Apache Spark Tutorial

Apache Spark is a data analytics engine. These series of Spark Tutorials deal with Apache Spark Basics and Libraries : Spark MLlib, GraphX, Streaming, SQL with detailed explaination and examples.

Apache Spark Tutorial

Following are an overview of the concepts and examples that we shall go through in these Apache Spark Tutorials.

Spark Core

Spark Core is the base framework of Apache Spark. It contains distributed task Dispatcher, Job Scheduler and Basic I/O functionalities handler. It exposes these components and their functionalities through APIs available in programming languages Java, Python, Scala and R.

To get started with Apache Spark Core concepts and setup :

- Install Spark on Mac OS – Tutorial to install Apache Spark on computer with Mac OS.
- Setup Java Project with Apache Spark – Apache Spark Tutorial to setup a Java Project in Eclipse with Apache Spark Libraries and get started.
- **Spark Shell** is an interactive shell through which we can access Spark's API. Spark provides the shell in two programming languages : Scala and Python.
  - Scala Spark Shell – Tutorial to understand the usage of **Scala** Spark Shell with Word Count Example.
  - Python Spark Shell – Tutorial to understand the usage of **Python** Spark Shell with Word Count Example.
- Setup Apache Spark to run in Standalone cluster mode
- Example Spark Application using Python to get started with programming Spark Applications.
- Configure Apache Spark Ecosystem
  - Configure Spark Application – Apache Spark Tutorial to learn how to configure a Spark Application like number of Spark Driver Cores, Spark Master, Deployment Mode etc.
  - Configuring Spark Environment
  - Configure Logger

Spark RDD

Spark is built on RDD (Resilient Distributed Database). RDD is the framework that provides Spark the ability to do parallel data processing on a cluster. We shall go through following RDD Transformations and Actions.

- About Spark RDD

- Create Spark RDD
- Print RDD Elements
- Read text file to Spark RDD
- Spark – Read multiple text files to a single RDD
- Spark – RDD with custom class objects
- Spark RDD Map
- Spark RDD Reduce
- Spark RDD FlatMap
- Spark RDD Filter
- Spark RDD Distinct
- Spark RDD with Custom Class Objects
- Spark RDD foreach to iterate over each element of Distributed Dataset.
- Read JSON File to RDD

Spark DataSet

- Read JSON File to Spark DataSet
- Write Spark DataSet to JSON File
- Add new column to Spark DataSet
- Concatenate Spark Datasets

Spark MLlib – Apache Spark Tutorial

A detailed explanation with an example for each of the available machine learning algorithms is provided below :

- Classification using Logistic Regression – Apache Spark Tutorial to understand the usage of Logistic Regression in Spark MLlib.
- Classification using Naive Bayes – Apache Spark Tutorial to understand the usage of Naive Bayes Classifier in Spark MLlib.
- Generalized Regression
- Survival Regression
- Decision Trees – Apache Spark Tutorial to understand the usage of Decision Trees Algorithm in Spark MLlib.
- Random Forests – Apache Spark Tutorial to understand the usage of Random Forest algorithm in Spark MLlib.
- Gradient Boosted Trees
- Recommendation using Alternating Least Squares (ALS)
- Clustering using KMeans – Apache Spark Tutorial to understand the usage of KMean Algorithm in Spark MLlib for Clustering.
- Clustering using Gaussian Mixtures
- Topic Modelling in Spark using Latent Dirichlet Conditions
- Frequent Itemsets
- Association Rules
- Sequential Pattern Mining

How Spark came into Big Data Ecosystem

When Apache Software Foundation has started Hadoop, it has two important ideas for implementation :

MapReduce and Scale-out Storage system. With institutional data, sensor data(IOT), social networking data etc., growing exponentially, there was a need to store vast amount of data with very less expenses. The answer was HDFS (Hadoop Distributed File System). In order to process and analyze these huge amounts of information from HDFS very efficiently, Apache Hadoop saw the need for a new engine called MapReduce. And soon MapReduce has become the only way of data processing and analysis with Hadoop Ecosystem. MapReduce being the only option, soon led to the evolution of new engines to process and analyse such huge information stores. And Apache Spark has become one of the interesting engine of those evolved.

Spark was originally designed and developed by the developers at Berkeley AMPLab. To take the benefit of wide open community at Apache and take Spark to all of those interested in data analytics, the developers have donated the codebase to Apache Software Foundation and Apache Spark is born. Hence, Apache Spark is an open source project from Apache Software Foundation.

Hadoop vs Spark

Following are some of the differences between Hadoop and Spark :

Data Processing

Hadoop is only capable of batch processing.

Apache Spark's flexible memory framework enables it to work with both batches and real time streaming data. This makes it suitable for big data analytics and real-time processing. Hence Apache Spark made, continuous processing of streaming data, rescoring of model and delivering the results in real time possible in the big data ecosystem.

Job Handling

In Hadoop, one has to break their whole job into smaller jobs and chain them together to go along with MapReduce. Also APIs are complex to understand. This makes building long processing MapReduce jobs difficult.

In Spark, APIs are well designed by the developers for the developers and did a great job in keeping them simple. Spark lets you describe the entire job and handles the job very efficiently to execute in parallel form.

Support to existing databases

Hadoop can process only the data present in a distributed file system (HDFS).

Spark in addition to the distributed file systems, also provides support to using much popular databases like MySQL, PostgreSQL, etc., with the help of its SQL library.

Features of Apache Spark

Apache Spark engine is fast for large-scale data processing and has the following notable features :

High Speed

Spark run programs faster than Hadoop MapReduce : 100 times faster with in-memory and 10 times faster with disk memory

Ease of Use

Spark provides more than 80 high level operations to build parallel apps easily.

Ease of Programming

Spark programs could be developed using various programming languages like Java, Scala, Python, R.

Stack of Libraries

Spark combines SQL, Streaming, Graph computation andMLlib (Machine Learning) together to bring in generality for applications.

Support to data sources

Spark can access data in HDFS, HBase, Cassandra, Tachyon, Hive and any Hadoop data source.

Running Environments

Spark can run on : Standalone machine in cluster mode, Hadoop, Apache Mesos or in the cloud.

Apache Spark's Runtime Architecture

Apache Spark works on master-slave architecture. When a client submits spark application code to the Spark Driver, Spark Driver implicitly converts the transformations and actions to (DAG)Directed Acyclic Graph and submits it to a DAG Scheduler (During this conversion to DAG, it also performs optimization such as pipe-line transformations). Now, DAG scheduler converts logical graph (DAG) into physical action plan containing stages of tasks. These tasks are bundled to be sent to cluster.

Cluster Manager keeps track of the available resources in the cluster. Once Driver has created and bundled the tasks, it negotiates with the Cluster Manager for Worker nodes. After the negotiation (which results in allocation of resources for executing spark application), Cluster Manager launches Executors on Worker nodes and let driver know about the Executors on Workers. Based on the placement of Executors and their reachability to data, Driver distributes them the tasks. Once the Executors are ready to start with the task, they register themselves with the Driver, so that Driver can have whole view of Executors and monitor them during task execution. Some of the tasks are dependent on the output data from other tasks. In such scenarios, Driver is responsible for scheduling these future tasks in appropriate locations based on location where data might get cached or persisted.

While Spark Application is running in the driver, it exposes information through Web UI to the user. Once SparkContext is stopped, the Executors get terminated.

Usage of Apache Spark

Apache Spark is being used in solving some of the interesting real-time production problems and following are few of the scenarios :

1. Financial Services
   - Identifying fraudulent transactions and adapting to the new fraud techniques and updating the model in real time is required.
   - In identifying the customer's buying pattern of stocks and making the predictions for stock sales etc.
2. Online Retail Market
   - Online Retail giants like Alibaba, Amazon, eBay use Spark for customer analytics like suggesting a product based on the buying product browsing history, transaction logging etc.
3. Expense Analytics
   - Concur is using spark for personalization and travel and expenses analytics.

A huge number of companies and organisations are using Apache Spark. The whole list is available here[http://spark.apache.org/powered-by.html].

Summary

This article provides a good introduction about what Apache Spark is; features of Apache Spark; its differences with Apache Spark; which modules are present in Apache Spark; different operations available in the modules and finally some of the use cases in real-time.

- Spark RDD - foreach
- Spark RDD - Create RDD
- Spark Parallelize
- Spark RDD - Read Text File to RDD
- Spark RDD - Read Multiple Text Files to Single RDD
- Spark RDD - Read JSON File to RDD
- Spark RDD - Containing Custom Class Objects
- Spark RDD - Map
- Spark RDD - FlatMap
- Spark RDD - Filter
- Spark RDD - Distinct
- Spark RDD - Reduce

Spark Dataseet

- Spark - Read JSON file to Dataset
- Spark - Write Dataset to JSON file
- Spark - Add new Column to Dataset
- Spark - Concatenate Datasets

Spark MLlib (Machine Learning Library)

- Spark MLlib Tutorial
- KMeans Clustering & Classification
- Decision Tree Classification
- Random Forest Classification
- Naive Bayes Classification
- Logistic Regression Classification
- Topic Modelling

Spark SQL

- Spark SQL Tutorial
- Spark SQL - Load JSON file and execute SQL Query

Spark Others

- Spark Interview Questions