

Tutorial Big Data CLEI 2019



Ciudad de Panamá, Panamá

Por: Edwin Montoya
emontoya@eafit.edu.co

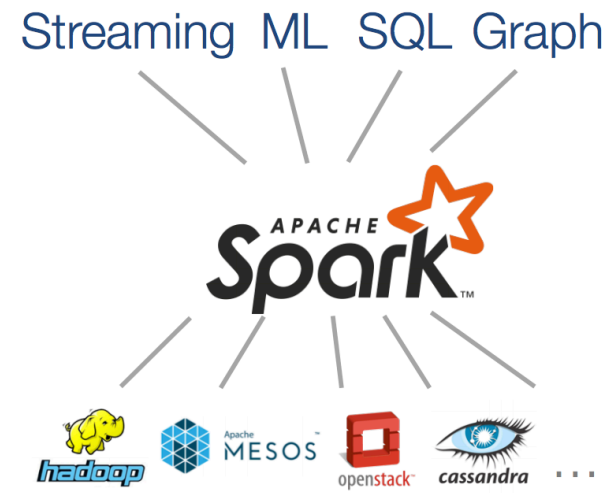
Apache Spark

APACHE SPARK

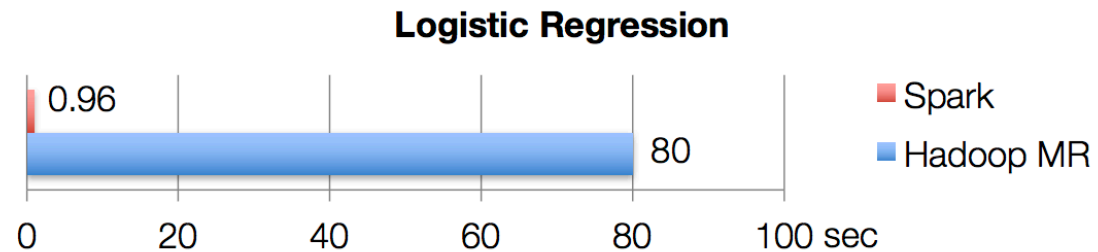
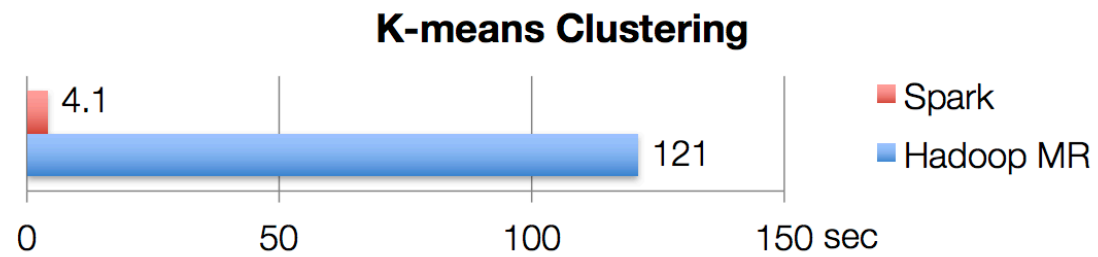
- Research project at UC Berkeley in 2009
- Fast and general-purpose cluster computing system
- APIs: Scala, Java, Python, R, and SQL
- Built by more than 1,000 developers from more than 200 companies

What is Apache Spark?

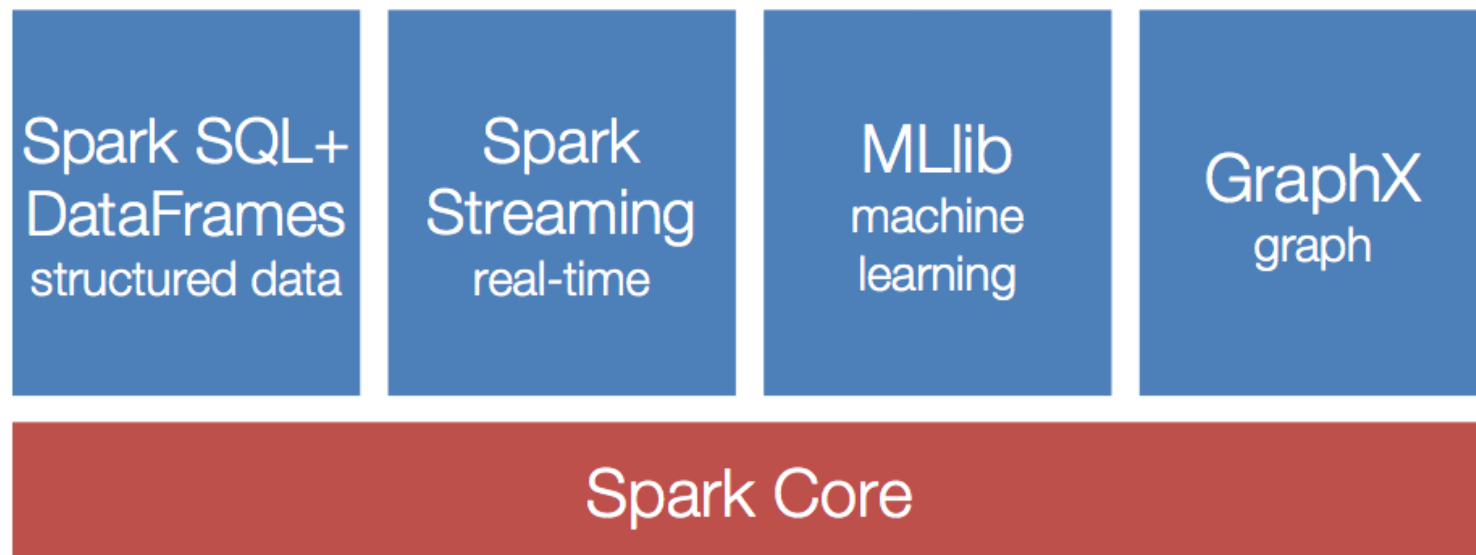
- Open source computing engine for clusters
- Many API and libraries
 - API: scala, java, python, R
 - SQL, ML, Graphs, Streaming



Throughput

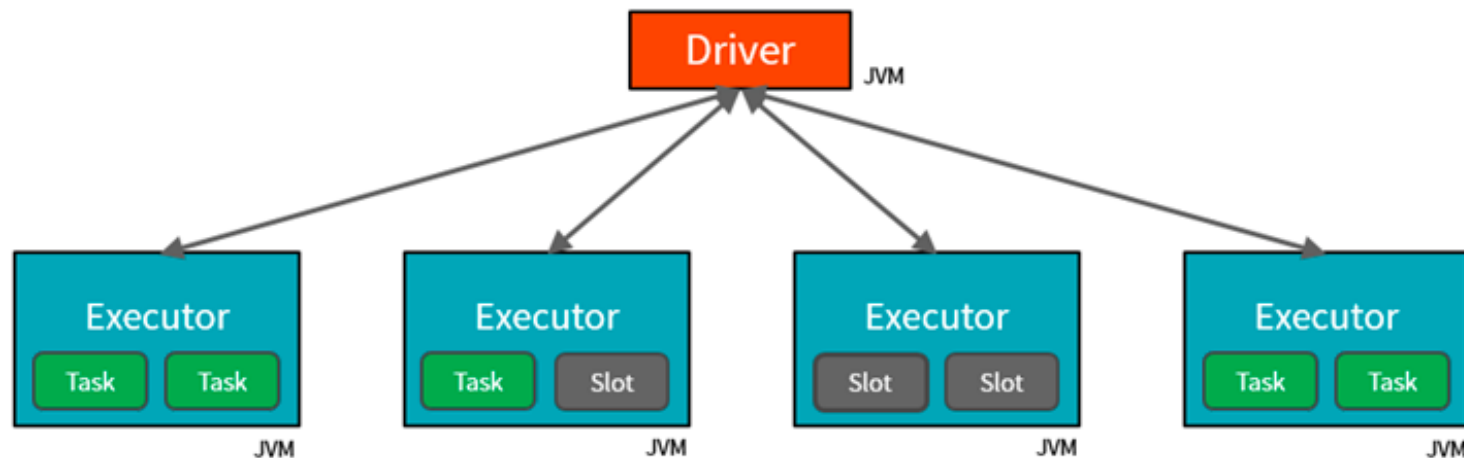


Libraries Built on Spark



SPARK CLUSTER

- One Driver and many Executor JVMs



RDD

- *Resilient*: Fault-tolerant
- *Distributed*: Computed across multiple nodes
- *Dataset*: Collection of partitioned data
- Immutable once constructed
- Track lineage information
- Operations on collection of elements in parallel

Basic Idea

- Write apps in terms of transformations on distributed datasets
- Resilient distributed datasets (RDDs)
 - Objects across the cluster
 - Transformations: Operations over RDD (map, filter, groupBy, etc)
 - Actions (count, collect, save, etc)
- In-memory computing

Transformations

Actions

Map

Reduce

Filter

Count

Sample

TakeSample

Union

Take

ReduceByKey

Collect

DATAFRAME

- Data with columns (built on RDDs)
- Improved performance via optimizations

Construct DataFrames via:

- Transforming existing DataFrame
- Files in storage system (e.g. HDFS, SQL DW)
- Parallelizing existing collections (e.g. Pandas DF)

Spark SQL & DataFrames

APIs for *structured data* (table-like data)

- » SQL
- » DataFrames: dynamically typed
- » Datasets: statically typed

Similar optimizations to relational databases

DataFrame API

Domain-specific API similar to Pandas and R

» DataFrames are tables with named columns

```
users = spark.sql("select * from users")
```

```
ca_users = users[users["state"] == "CA"]
```

```
ca_users.count()
```

Expression AST

```
ca_users.groupBy("name").avg("age")
```

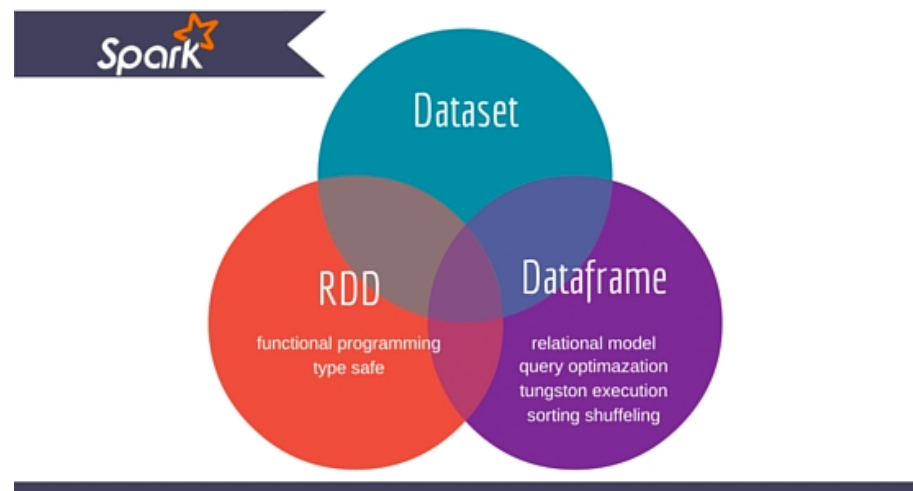
```
caUsers.map(lambda row: row.name.upper())
```

WHY SWITCH TO DATAFRAMES?

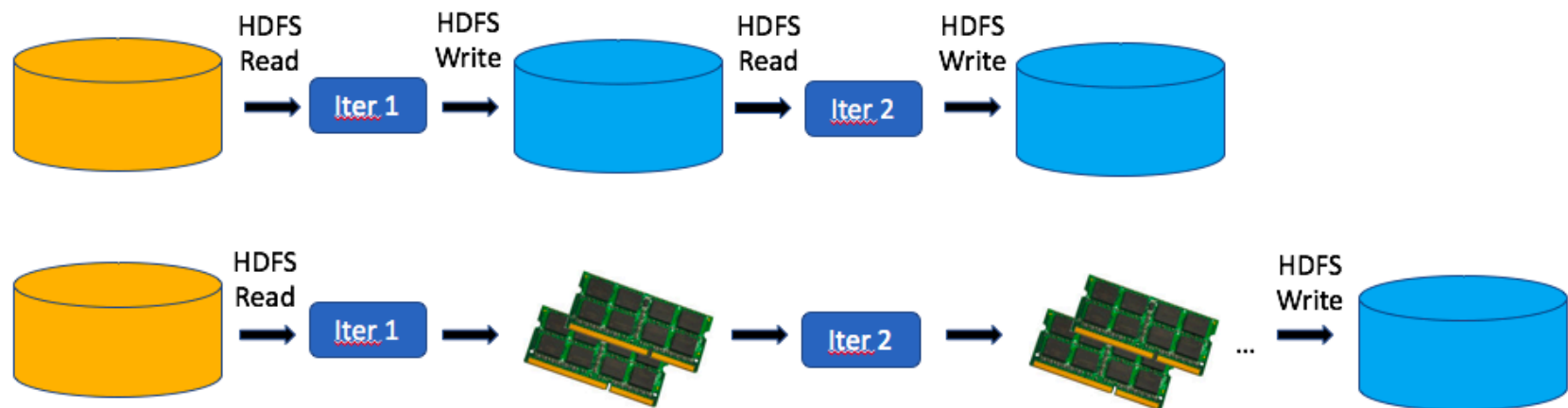
- User-friendly API
- Benefits:
 - SQL/DataFrame queries
 - Tungsten and Catalyst optimizations
 - Uniform APIs across languages
- Facilitate ML Pipelines

DATASETS

- Generalization of DataFrames
- Supports more data types and stronger type enforcement



SPARK VS MAPREDUCE



WHEN TO USE SPARK?

- Scale out: Model or data too large to process on a single machine
- Speed up: Benefit from faster results

Wordcount in pyspark

```
$ pyspark
```

```
>>> lines = sc.textFile("hdfs:///datasets/gutenberg-small/")
```

```
>>> counts = lines.flatMap(lambda line: line.split(" ")).map(lambda word:  
(word,1)).reduceByKey(lambda x,y: x + y)
```

```
>>> output = counts.collect()
```

```
>>> for (word,count) in output:
```

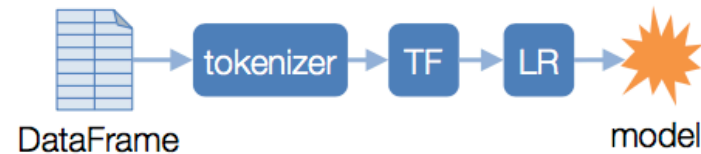
```
...     print("%s: %i" % (word,count))
```

MLlib

High-level *pipeline* API
similar to SciKit-Learn

Acts on DataFrames

Grid search and cross
validation for tuning



```
tokenizer = Tokenizer()
tf = HashingTF(numFeatures=1000)
lr = LogisticRegression()

pipe = Pipeline(
    [tokenizer, tf, lr])
model = pipe.fit(df)
```

MLlib Algorithms

Generalized linear models

Alternating least squares

Decision trees

Random forests, GBTs

Naïve Bayes

PCA, SVD

AUC, ROC, f-measure

K-means

Latent Dirichlet allocation

Power iteration clustering

Gaussian mixtures

FP-growth

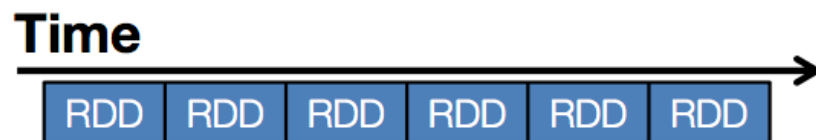
Word2Vec

Streaming k-means

Spark Streaming



Spark Streaming



Represents streams as a series of RDDs over time

```
val spammers = sc.sequenceFile("hdfs://spammers.seq")

sc.twitterStream(...)
  .filter(t => t.text.contains("Stanford"))
  .transform(tweets => tweets.map(t => (t.user, t)).join(spammers))
  .print()
```