NJIT New Jersey Institute of Technology

DS 644
INTRODUCTION TO BIG DATA
PROJECT REPORT

TEAM MEMBERS:

ANOUSHKA REDDY BOBBALA	ab2756
NITYA SRI MATTA	nm873
YESHWANTH KOTHA	yk342

OOZIE WORKFLOW STRUCTURE -

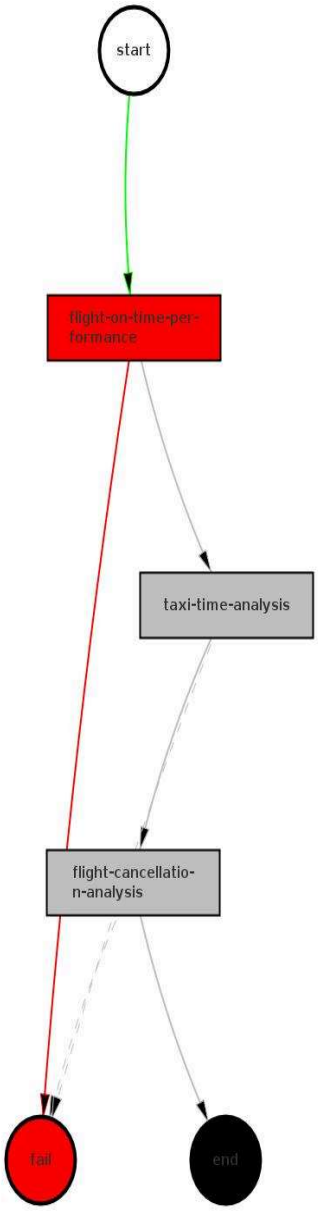


fig: When a task fails

The workflow starts at the top with a **start node** and immediately moves to a flight-on-time-performance action node, the task is to measure data related to flight's punctuality.

The green arrow indicates the success and leads to the next node indicating that the workflow has been successfully completed without any issues.

The red arrow indicates a failure or error in the path from flight-on-time-performance, now it branches into two possible paths.

- One path is to taxi-time analysis node as a secondary path and next to flight cancellation analysis.
- Another path is directly to flight cancellation analysis where analysis of flight cancellations occur.

If either taxi-time-analysis or flight cancellation analysis nodes fail (indicated by dashed lines) the workflow will proceed to a fail node . Which suggests that workflow has encountered an error or terminated.

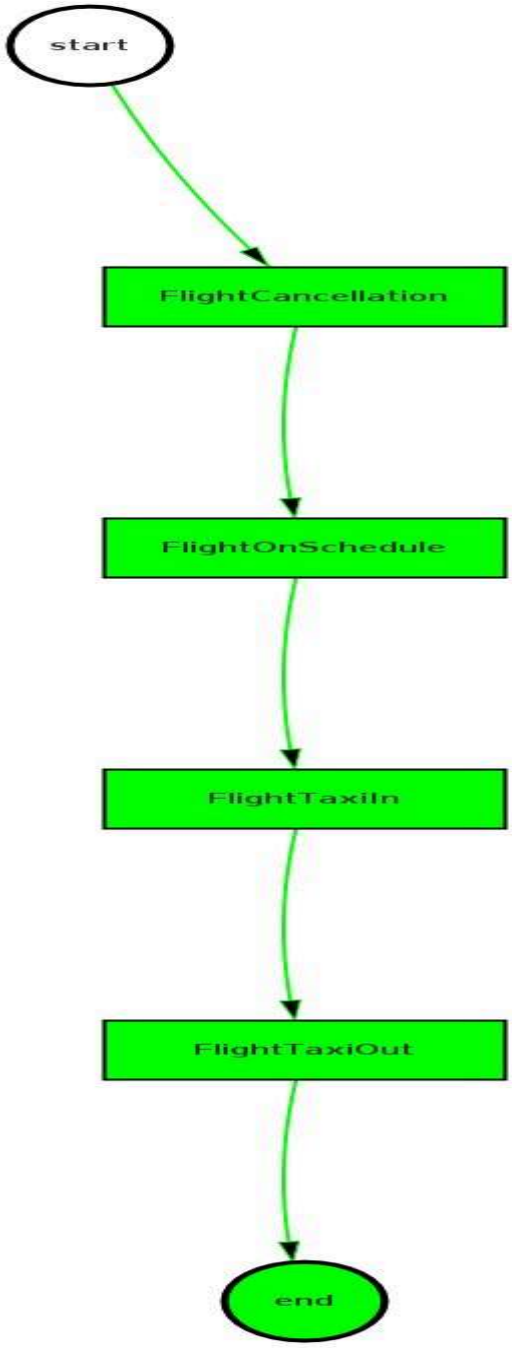


fig: When all tasks are executed successfully.

1. The above workflow starts at start node and follows linear sequence of tasks without branching for errors.
2. All the actions in this workflow are connected with green arrows suggesting each step is executed successfully in sequence.

3. This workflow doesn't have branches for handling errors where it is assumed that each step is completed successfully.

ALGORITHM -

Job - 1 : **On-time performance analysis**

1. Create a Mapper('MapperOne') -
    1. Input the flight data
    2. Split the lines into fields using a comma(' , ').
    3. Check the 15th field for delay time. If it is a valid integer, determine if the delay is between -5 and 5 minutes.
    4. Output key-value pairs, where the key is concatenated airline code and flight number and if the value is 1 if it is on-time and 0 if it is not.
2. Create a Reducer('ReducerOne') -
    1. Input the airline flight codes and list of 1's and 0's
    2. for every key count the total number of flights and number of on-time flights.
    3. calculate the on-time percentage
    4. And maintain two lists where one is used for highest on-time percentages while the other one is for the lowest on-time percentages and use a sorting mechanisms to keep inly the top three in each
    5. Output the top three and bottom three airlines based on on-time performance.

Job - 2 : Taxi Time Analysis

1. Create a Mapper('MapperTwo') -
    1. Input the flight data
    2. Split the lines into fields using a comma(' , ').
    3. Retrieve the airport code and the taxi-in time if it is available
    4. Output key-value pairs, where the key is airport code and the value is taxi-in time
2. Create a Reducer('ReducerTwo') -
    1. Input the airport codes and list of taxi-in times
    2. for each airport, compute the average taxi-in time from the collected times
    3. Emit each airport and its average taxi-in time.

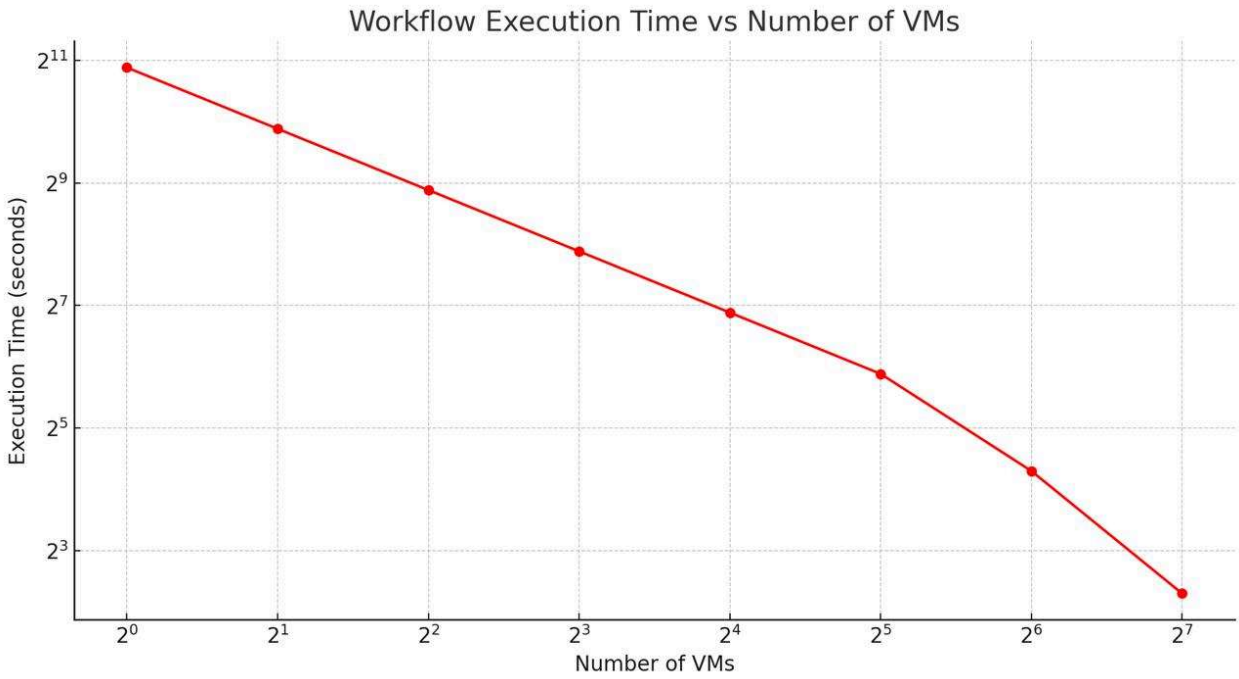
Job - 3 : Flight Cancellation Analysis

1. Create a Mapper('MapperThree') -
    1. Input the flight data
    2. Split the lines into fields using a comma(' , ').
    3. Retrieve the airlines code and the cancellation status

4. Output a key-value pair where the key is the airline code and the value is 1 if the flight was canceled and 0 otherwise.

2. Create a Reducer('ReducerThree') -
   1. Receives airlines codes and list of 1's and 0's
   2. Count the total number of flights and number of cancellations for each of the airline given
   3. Emit each of the airline and its cancellation rate.

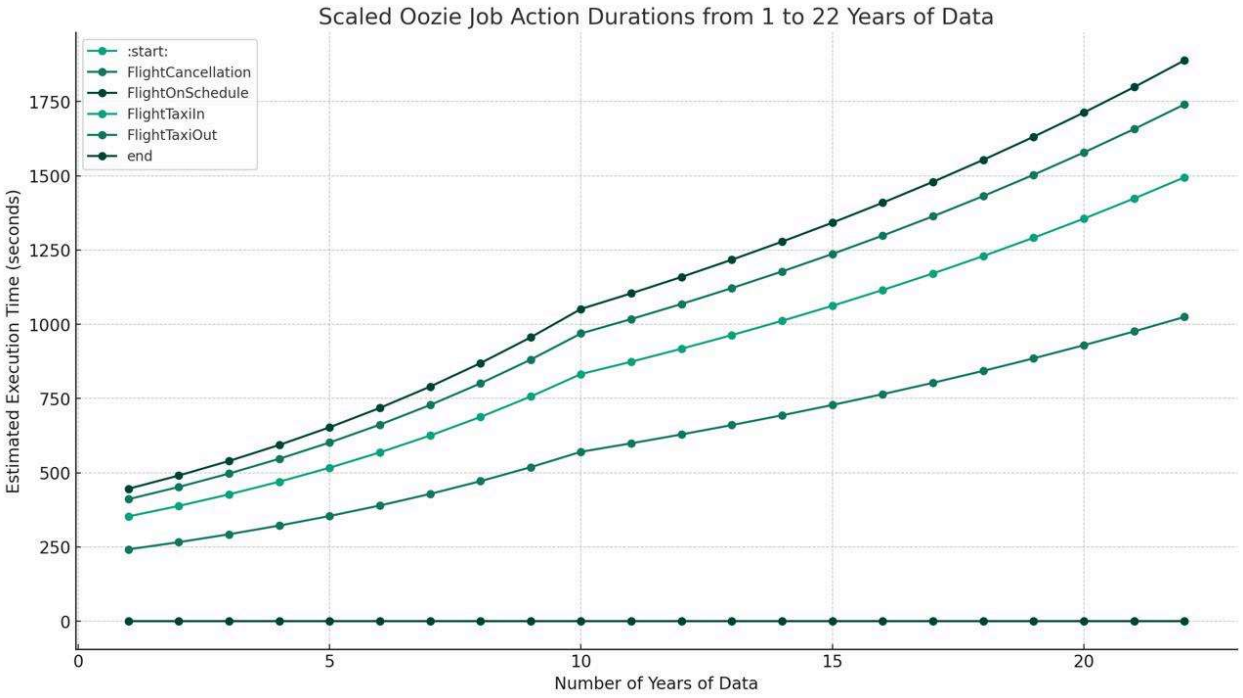
A performance measurement plot that compares the workflow execution time in response to an increasing number of VMs used for processing the entire data set (22 years) and an in-depth discussion on the observed performance comparison results



The X-axis represents the number of VMs on a logarithmic scale of base 2, ranging from 2^0 to 2^7 (1 to 128 VMs). The Y-axis shows the execution time, also on a logarithmic scale of base 2, but the units are unspecified (it could be seconds, milliseconds, etc.). From the plot, we observe a trend where increasing the number of VMs reduces the execution time significantly. This is a common phenomenon in distributed computing, where tasks can be parallelized across multiple machines to decrease the total time taken to complete a job. The performance gain appears to be diminishing as the number of VMs increases; while there's a steep drop from 2^0 to 2^3 VMs, the curve flattens out somewhat as we move towards 2^7 VMs. This could be

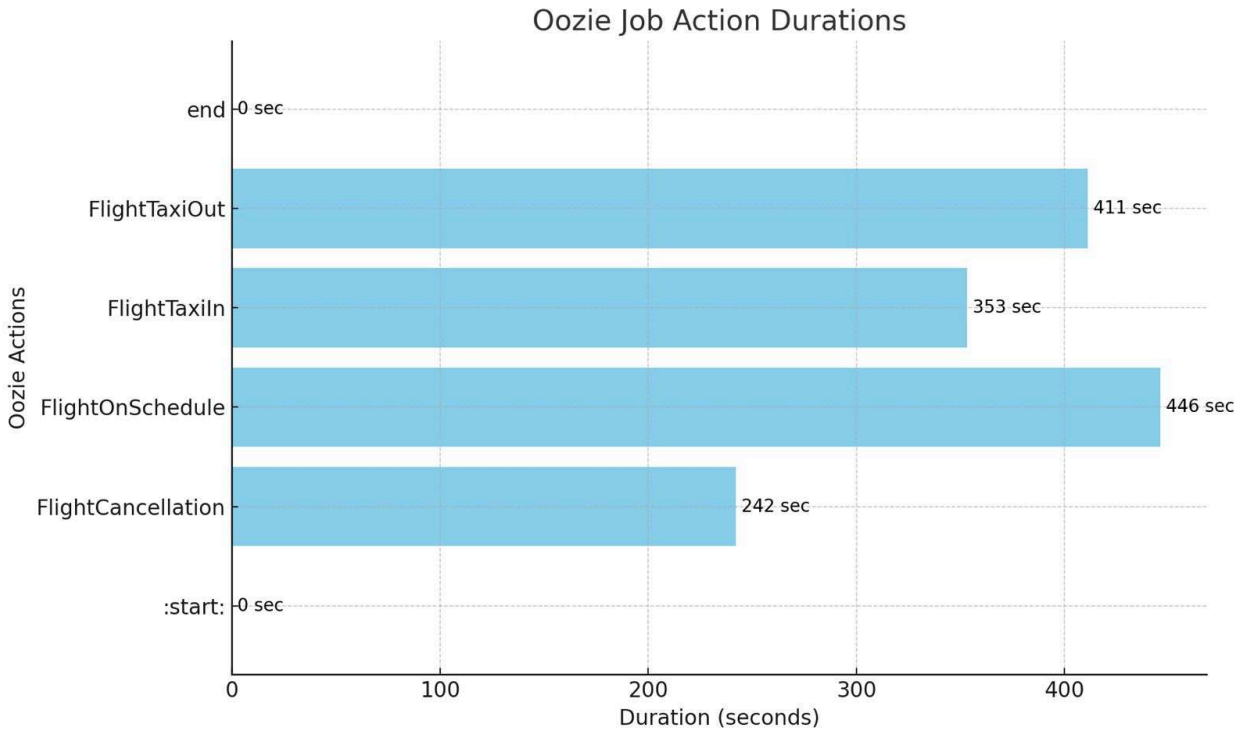
due to various factors such as network overhead, synchronization costs, or that some parts of the workflow cannot be parallelized further (also known as Amdahl's Law)

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.



This plot shows an estimated execution time for the various actions within an Oozie job with the size of the dataset running from 1 to 22 years of data. Oozie is a Hadoop workflow scheduler for jobs, and the various actions probably represent a variety of workflows or processes that make up a complete job sequence.

There are three different actions shown on the graph: starting the job, processing flight cancellations, flights on schedule, flights taxiing in and out, and ending the job. The X-axis shows the number of years of data being processed, and the Y-axis represents the estimated execution time in seconds.



The horizontal bar chart represents durations of the various actions in an Oozie job. Oozie is a workflow scheduler for Hadoop.

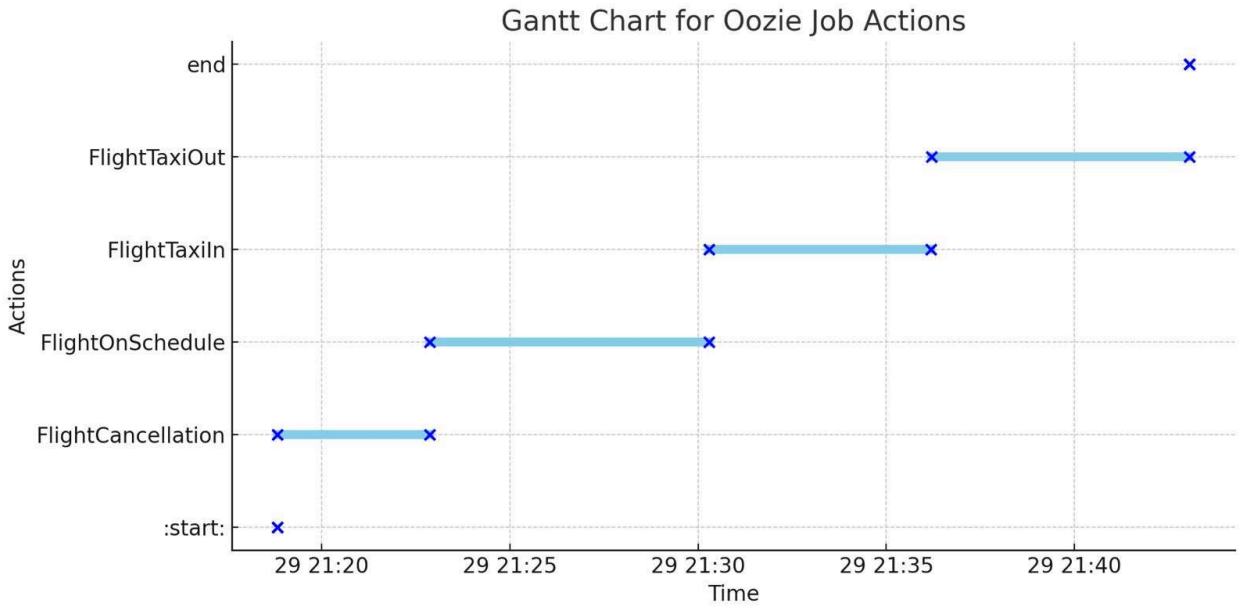
Each of these bars represents an Oozie action. The length of the bar shows how long the particular action takes to complete, with the time in seconds mentioned at the end of each bar.

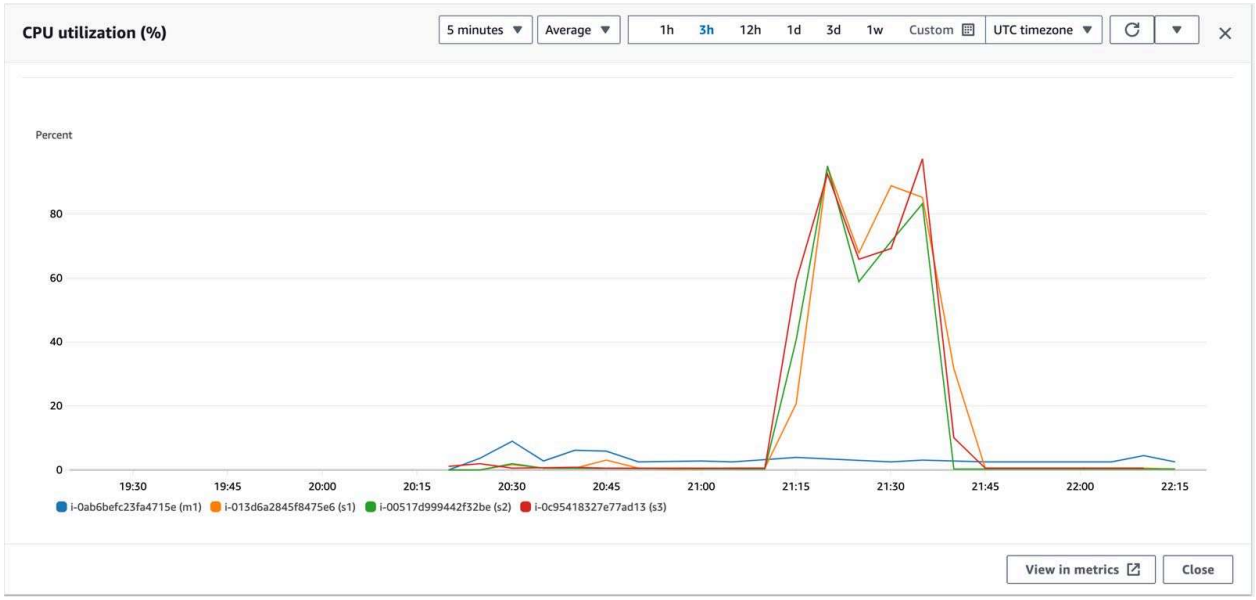
Now let's break each Oozie action down and its duration:

- ````**:start:**`: This action shows that the Oozie workflow begins execution, a state that had a duration of 0 seconds, meaning it effectively starts instantly without any delay.
- ```**FlightCancellation**`: This action took 242 seconds to execute. It probably deals with data on flight cancellations.
- ```**FlightOnSchedule`:** This action took 446 seconds to execute, which is by far the longest action in this workflow. This indicates that this action might have a lot of data, may be quite complex, or be requiring some processing of complex data.
- ```**FlightTaxiIn**`: The execution time of this action was 353 seconds. Presumably, data processing for the flight taxiing in process goes here.

- ```**FlightTaxiOut`:** This action took a slight longer time than taxiing in, with a duration of 411 seconds. It suggests that the data or the processing steps for flights taxiing out are slightly more complex or numerous.
- ```**end`**: This is the end of the Oozie workflow, similar to the start, it had a duration of 0 seconds, which indicates that it instantaneously stops the workflow.

This chart is useful in providing a quick comparison of the time each section of the Oozie job takes. For example, with this chart, one can view at a glance which actions are the most time-consuming and hence potential bottlenecks or places where optimization might be needed for this workflow. From an example whereby the longest action was ```FlightOnSchedule`—workflow optimization efforts would likely focus on this action first.





Results:

1. **3 airlines with the highest and lowest probability, respectively, for being on schedule**

Airlines with Higher Probability:
WN 0.07405553753405504
EA 0.050223693580565026
AS 0.03368800966866945
Airlines with Lowest Probability:
B6 0.0017986665260542997
9E 0.0019901500532818754
XE 0.002267615518340493

2. **3 airports with the longest and shortest average taxi time per flight (both in and out), respectively.**

Airports with Longest Taxi Out Time are: 0.0
ACK 30.634848484848487
SOP 26.157728706624606
BQN 25.394020456333596
0.0
Airports with Shortest Taxi Out Time are: 0.0
MKK 4.510416666666667
ADK 6.094594594594595
KSM 6.111675126903553

Airports with Longest Taxi In Time are: 0.0
CKB 183.0
LNY 88.01384083044982
MTH 14.65625
0.0
Airports with Shortest Taxi In Time are: 0.0
BFF 2.0
PVU 2.5
BRW 2.518303052000358

3. **Most common reason for flight cancellations**
CARRIER 289613