Overview of Apache Spark™

A Distributed Processing Engine

Agenda

We will cover the following topics:

- Spark and Big Data basics.
- Setting up Spark in various ways.
- Python Review
- Python and Spark 2.0 DataFrames
- PySpark Project Exercise

Resources

http://arif.works/shd

Spark

- This lecture will be an abstract overview, we will discuss:
  - Spark
  - Spark vs MapReduce
  - Spark RDDs
  - Spark DataFrames

Spark

- Spark is one of the latest technologies being used to quickly and easily handle Big Data
- It is an open source project on Apache
- It was first released in February 2013 and has exploded in popularity due to it's ease of use and speed
- It was created at the AMPLab at UC Berkeley

Spark

- You can think of Spark as a flexible alternative to MapReduce

- Spark can use data stored in a variety of formats
  - Cassandra
  - AWS S3
  - HDFS
  - And more

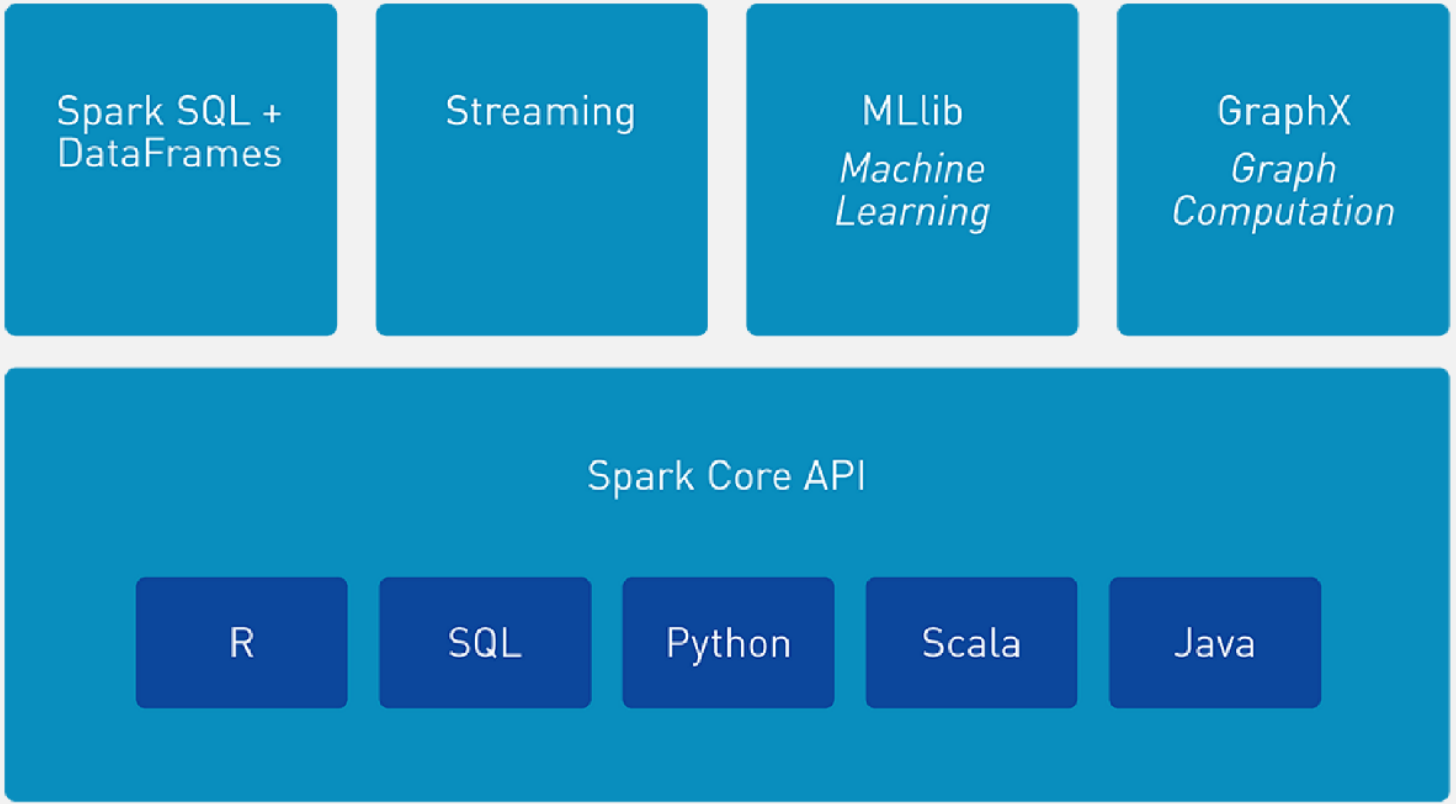
Spark Vs MapReduce

- MapReduce requires files to be stored in HDFS, Spark does not!
- Spark also can perform operations up to 100x faster than MapReduce
- So how does it achieve this speed?

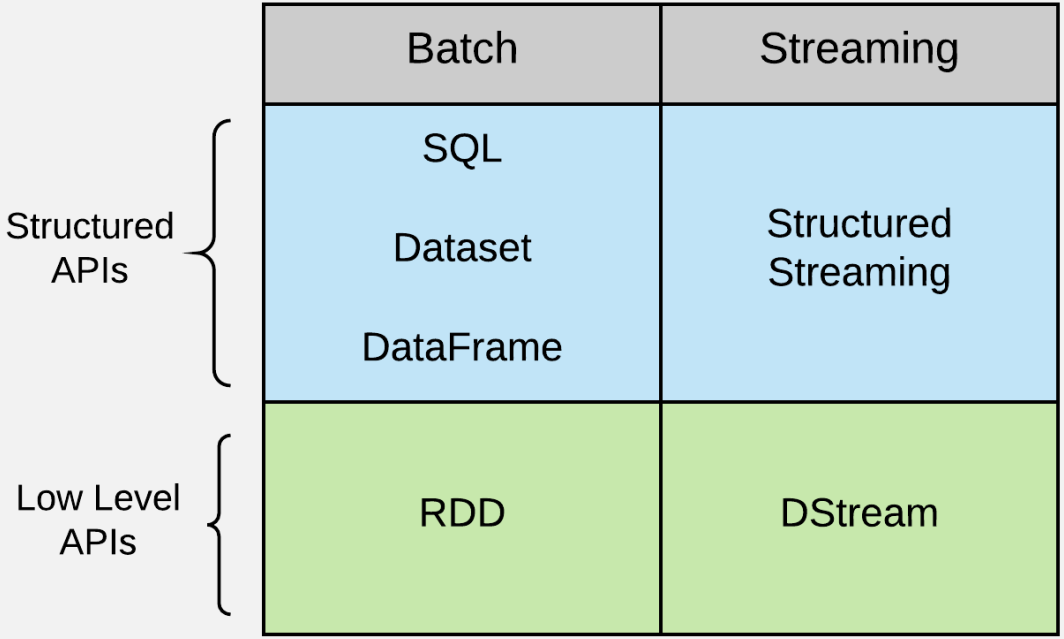
Spark Vs MapReduce

- MapReduce writes most data to disk after each map and reduce operation
- Spark keeps most of the data in memory after each transformation
- Spark can spill over to disk if the memory is filled

Spark API


Spark SQL + DataFrames
Streaming
MLlib
Machine Learning
GraphX
Graph Computation
Spark Core API
R
SQL
Python
Scala
Java

Spark API

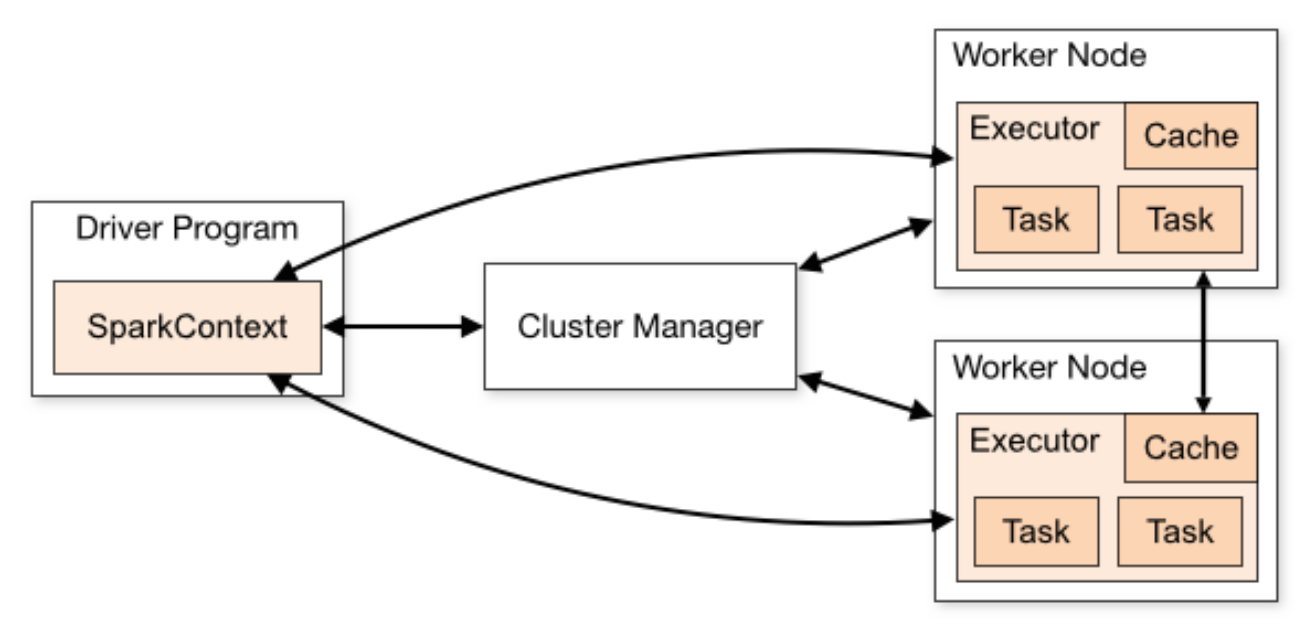

Batch
Streaming
SQL
Dataset
DataFrame
Structured Streaming
RDD
DStream
Structured APIs
Low Level APIs

Spark RDDs

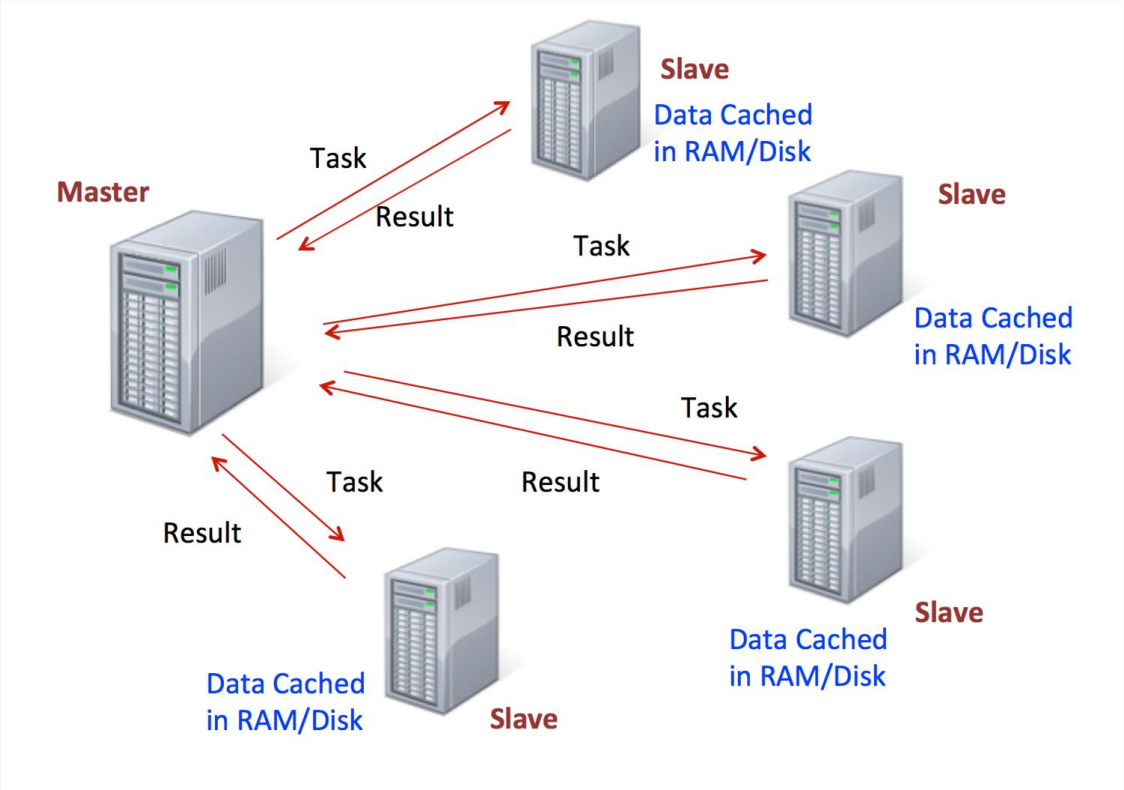
- At the core of Spark is the idea of a Resilient Distributed Dataset (RDD)

- Resilient Distributed Dataset (RDD) has 4 main features:
  - Distributed Collection of Data
  - Fault-tolerant
  - Parallel operation - partioned
  - Ability to use many data sources

Spark RDDs


Worker Node
Executor
Cache
Task
Task
Driver Program
SparkContext
Cluster Manager
Worker Node
Executor
Cache
Task
Task

Spark RDDs

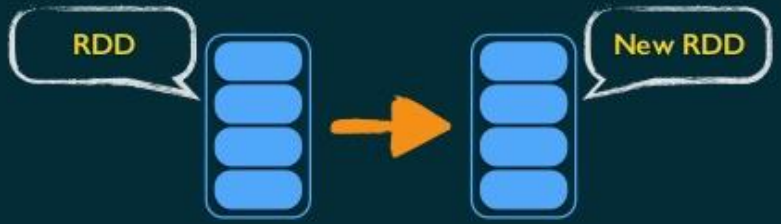

Slave
Data Cached
in RAM/Disk
Task
Result
Master
Slave
Task
Data Cached
in RAM/Disk
Result
Task
Task
Result
Result
Slave
Data Cached
in RAM/Disk
Data Cached
in RAM/Disk
Slave

Spark RDDs

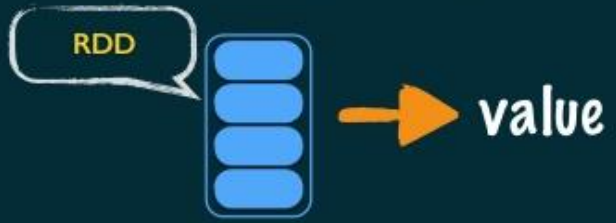
- RDDs are immutable, lazily evaluated, and cacheable
- There are two types of Spark operations:
  - Transformations
  - Actions
- Transformations are basically a recipe to follow.
- Actions actually perform what the recipe says to do and returns something back.

Spark RDDs Operations

1. **Transformations**: define new RDDs based on current one, e.g., filter, map, reduce, groupBy, etc.



2. **Actions**: return values, e.g., count, sum, collect, etc.



Spark RDDs

- This behaviour carries over to the syntax when coding.
- A lot of times you will write a method call, but won't see anything as a result until you call the action.
- This makes sense because with a large dataset, you don't want to calculate all the transformations until you are sure you want to perform them!

Spark RDDs

- When discussing Spark syntax you will see RDD versus DataFrame syntax show up.

- With the release of Spark 2.0, Spark is moving towards a DataFrame based syntax, but keep in mind that the way files are being distributed can still be thought of as RDDs, it is just the typed out syntax that is changing

Spark RDDs

- We’ve covered a lot!
- Don’t worry if you didn’t memorize all these details, a lot of this will be covered again as we learn about how to actually code out and utilize these ideas!

Spark DataFrame

- Spark DataFrames are also now the standard way of using Spark's Machine Learning Capabilities.
- Spark DataFrame documentation is still pretty new and can be sparse.
- Let's get a brief tour of the documentation!

http://spark.apache.org/