SOFE 4360U- Cloud Computing
Project Milestone

Data Processing: DataFlow -Apache Beam

Alex Dafoe 100599423

Ethan Bautista 100657250

Hamza Farhat Ali 100657374

Nick Kvrgic 100613386

Github Link: https://github.com/HamzaFarhat/CloudProject1Group8

Comparison between the data processing services Dataflow, DataProc, DataPrep

Dataflow

This service allows you to manage stream and batch data processing through the running of Apache Beam jobs. It provides a layer of abstraction between the processing logic and the underlying Compute infrastructure to the point where it isn't necessary to provision infrastructure ahead of time in the same manner that you would on other platforms. In this way, Dataflow fundamentally works on a serverless paradigm and fully takes care of all the management when it comes to scaling your clusters and distributing tasks.

Dataproc

This is Google's version of a fully managed service used for Big Data processing applications. While it is commonly used to implement Apache Spark/Hadoop (vs Apache Beam), it is compatible with other similar platforms as well, given that under the hood it relies on Compute Engine instances to run incoming jobs. In addition to this added layer of management abstraction, one of the primary advantages of Dataproc is the fact that it separates data from the processing instance, which is more efficient from a cost perspective as the equivalent server clusters need not be running all of the time. That all being said, it runs "closer to the metal" than Dataflow does, despite being suitable for similar use-cases such as ETL, allowing a more hands-on (server-based) approach to managing the underlying infrastructure.

Dataprep

The most dissimilar of the three, Dataprep's main utility lies in its ability to provide visualizations on Big Datasets and perform cleaning/prep operations upon them. It is largely UI-driven, as it is meant to be used in conjunction with some other data source in most cases. As such its ability to comprehensively ingest streams of data is very limited, however it is much easier to manage than the other two options if this is not a requirement.

Comparisons

- Differences:
  - DataPrep and DataFlow clusters are created using automation while DataProc relies on manual setup for a cluster.
  - Data Proc doesn't have stream processing unlike DataFlow
  - Data Proc is used for data mining and analysis in datasets of known size while Dataflow can handle datasets of unpredictable sizes

- Dataproc has managed and serverless processing. While Dataprep and Dataflow are serverless

- Advantages:

- Dataflow has many use cases such as real time AI, stream analytics and sensor data processing for real time scenarios.
- High security as the dataset is protected due to the access management for users.
- All services allow for scaling up and down processing
- All services allow for encryption the pipeline with customer managed keys

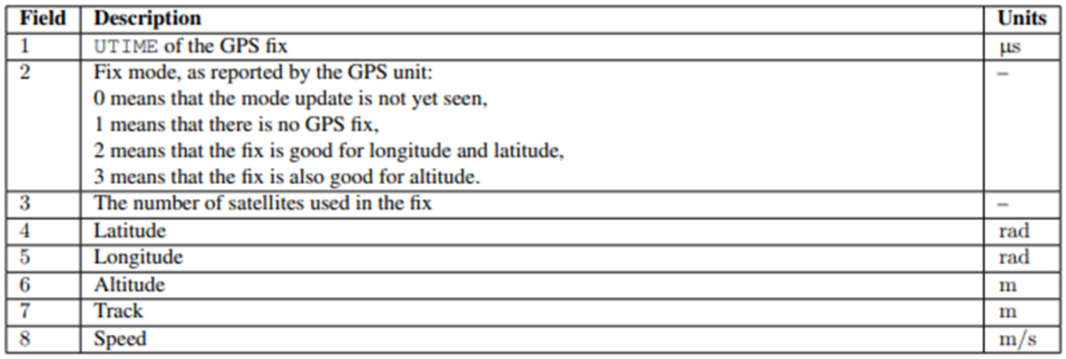
- Disadvantage:

- The support decreases as GCP has limited data centers across the globe which leads to the limitations listed below.
- Data Proc is unable to preprocess for machine learning
- GCP DataFlow has a ceiling on the maximum stream value at one time.
- GCP DataFlow has a ceiling on custom job metrics to 100.
- GCP DataFlow only allows for 15 000 messages every 30 seconds.

Practical Application of both stream and batch processing to a given dataset

A practical application for this data could be autonomous robot monitoring. One could use the data to monitor if the robot is moving. In the case of an autonomous food delivery robot, it should almost always be on the move. If it has stopped for more than a few minutes, likely something is wrong and may require manual intervention. Below I propose two dataflow pipelines that could be used to monitor the health of a fleet of delivery robots.

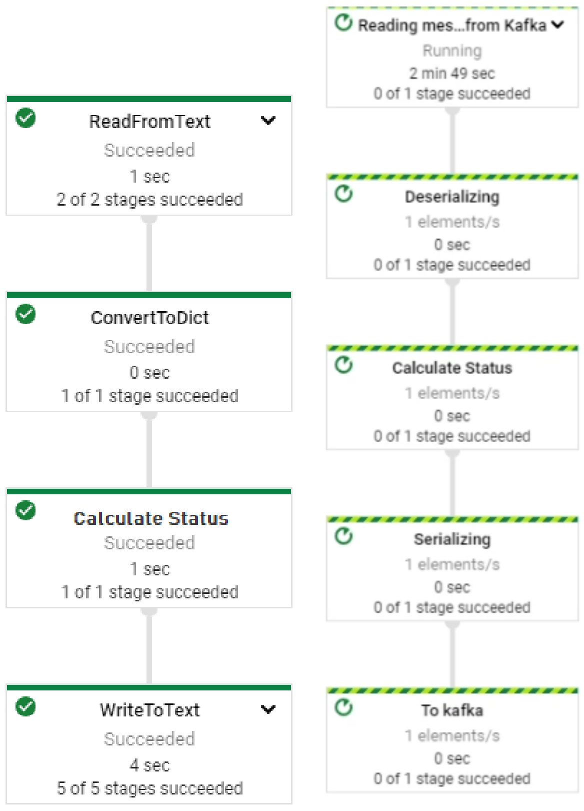
The gps_rtk.csv dataset has the following schema



Field	Description	Units
1	UTIME of the GPS fix	µs
2	Fix mode, as reported by the GPS unit: 0 means that the mode update is not yet seen, 1 means that there is no GPS fix, 2 means that the fix is good for longitude and latitude, 3 means that the fix is also good for altitude.	–
3	The number of satellites used in the fix	–
4	Latitude	rad
5	Longitude	rad
6	Altitude	m
7	Track	m
8	Speed	m/s

Source: http://robots.engin.umich.edu/nclt/nclt.pdf

Proposed Pipelines



1. Robot continuously publishes GPS data to Kafka on where it is
2. In Dataflow a streaming pipeline could be
    a. Consume Kafka message of delivery robot GPS data
    b. Deserialize message
    c. Calculate its trip stats (distance, time) based on the current position and last reported position
    d. Serialize the status message
    e. Publish the message in Kafka under a topic for processed streaming data
3. In Dataflow a batch pipeline could be
    a. Read the whole dataset
    b. Convert data to a dictionary
    c. Calculate overall trip stats (distance, time) over the past 30 minutes
    d. Determine health of robot based on above stats
    e. Write processed data to text file

Why use batch and streaming for this you may ask. Well, a streaming process may have a lot of false positives of non-movement being an issue, e.g., waiting for an obstruction to clear. This would be especially true with very frequent reporting time internals (<1m). Real-time status allows those monitoring the machine to see potential problems early. The batch job on the other hand will have way less false positives of non-movement being an issue as it's monitored over a longer time span. Though with the batch job issues may not be caught right away but will be more definite if there is an issue.