Big Data Analytics and Reasoning - Practice 06

Giuseppe Mazzotta

Apache Spark

Apache Spark is a cluster computing platform designed to be fast and general purpose

Main feature: run computation in memory ram

Spark is designed to be highly accessible, offering simple APIs in Python, Java, Scala, SQL, and more

Spark can run in Hadoop clusters and access any Hadoop data source

1. Spark Key Factors

➔ **Speed**

Works on distributed environment - Efficient query optimizer - Reduce at minimum disk I/O

➔ **Ease of Use**

It provides a simple programming model based on RDD and a set of operations (transformations and actions)

➔ **Modularity**

Provides different API for different workloads (MLlib, SQL) that can live together in a spark application

➔ **Extensibility**

It focuses on fast parallel computation engine and let the possibility to extended it (e.g. reading from source not directly supported)

A Unified Stack

It provides different components as libraries used for different workloads. Spark libraries are separated components from the spark core

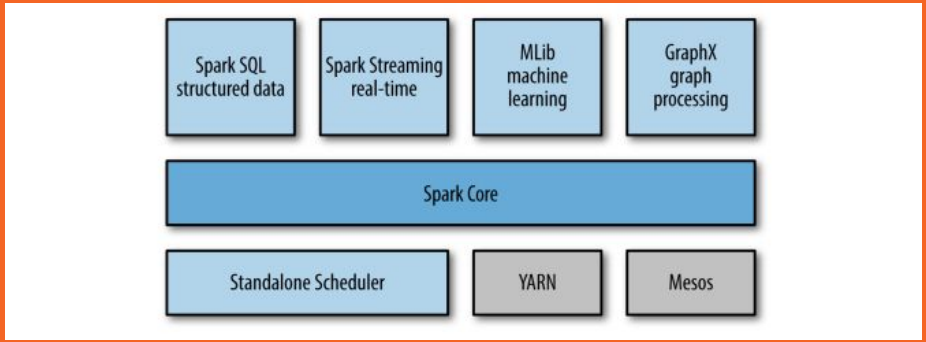
Spark SQL allows you to read structured data from different sources, run sql query or create temporary/permanent view/table

Spark MLlib provides many different machine learning algorithms that allow you to build models on top of spark datasets

Spark Structured Streaming is built on top of Spark SQL and allows you to deal with real-time data processing

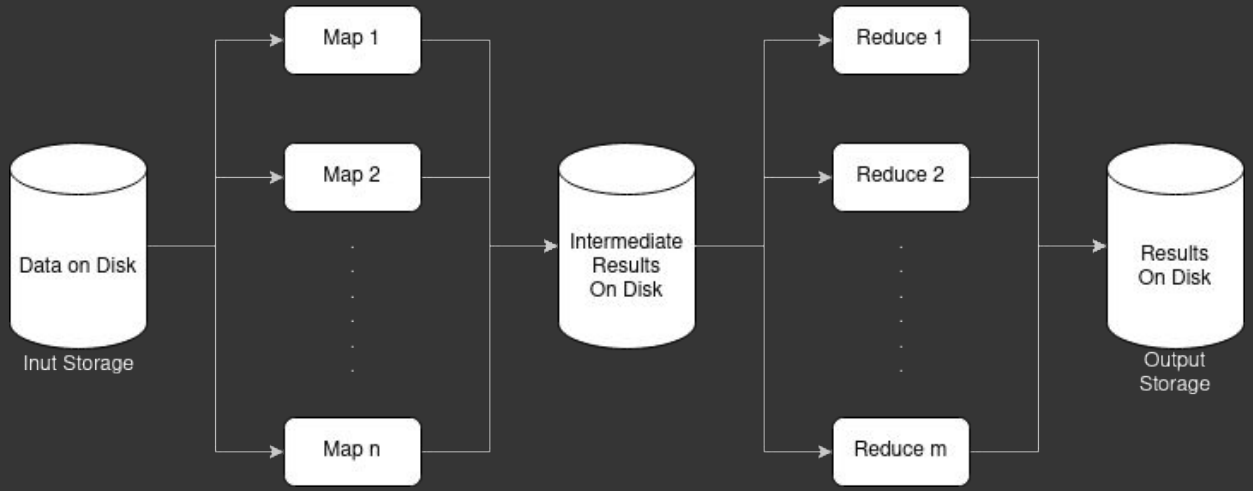
GraphX is a library for manipulating graphs and perform parallel computations on them

Spark core gestisce tutte le funzioanlità



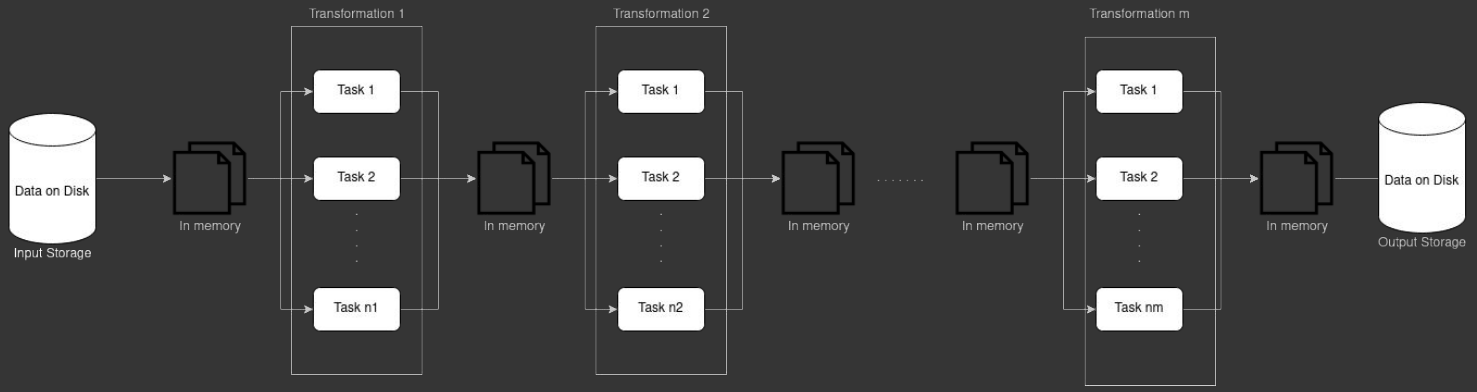
Spark vs MapReduce

Typical MapReduce Workflow



Spark vs MapReduce

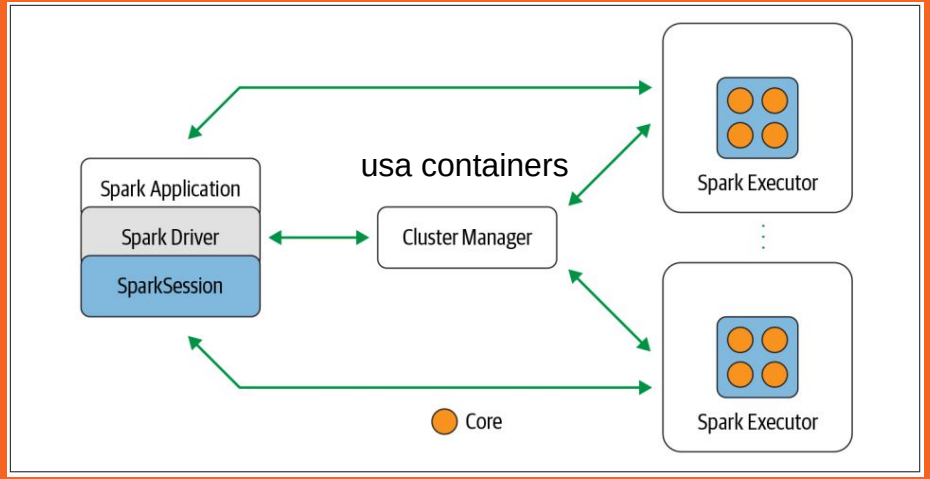
Typical Spark Workflow



Spark Components

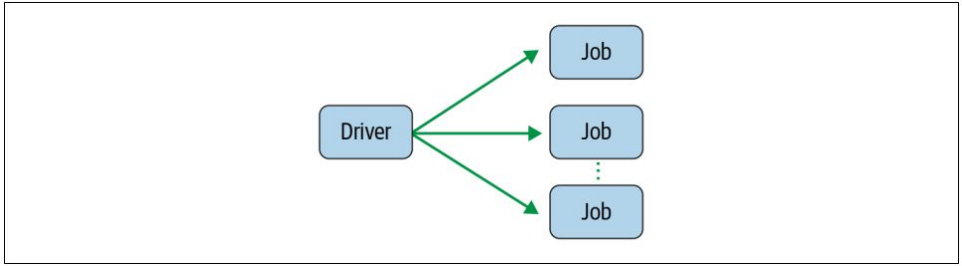
At higher level a Spark application consists of a driver program that orchestrates parallel operations on cluster machines

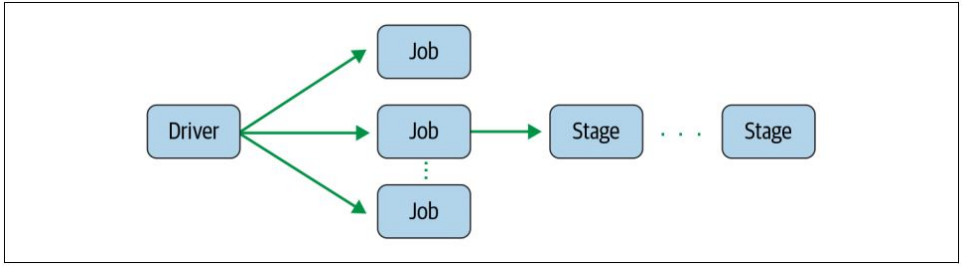
- **Spark Driver**
  Is responsible for negotiating resources; transform and scheduling spark operation among the Executors
- **Spark Session**
  Allows you defining datasets, reading data from differents sources, run sql queries
- **Cluster Manager**
  Is responsible for managing and allocating resources
- **Spark Executor**
  Is responsible for executing task on workers machines

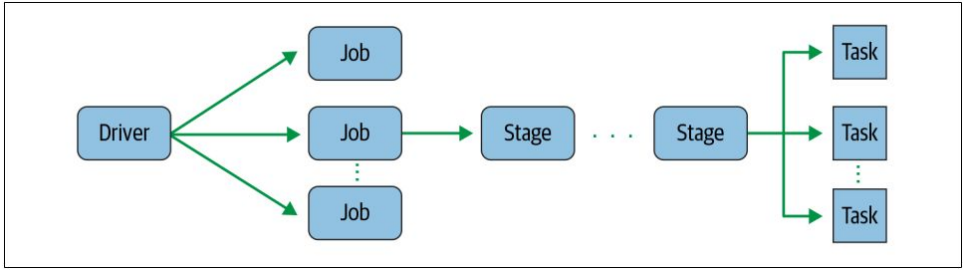


Spark App Execution

- **Spark Application**
  User program defined using Spark API that include both spark driver program and sparksession
- **Spark Job**
  Parallel computation that get spawned in response to a spark action
- **Spark Stage**
  Set of tasks that belong to a given job
- **Spark Task**
  Single unit of work assigned to a Spark Executor







2. Spark Data Structure

Sparks provide basic structured data abstraction that are:

➔ **RDD (Resilient Distributed Dataset)**

Is a collection of object distributed across main memory of cluster machines

➔ **Dataframe**

Is a collection of structured data similar to relational table distributed across main memory of cluster machines

➔ **Dataset**

Is a strongly typed collection of object that is merged with dataframe under the StructuredAPI

NOTE all this abstraction are immutable

3. Spark Operations

Sparks provide two different operations:

➔ **Transformation**

Transform a spark Dataset into a new one without altering the original data
Narrow dependencies - each output partitions can be computed from one input partition (filter, map …)
Wide dependencies - transformation in which data from other partitions is read (group by, order by, …)

➔ **Action**

Compute a value based on Dataset data and returns it to the driver program

NOTE
Transformations in spark are lazy. They will be computed as soon as some actions trigger them. *cache()* method enforce dataset materialization

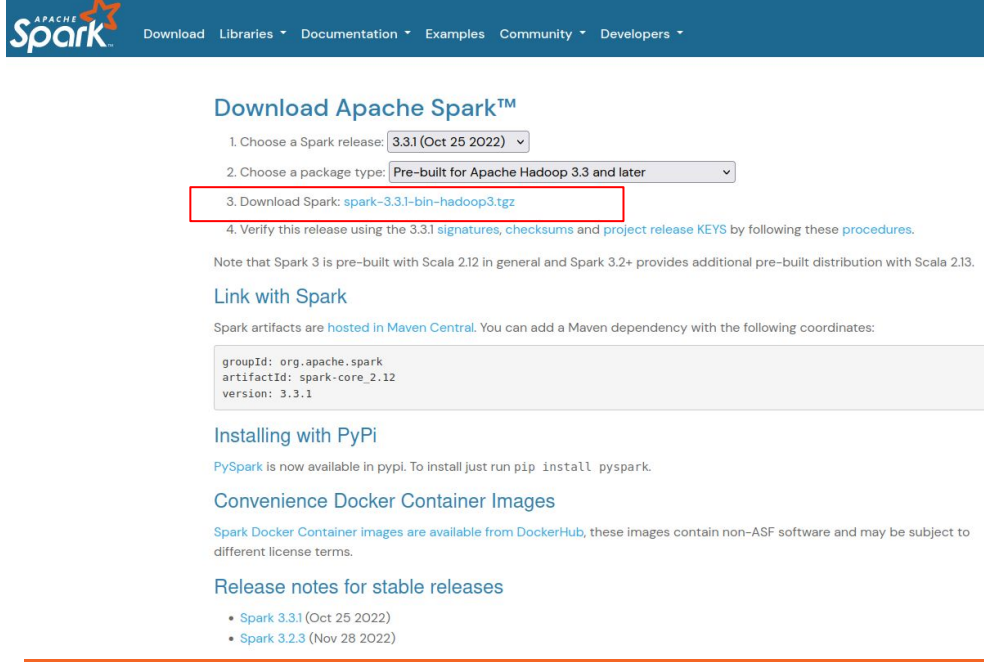
Spark configuration

Download a binary distribution from the spark website

Unfold the archive and update *.bashrc* file:

Export the SPARK_HOME and HADOOP_CONF_DIR environment variable

Add to the PATH variable both *bin* and *sbin* directory of the spark distribution



Tip

Download and configure it directly into the master

Spark Configuration

GNU nano 4.8 spark-defaults.conf

```
#
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#

# Default system properties included when running spark-submit.
# This is useful for setting default environmental settings.

# Example:
spark.master                     yarn
spark.eventLog.enabled           true
spark.eventLog.dir               hdfs://master:9000/spark_logs
spark.eventLog.provider          org.apache.spark.deploy.history.FsHistoryProvider
spark.history.fs.logDirectory    hdfs://master:9000/spark_fs_logs
spark.history.fs.update.interval 10s
spark.history.fs.port            18080
spark.driver.memory              1g
```

GNU nano 4.8 workers.template

```
#
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#

# A Spark Worker will be started on each of the machines listed below.
slave1
slave2
```

Tip

Before launching spark application check/create spark_logs folder into hdfs

4. Running Spark

➔ **Pyspark shell**

Python shell that can be used to fast analysis and interactive queries

> *pyspark*

➔ **Spark-submit**

Allows you to submit a spark application on the cluster

> *spark-submit --class package.MainClass app.jar*

➔ **Example**

echo “Hello Spark World” >> hello.txt
hdfs dfs -put hello.txt .
pyspark
>>> spark.read.text(“hello.txt”).show()

Let practice with spark ...