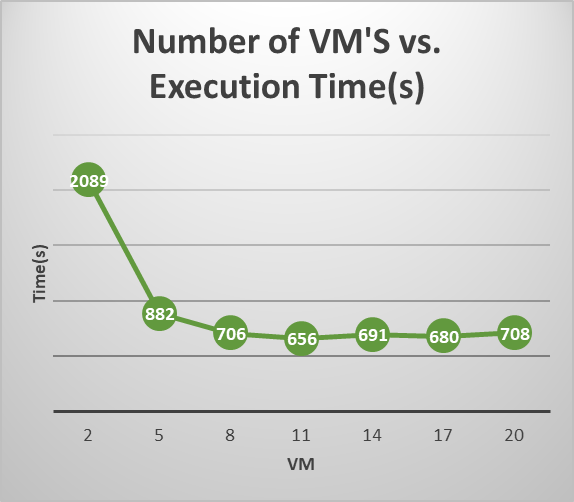


Figure 1 - HDFS Performance by VM number (based on Table 1)

In order to gauge the HDFS cluster performance, the workflow was run for the range of VMs. The minimum number of VMs used is 2. That number was increased to the total of 20 as additional 3 VMs were added in series to the Hadoop/Oozie cluster.

As shown in Figure 1, the execution time drops sharply as VM’s are increased from 2 to 5. After the 5 VMs mark, there is a much more gradual decrease in execution time before the curve becomes virtually flat after 8 VMs. For a cluster with 8 VMs, the total time is 708 seconds, which is an improvement by a factor of around 2.96 in comparison to that of the initial cluster with only 2 VMs (2089 seconds). See Table 1 below for workflow run data.

The above workflow execution times are consistent with the theory. Additional VMs contribute to parallelism, allowing tasks to be distributed across more nodes. HDFS decreases execution time by processing data in parallel. Ultimately, the impact of adding more VMs on execution time can vary based on factors like the nature of the job, data distribution, network configuration, and how Hadoop is configured to utilize these additional resources.

Table 1: Workflow Execution Time per Virtual Machine(s)

| **VM** | **Time(s)** | **Mins** | **Secs** |
| --- | --- | --- | --- |
| **2** | 2089 | 34 | 49 |
| **5** | 882 | 14 | 42 |
| **8** | 706 | 11 | 46 |
| **11** | 656 | 10 | 56 |
| **14** | 691 | 11 | 31 |
| **17** | 680 | 11 | 20 |
| **20** | 708 | 11 | 48 |

# Performance Assessment II

Compares the workflow execution time in response to an increasing data size (from 1 year to 21 years).

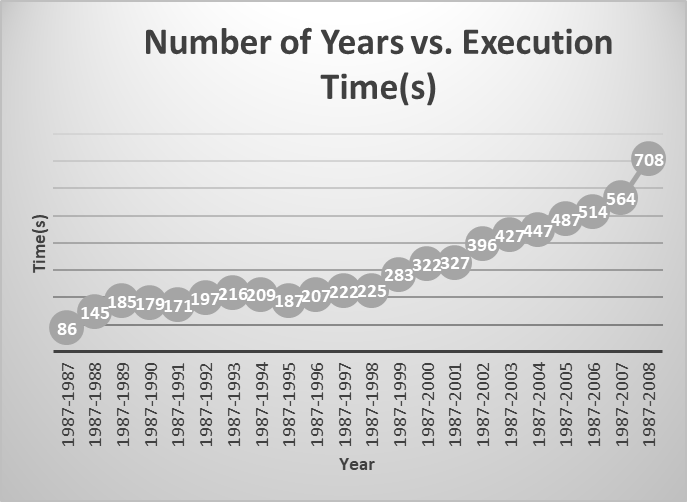


Figure 2 - HDFS Performance by Data File (based on Table 1)

Alternative HDFS cluster performance assessment was run with varying data volume. The minimum amount of data corresponds to one year (1987). That number was increased by the increment of 1 as subsequent year data from 1987 through 2008 was added to analysis.

As shown in Figure 1, the execution time rises sharply within the first three years. After the 1989 mark, the curve became generally flat indicating system resistance to increasing data volume. However, the execution time curve continued to slope upward after 1999. The total 21 year execution time is 708 seconds, which is an increase by a factor of around 8.23 in comparison to that of the single year data (86 seconds). See Table 2 below for workflow run data.

This can be understood from the fact that since the computational resources are fixed in this case, increasing the data size will require more processing to compute the final output.

Table 2: Workflow Execution Time per Record Year(s)

| **Year** | **Time(s)** |  |
| --- | --- | --- |
| **1987-1987** | 86 | 1 minutes 26 seconds |
| **1987-1988** | 145 | 2 minutes 25 seconds |
| **1987-1989** | 185 | 3 minutes 05 seconds |
| **1987-1990** | 179 | 2 minutes 59 seconds |
| **1987-1991** | 171 | 2 minutes 51 seconds |
| **1987-1992** | 197 | 3 minutes 17 seconds |
| **1987-1993** | 216 | 3 minutes 36 seconds |
| **1987-1994** | 209 | 3 minutes 29 seconds |
| **1987-1995** | 187 | 3 minutes 07 seconds |
| **1987-1996** | 207 | 3 minutes 27 seconds |
| **1987-1997** | 222 | 3 minutes 42 seconds |
| **1987-1998** | 225 | 3 minutes 45 seconds |
| **1987-1999** | 283 | 4 minutes 43 seconds |
| **1987-2000** | 322 | 5 minutes 22 seconds |
| **1987-2001** | 327 | 5 minutes 27 seconds |
| **1987-2002** | 396 | 6 minutes 36 seconds |
| **1987-2003** | 427 | 7 minutes 07 seconds |
| **1987-2004** | 447 | 7 minutes 27 seconds |
| **1987-2005** | 487 | 8 minutes 07 seconds |
| **1987-2006** | 514 | 8 minutes 34 seconds |
| **1987-2007** | 564 | 9 minutes 24 seconds |
| **1987-2008** | 708 | 11 minutes 48 seconds |