Optional HDFS usage procedure.

Jesús M. Antoñanzas, Alex Carrillo.

Advanced Databases, GCED, UPC - December 17th 2019

Hadoop Distributed File System (HDFS) is "a distributed file system designed to run on commodity hardware ... suitable for applications with large datasets". In the case of our project, generated files such as sensor data can conform a huge dataset in little time. To be specific, for every known flight, a CSV is generated containing as many rows as 5 minute intervals the flight had. So, it is a good practice to distribute this sensor data in a distributed file system so we can later load it and use it in the modelling of the problem. Because of our specific application and having in mind how data is processed in the first pipe '`data_management`', instead of just loading the raw sensor files we are going to process them and save them in a specific format. The steps followed are:

First and foremost, install and configure a single machine local HDFS cluster. We followed this tutorial [1] in our machines.

Then, we analyse the structure of the sensor data and consider a new format in which to load them into HDFS. Formats considered have been CSV, AVRO and Parquet. Having in mind the different reasons why one would use each format, AVRO seems like the best option in our case. That is because Parquet is a columnar storage format and our data has 3 columns, with access being by rows, so no need to optimize there. On the other hand AVRO is row based and optimized to read sequences of rows, which we do. It also takes into account the defined data schema so there is no writing per-value overhead, making serialization both fast and small. Therefore, AVRO is a very good option for our problem, although CSV could also be viable as it behaves well for small file sizes (our case), but does not have the advantage that AVRO's schemas offer.

Finally, having decided the format, the next step is to load sensor data from the local files, process them with Spark (we compute averages of the sensor data per aircraft and day. A raw CSV data lake is out of the scope of this project) and load it into HDFS. In this sense, it is very easy to load the data in AVRO format into HDFS, as the only thing we have to do is define a Spark DataFrame containing the processed data (thus forcing us to define columns - schema) and load it specifying the path (for example, `'hdfs://localhost:9000/user/username/sensordata'`) and that it should be in AVRO format (a single line of code).

Having loaded sensor data, we read it with Spark from the same path it was saved in and transform it into an rdd, from where little formatting is needed because of the preprocessing. Then, we continue with the data management pipe.

References

[1] Sámano, E., 2018. *Hadoop on OS X (High Sierra) Installation of Apache Hadoop Single Node Cluster using Homebrew and "regular" way trough the binaries.* 05.