Lecture 7: Use Spark and EMR workflow

Contents

7.1. Announcements

- Here is video for setting up foxyproxy

7.2. Objectives

- Showcasing Spark Workflow. EMR way of executing the entire pipeline.
- Use of Spark and various languages to interact with Spark.
- Run Spark using EMR steps. (Run entire workflow in EMR)

7.3. Refresher

7.3.1. Theory

- Can you list the advantages of using Spark?
- Are you able to name Spark core components and their functions?
- Do you know why Spark is known as a `Unified engine` for large-scale data analytics?

7.3.2. Practical (from last class demo)

- Are you comfortable setting up an EMR cluster?
- Do you know to ssh into the master node?

- Are you comfortable setting up an EMR cluster? E.g., Can you set up a 5 node EMR cluster with
  - 1 master node having 64 GB RAM/16 cores
  - 2 core node with 16GB RAM/ 4 cores
  - 2 task node with 32GB RAM/ 8 cores
  - All task nodes being spot instances and other on-demand instances
- Do you know how to use the EMR command to set up your cluster from `AWS CLI import`?
- Can you spin up an EMR instance using CLI?
- Are you aware of 2 security groups managed by AWS while setting up an EMR instance?
- Can you edit security groups if needed? For eg
  - We added the ssh rule in "default master security group".
- How to terminate and clone an EMR instance?
- Can you resize your cluster by increasing the number of task nodes or core nodes?

- Do you know how you can use your EMR cluster to do Spark programming?
- Access EMR notebooks and using pyspark kernel?
- Can you set up EMR notebooks?
  - Do you know how to specify the S3 bucket for saving your notebooks automatically?
  - Open the pyspark kernel notebook and run a simple spark program?
- Do you know how you can use your EMR cluster to do Spark programming?

Theory

- Can you list the advantages of using Spark?
  - It is easier to use than Hadoop MapReduce.
  - It is faster than Hadoop MapReduce.
  - It is data scientist friendly and supports various languages like R and Python. NOTE there are a lot more that you can find from our lectures 5 and 6.
- Are you able to name Spark core components and their functions?
  - Spark Core: It is the heart of Spark. It is responsible for scheduling, distributing, and monitoring the execution of tasks across the cluster. It also provides the basic APIs for working with RDDs, and dataframe APIs
  - Spark SQL: It is a Spark module for structured data processing. It provides a programming abstraction called DataFrames and can act as a distributed SQL query engine.
  - Spark Streaming: It is a Spark module for processing streaming data. It provides a high-level abstraction called DStreams that represents a continuous data stream.
  - Spark MLlib: It is a Spark module for machine learning. It provides a uniform high-level APIs that help users create and tune practical machine-learning pipelines.
  - Spark GraphX: It is a Spark module for graph processing. It provides a high-level API for expressing graph-parallel computations.
- Do you know why Spark is known as a `Unified engine` for large-scale data analytics?
  - Spark is a unified engine because it provides a single API for working with different types of data. For example, you can use the same API to work with structured data in Spark SQL, unstructured data in Spark Streaming, and graph data in Spark GraphX. While in the case of Hadoop, you need to use different APIs for different types of data. For example, you need to use MapReduce for batch processing, Hive for structured data processing, Pig for unstructured data processing, and Mahout for machine learning. Or in other simple words, Spark is just a single

installation, and it comes with all the pieces (like all the components in the previous question) that you need to do large-scale data processing. In contrast, in the case of Hadoop, you need to install all the components separately.

Practical (from last class demo)

- Are you comfortable setting up an EMR cluster?
    - You might need to do this at your work to run your workload on EMR.
- Do you know to ssh into the master node?
    - You can find how to do this from detail in the EMR cluster you set up.
- Are you comfortable setting up an EMR cluster? E.g., Can you set up a 5 node EMR cluster with
    - remember the various options we selected and see if you can set up one based on your situation.
- Do you know how to use the EMR command to set up your cluster from `AWS CLI import`?
    - You can get the EMR command from the cluster that setup (by just clicking the button we did in last class). You can use that command to setup your cluster.
- Can you spin up an EMR instance using CLI?
    - You can use the EMR command you got from the cluster you set up to spin up an EMR instance using CLI.
- Are you aware of 2 security groups managed by AWS while setting up an EMR instance?
    - You can find the security groups from the cluster that you have setup. One security group is for the master node, and the other is for slave nodes (core and task node).
- Can you edit security groups if needed?
    - You can click on the security groups from the cluster you set up. From there, you can edit the security groups to add inbound rules to allow ssh access to the master node.
- How to terminate and clone an EMR instance?
    - You saw the options in the cluster that you set up. You can terminate and clone the cluster from there.
- Can you resize your cluster by increasing the number of task nodes or core nodes?
    - You can resize your cluster by increasing the number of task nodes or core nodes from the cluster you set up.
- Do you know how you can use your EMR cluster to do Spark programming?
    - You can use your EMR cluster to do Spark programming by using the EMR notebooks that EMR provides.
- Access EMR notebooks and use Pyspark kernel?
    - You saw it from the last lecture. You can also find details in the second video in the video zone.
- Can you set up EMR notebooks?
    - Watch the second video in the video zone.
- Do you know how you can use your EMR cluster to do Spark programming?
    - You can use your EMR cluster to do Spark programming by using the EMR notebooks that EMR provides. You will see how to use Spark in today's class and the supplement materials I will provide you today.

7.4. Spark story

In 2003 Google was dealing with lots of big data, and they needed something to manage large flow of data. All the existing in-house systems, like relational databases and other cluster computing, couldn't help with their needs, so they started developing new technologies. This requirement led them to create a new file system called Google File System (GFS), a new processing framework called MapReduce (MR), and a new database called bigtable.

These papers from Google on Google File System (GFS) and MapReduce (MR) serve as the cornerstone for Hadoop (HDFS & MapReduce). This initiative was led by a couple of engineers in Yahoo, led by Doug Cutting.

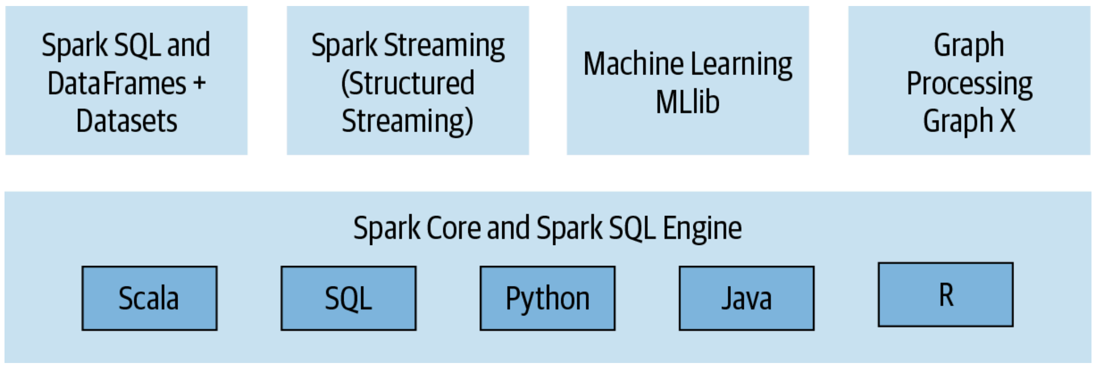
Let's list down some disadvantages of Hadoop.

- Verbose
- Single language
- Many other silos (kind of) systems to handle other workloads like Mahout, storm, giraph etc…
- The intermediate computed result is written to the local disk

To overcome these shortcomings and make Hadoop and MR simpler and faster, a couple of researchers at UC Berkely and Matei Zaharia developed spark in 2009.

Over the years, Spark has become the de-facto big data universe. Here are some advantages

- Highly fault-tolerant
- Embarrassingly parallel
- Support in-memory storage for intermediate results
- Easy and composable APIs in
- Multiple languages support other workloads in a unified manner
- Unified processing engine. The below diagram describes the unification…


Spark SQL and DataFrames + Datasets
Spark Streaming (Structured Streaming)
Machine Learning MLlib
Graph Processing Graph X
Spark Core and Spark SQL Engine
Scala
SQL
Python
Java
R

Spark components

About the components

- Spark SQL (more on this next week) You can read data from a file or any relational database into Spark and register it as a temporary or permanent table. Once it is a table, you can use SQL-like queries.
- Spark MLib A library with machine learning modules implemented like scikit-learn. You might not find all functionalities you can see in scikit-learn, but they add new functionalities with every release.
- Structured streaming To process streaming data
- GraphX Library for manipulating graphs

7.5. Spark architecture

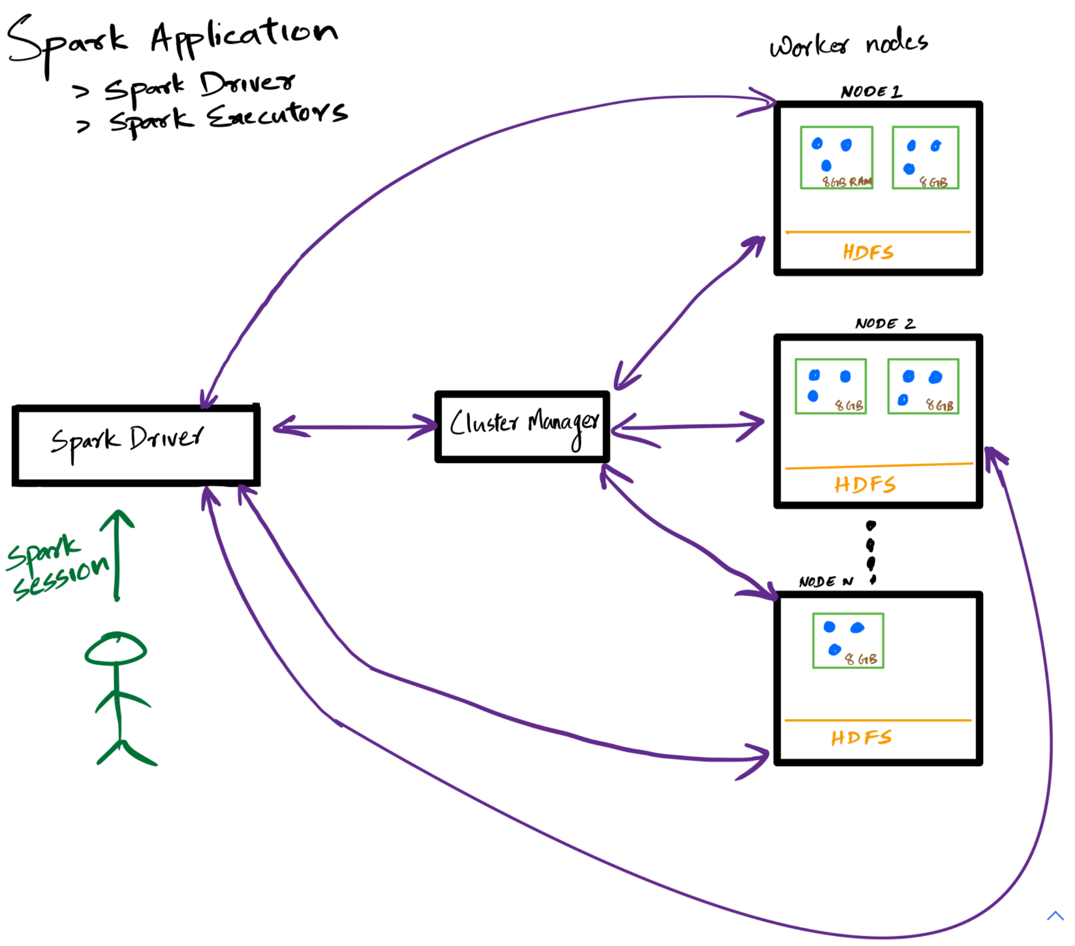
Any `spark application` consists of a `spark driver` and `spark executors`. `Spark executors` is where the spark tasks are executed. `Spark driver` program is responsible for orchestrating all parallel operations on a spark cluster. From the user end, the first step is to initialize a `spark session` that gives information to the `spark driver` about your `spark application` and your required resources. In short, the `Spark session` is the entry point into all functionality in Spark.

Spark driver responsibilities.

- Communicate to cluster manager - e.g., YARN in our case
- Requests for resources - number of cores, RAM, number of executors, etc…

Here are a few parameters that you can use to tune your spark application.

- –driver-memory 3g
- –executor-memory 3g
- –executor-cores 4
- –num-executors 2



16

> **Note**
>
> We will explore the above diagram in more detail in the next class. I will also demo how this works behind the scene by exploring a spark job. For now, let's get what's needed for the milestone.

- It processes your query computations as a directed acyclic graph (DAG) and distributes their execution as `tasks` across the Spark `executors`.

7.6. How to use Spark - user view

NOTE: Demo at the end

7.6.1. Spark shell (Command-line)

7.6.2. EMR notebooks (TBD, yet to hear from AWS about the permission issues)

Heard back from AWS Academy Program Manager, they are still investigating the issue, meanwhile here is the workaround you can use.)

I will demo this in class, and if you want to get to it earlier, please watch this short video. Passcode: `ZgZ=9M=K` - WON'T WORK THIS YEAR (2024)

See also

- Jupyter-based Notebooks
  - You can access your spark cluster from jupyter on your laptop, in your EC2 instance, ubc.syzygy.ca, or anywhere else. Check out this Kernel.
  - EMR managed JupyterHub Notebooks
  - EMR notebooks - we will be using this.
  - EMR Studio
  - SageMaker notebooks
- Non-jupyter based notebooks:
  - Apache Zeppelin
  - Hue
  - Databricks notebook
- Other vendor-based products
  - Dataiku
  - DataRobot
  - Alteryx

7.6.3. EMR way using Steps (demo)

Note

This is very important. Let's table it for now. I put it here just so you know that there are other ways to use Spark. But you don't need to use it for milestones. So I am keeping it for next week to manage the contents.

Note

The Spark driver instantiates a SparkSession for you in an interactive Spark shell and Notebooks, so you DON'T WANT to invoke one explicitly. But when you write spark scripts for spark-submit and EMR way using Steps, you need to initiate a SparkSession like the following;

spark = SparkSession
.builder
.appName("gittu")
.getOrCreate()

7.6.4. Spark-submit using scripts

Note

This is very important. Let's table it for now. I put it here so you know there are other ways to use Spark. But you don't need to use it for milestones. So I am keeping it for next week to manage the contents.

```
spark-submit --deploy-mode cluster --driver-memory 3g --executor-memory 3g --executor-cores 4 s3://scripts
```

7.7. Languages and packages for spark interaction

We saw in last class we can use many languages to interact with Spark. Here are some more details and various packages that are out there in various programming languages to interact with spark.

7.7.1. R

- sparklyr official docs. Here is a Great Book to refer
- Rspark
- h20

Here are the instructions on how to setup an EMR cluster with sparklyr, spark and Rstudio. Here is the AWS CLI script to setup the cluster with sparklyr.

```
aws emr create-cluster --applications Name=Hadoop Name=Spark Name=Livy Name=JupyterHub Name=JupyterEnterp
```

Once you have the cluster up and running, you can use the R studio with the cluster to run your sparklyr codes. For example, here is how the sample problem we executed in our first week will look in sparklyR. As you may notice, the code is the same as the dplyr, but with the below code, you are using sparklyr to connect to the spark cluster and execute the code in a distributed fashion with all the great things we learned about spark and Hadoop.

```
# load the sparklyr, dplyr, ggplot2, and DBI packages
library(sparklyr)
library(dplyr)
# $spark.dynamicAllocation.enabled
config <- spark_config()
config["spark.executor.memory"] <- "3G"
config["spark.driver.memory"] <- "2G"
config["spark.executor.cores"] <- "4"
config["spark.driver.cores"] <- "2"

# create the Spark context
sc <- spark_connect(master = "yarn", version = "3.1.2", config = config)
ds <- spark_read_parquet(sc, path = "s3://testmds/combined_data_partition.big10.parquet",memory = FALSE)
ds %>%
  filter(year==2002,Origin=="ORD",Dest=="PHL",DepDelay > 10) %>%
  group_by(UniqueCarrier) %>% summarise(average_delay = mean(DepDelay))
```

See also

Considering the short time, here is my initiative to provide you with some resources to quickly get to spark programming in R and Python based on what you learned in previous courses. Link to 571 mix. Sorry not complete, as I didn't get much time this year, but hopefully will be able to provide access to you in the upcoming months.

7.7.2. SQL (Spark SQL)

See also

Check this SparkSQL notebook to see how your 513 lab1 codes are running in sparkSQL. You might notice it is the same SQL query that you learned. But now you are running it in a distributed fashion with the infrastructure you set up in EMR.

7.7.3. Python

- Pyspark
- Koalas
- Sk-dist
- elephas
- Tensorflow on Spark

See also

For more information: Check Tutorial 2 Check Tutorial 3 Check Mini tutorial done during lab 3

7.7.4. Scala

7.7.5. Java

7.7.6. From other tools using drivers

- Tableau
- Qlikview
- PowerBI

7.8. How to use documentation

This is the starting point. Checking the documentation with the correct version is important.

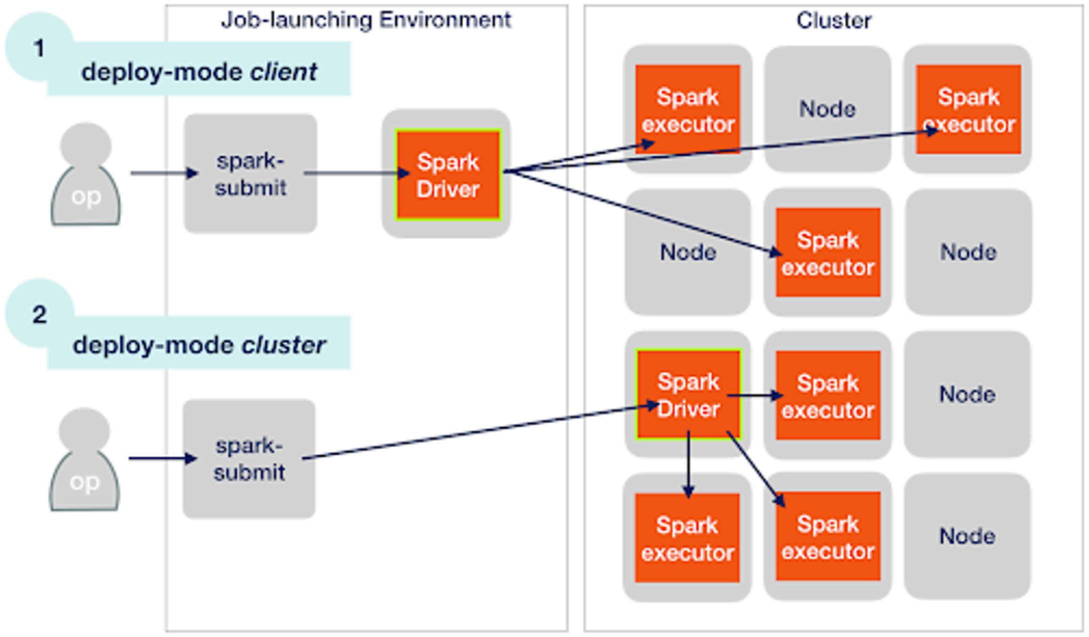
Exercise

Try to work out these exercises when you have more time…

- Can you write your linear regression ML code using MLlib?
- Can you take some other SQL codes and write them using sparkSQL?
- Take some code from your 523 and write it using sparklyr?

7.9. Various spark run modes

- Standalone
- Client
- Cluster - This is how enterprise Spark jobs are deployed.



7.10. Connecting dots (Where are we at?)

7.10.1. Task performed as part of development.

- Milestone 1:
  - You saved the parquet file into your S3.
- Milestone 2:
  - You set up an EC2 instance capable of handling all the wrangling.
  - Installed all necessary packages that are needed.
  - Did wrangling using pandas and other libraries that you are already comfortable with.
  - Saved ML deployment ready file in S3 (joblib file)
- Milestone 3:
  - Setup your EMR cluster
  - Install all necessary packages that are needed.
  - You read that file from S3.
  - Did your modeling stuff in scikit-learn
  - Used spark MLlib to get some hyperparameters.
  - Saved your model to S3

First, let's summarize the main tasks;

- Setting up your account and permissions to access the S3 bucket for data.
- Set up your compute resource - running pandas and scikit-learn, Spark for getting the hyperparameters.
- Install necessary packages.
- How to run Spark. Check first half of today's lecture for details on ways to run Spark.
  - Notebooks (Jupyter, Zeppelin, Hue etc..). Here is the notebook.
  - Spark-submit using scripts. Convert the notebook to scripts and add the code to initiate a SparkSession.
  - Spark shell (Command-line). If you like CLI. But jupyter is friendly.

> **Important**
>
> ***Let's complete the workflow you did as part of your milestones in the AWS recommended style.***
>
> ***IDEAL APPROACH FOR THE ABOVE PIPELINE:*** Let's work out the above tasks using EMR way using Steps and bootstrap. Here is the script that you can use for steps, and here is the script for bootstrap.

7.11. What we learned today?

- Difference between various spark deployment modes and when to use what.
- Spark architecture and terminologies.
- Spark application concepts.
- How to include EMR as part of your workflow.
- List some external packages that use Spark