Flight Data Analysis - Ozzie Workflow

Milestone 2 - DS644-004

Team
Udeda.Suchithra - Su252
Sanjana Akula - Sa2844
Alagani Sai Krupesh Goud - Sa2834

1.A diagram that shows the structure of your Oozie workflow

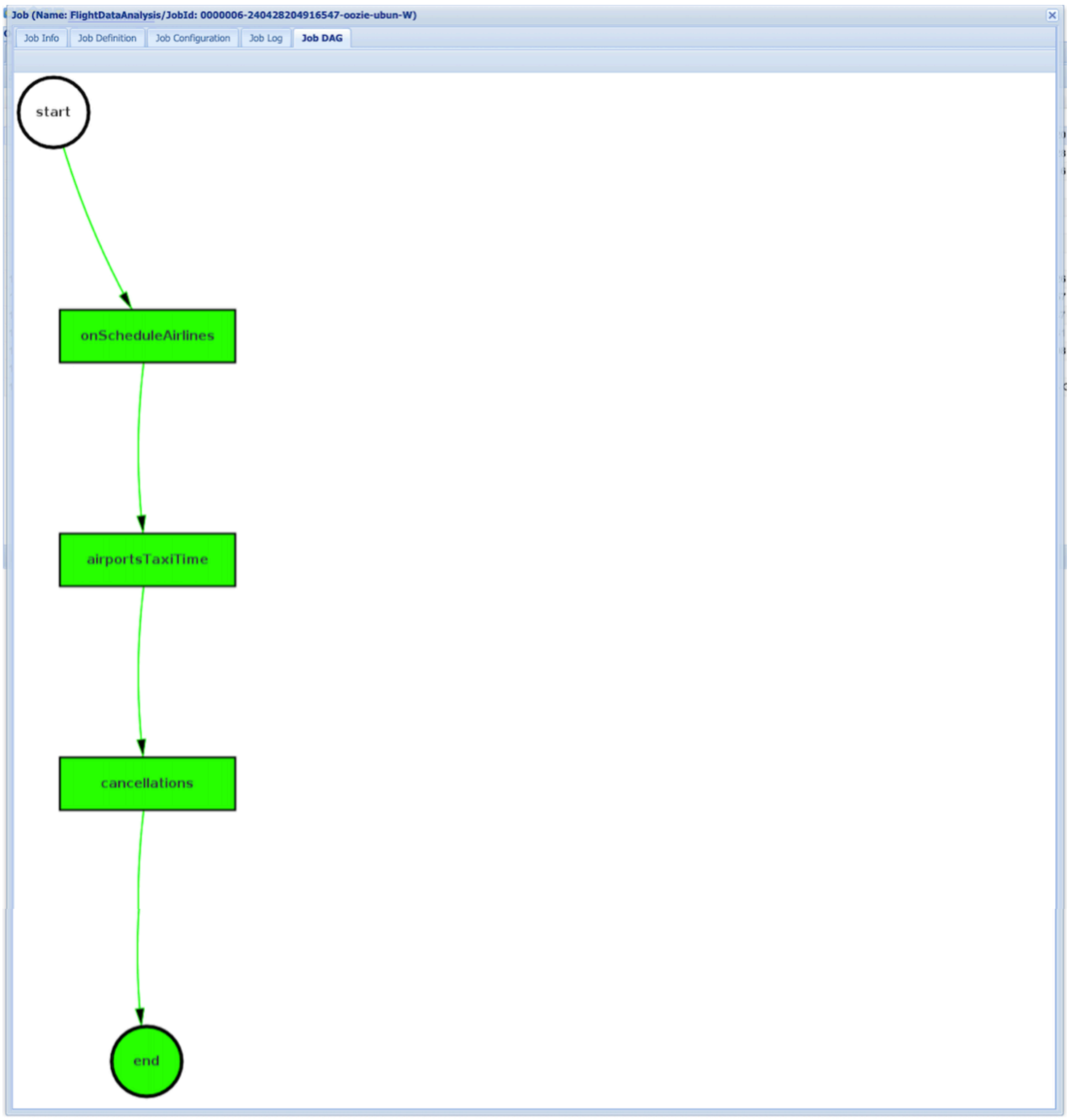

Figure 1

2.A detailed description of the algorithm you designed to solve each of the problems

a) Airline on-time performance

1.Mapper
- Key-value pairs, with the airline code as the key and the on-time indicator as a tuple (1 for on-time, 0 for delayed), should be sent for each flight record.

2.Reducer
- Compute each airline's total number of flights and total number of on-time flights.
- Calculate the on-time percentage (total on-time flights / total flights) for each airline.

3.Sorting
- The airlines are sorted according to their on-time %; the highest likelihood airlines are found by sorting them in ascending order, while the lowest probability airlines are found by sorting them in descending order.

b)Airports taxi time analysis

1.Mapper
- Emit key-value pairs for every flight record, where the airport code is the key and the value is a tuple with the taxi-in and taxi-out times.

2.Combining
- To maximise efficiency, combine intermediate records.

3.Reducer
- Add up all of the taxi arrival and departure times for every airport.
- Determine how many flights there are in total for each airport.
- Calculate the average taxi time (total taxi time / total flights) for each airport.

4.Sorting
- To determine which airports have the longest average taxi times per flight, sort the airports based on their average taxi times in descending order; to find the airports with the lowest average taxi times per flight, sort them in ascending order.

c)Cancellation reasons analysis

1.Mapper
- Emit key-value pairs, where the key is the cause for the cancellation and the value is 1, for each flight record that has been marked as cancelled.

2.Combining
- To maximise efficiency, combine intermediate records.

3.Reducer
- Add up all of the cancellation reasons.
- Find the cancellation reason that has the largest number

3.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) and an in-depth discussion on the observed performance comparison results

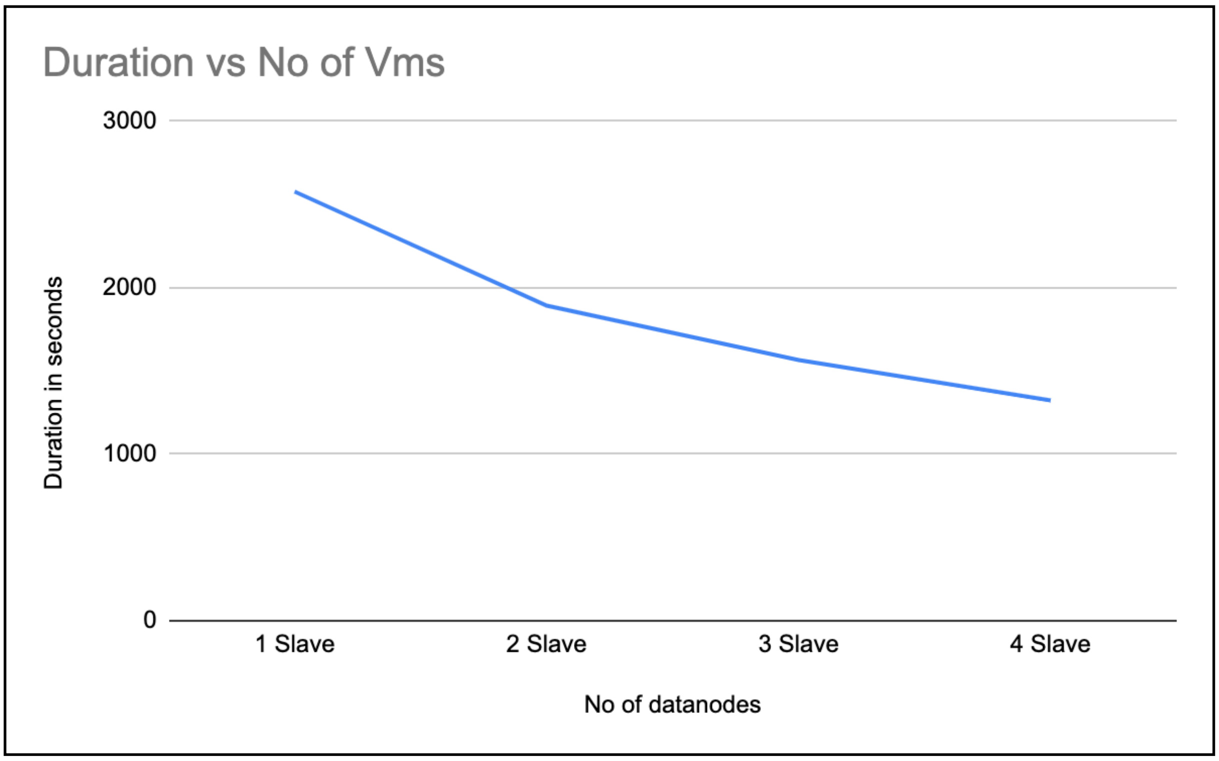

Figure 2

The link between the execution time of the workflow and the increase in the number of virtual machines (VMs) used to process the whole 22-year dataset from the given source is depicted in the performance measurement figure. The time required for workflow execution is predicted to decrease as the number of VMs rises, demonstrating the capacity of numerous VMs for parallel processing.

After the plot is examined, a number of observations become clear. Because the burden is spread among the available resources, there may be a notable reduction in execution time at first when there are fewer virtual machines (VMs). The additional virtual machines (VMs) may, however, have a diminishing return phenomenon above a certain level, whereby they contribute less to the total performance improvement.This could be caused by things like resource contention, connection slowness, or the overhead of managing virtual machines.

In addition, the plot can show ups and downs in performance, which could point to resource constraints or bottlenecks in the system. Further workflow execution time optimisation may result from the identification and removal of certain bottlenecks. Furthermore, the plot might shed light on the efficiency and scalability of the processing framework used for the dataset, assisting in choices on how best to allocate resources and design the system architecture for maximum performance.

4. A performance measurement plot that compares the workflow execution time in response to an increasing data size (from 1 year to 22 years) and an in-depth discussion on the observed performance comparison results

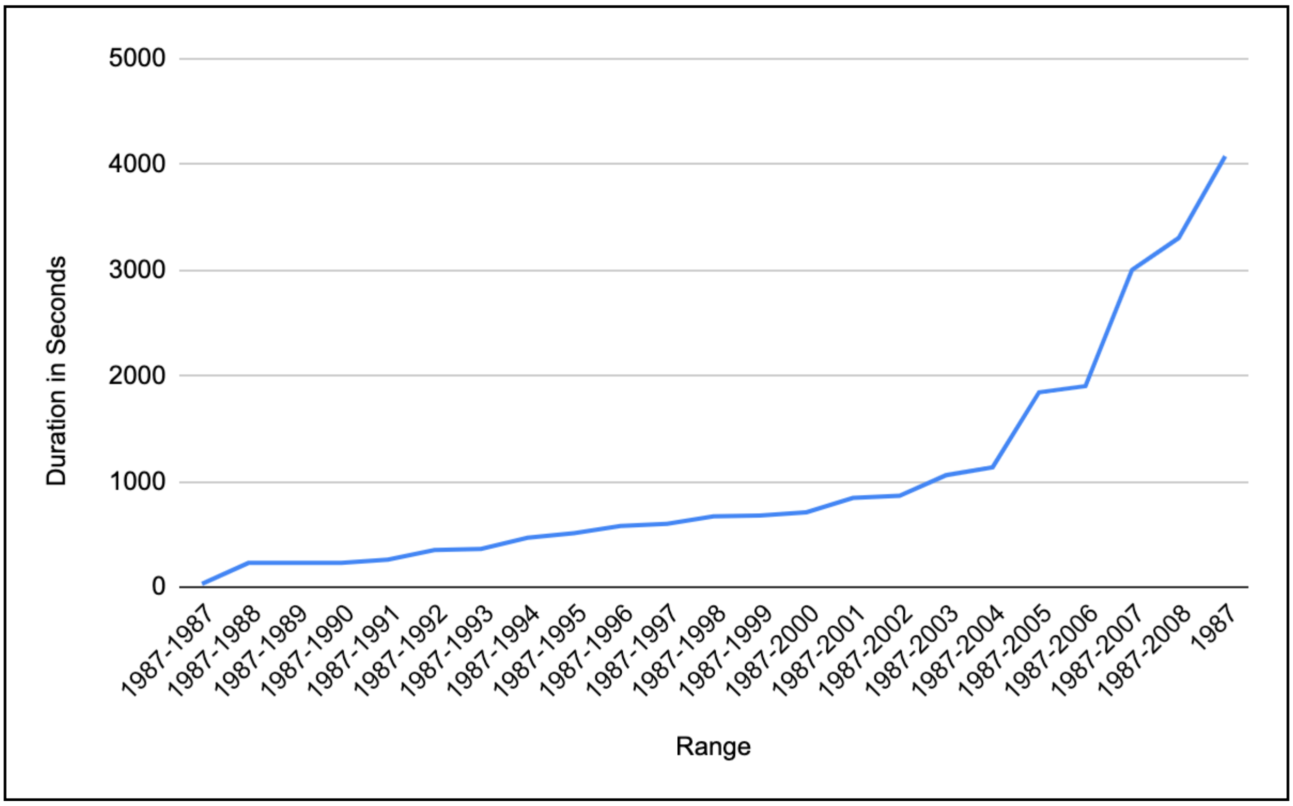

Figure 3

The provided dataset illustrates the performance measurement plot, which presents the workflow's execution time in response to different data sizes, spanning from one year to twenty-two years. This graphic illustrates how the efficiency of the workflow increases with the amount of input data.

Plot analysis allows for the identification of various key points. At first, there is a discernible linear increase in execution time with increasing data size, suggesting a corresponding rise in computational workload. Nevertheless, the execution time exhibits a more marked increasing trend beyond a given point, usually around the 10 to 15-year mark, indicating declining performance efficiency returns. System overhead, resource limitations, and algorithmic complexity are a few possible explanations for this phenomena.

In addition, the graphic makes it possible to compare various data processing approaches or system configurations, which facilitates well-informed decision-making about resource allocation and workflow optimisation. Through an understanding of the correlation between execution time and data size, organisations can develop tactics aimed at alleviating performance bottlenecks and optimising overall workflow efficiency in data processing.

5. Environment variables of master and slaves
~/.bash_profile

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/home/ubuntu/hadoop-2.6.5
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export M2_HOME=/home/ubuntu/apache-maven-3.5.3
export PATH=$PATH:$M2_HOME/bin
export OOZIE_HOME=/home/ubuntu/oozie-4.1.0
export OOZIE_CONFIG=$OOZIE_HOME/conf
export CLASSPATH=$CLASSPATH:$OOZIE_HOME/bin
```