PySpark: Real-time large-scale data processing with Python and Spark

Philipp Pahl
PyData Berlin Meetup
11/11/2014

Outline

- What is Spark
  - Spark stack and architecture
- PySpark: Spark and Python
- PySpark demo

Apache Spark

- Data processing framework
  - fast and general purpose
  - in-memory
  - rich APIs in Scala, Java and Python
  - reads data from local disk, HDFS (any Hadoop data source), S3
  - replacement/successor of MapReduce?
- Sparks' scope
  - Data warehousing
    - batch and real-time
  - Interactive data analysis
    - supports data exploration
  - Machine learning for iterative algorithms

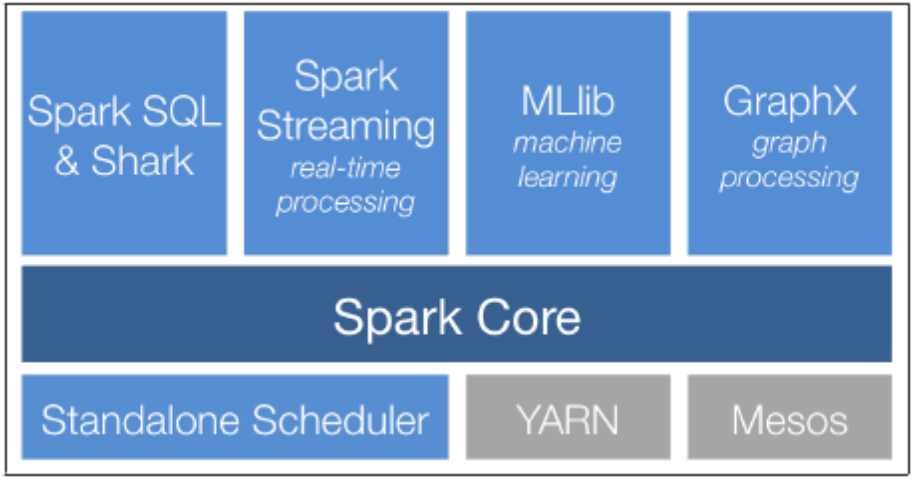
History

- Started in 2009
- Open sourced in 2010
- Today over 400 contributors from 50+ organizations
  - “Yahoo!, Intel, Adobe...”
- Research project of the UC Berkley AMPLab
- databricks: spin-off of the AMPLab
  - “A single platform for big data processing, from the creators of Spark”

Spark Stack

- Spark Core
  - contains basic functionality
  - home of Resilitent Distributed Datasets (RDD)
- Spark SQL
  - Superset of HiveQL (Hive's SQL)
- Spark Streaming
  - live data stream processing

- Different cluster managers
- Input from local disk, HDFS, S3



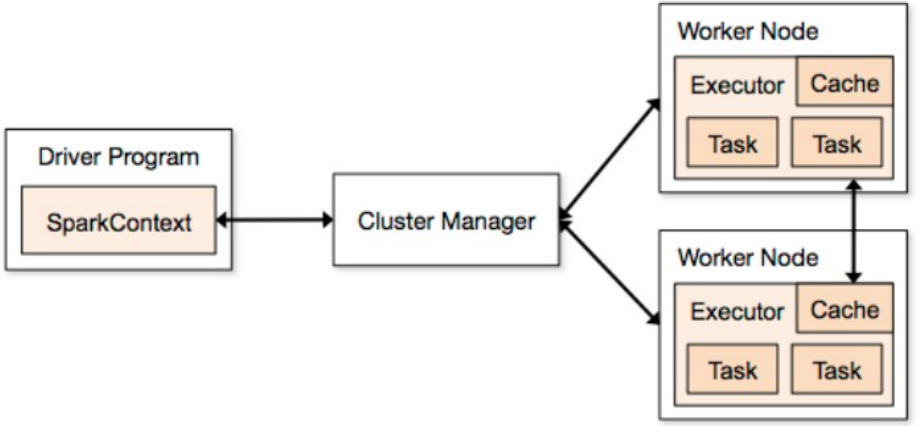
Operators

- Transformations
- Actions
- Set operations

```
map             reduce          take
filter          count           first
groupBy         fold            partitionBy
union           reduceByKey     pipe
join            groupByKey      distinct
leftOuterJoin   cogroup         save
rightOuterJoin  flatMap         ...
```

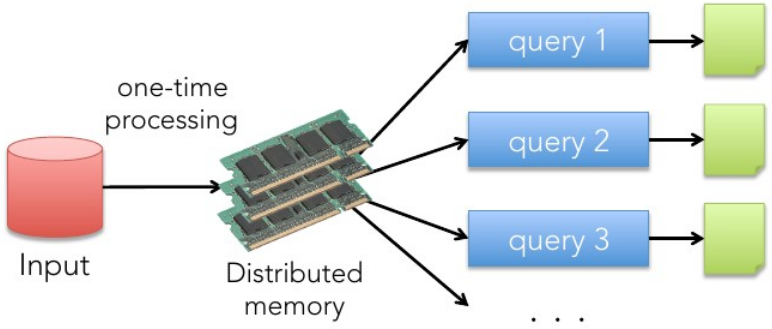
Spark Application

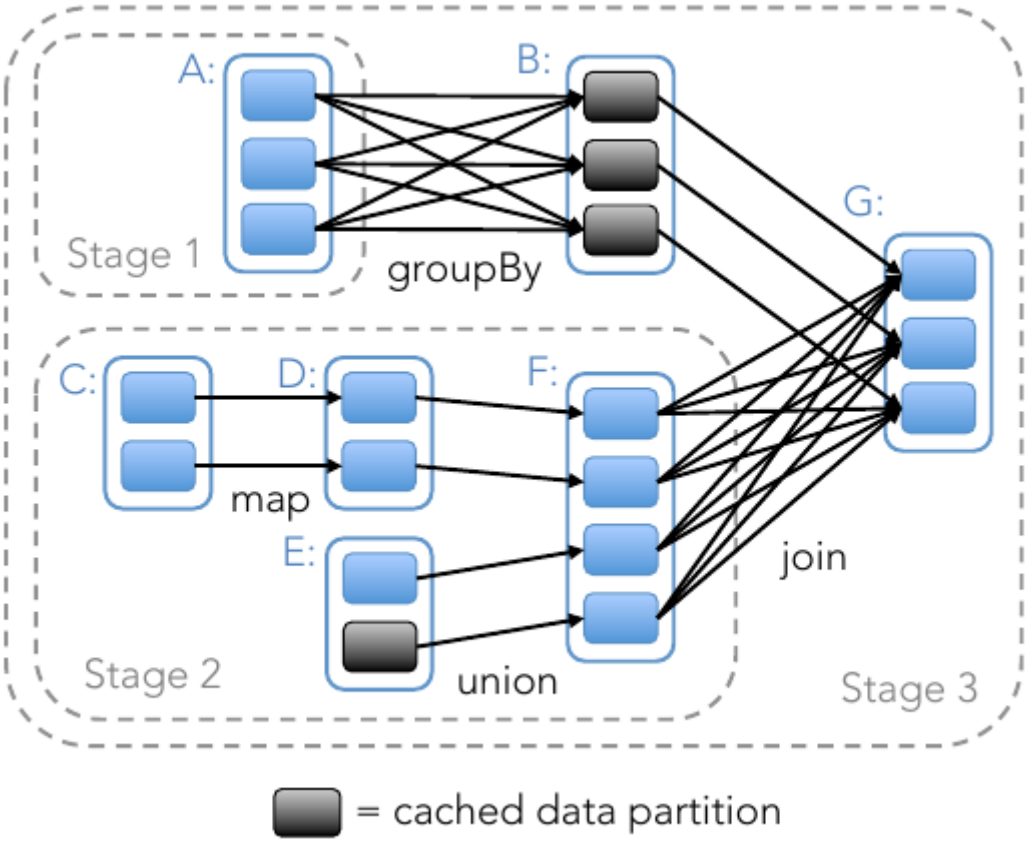
- Run as independent sets of processes on a cluster
- Coordinated by the SparkContext (driver program)
- Connects to several types of cluster managers
- Once connected, Spark acquires executors
- Next, it sends the application code to executors



Job Scheduler

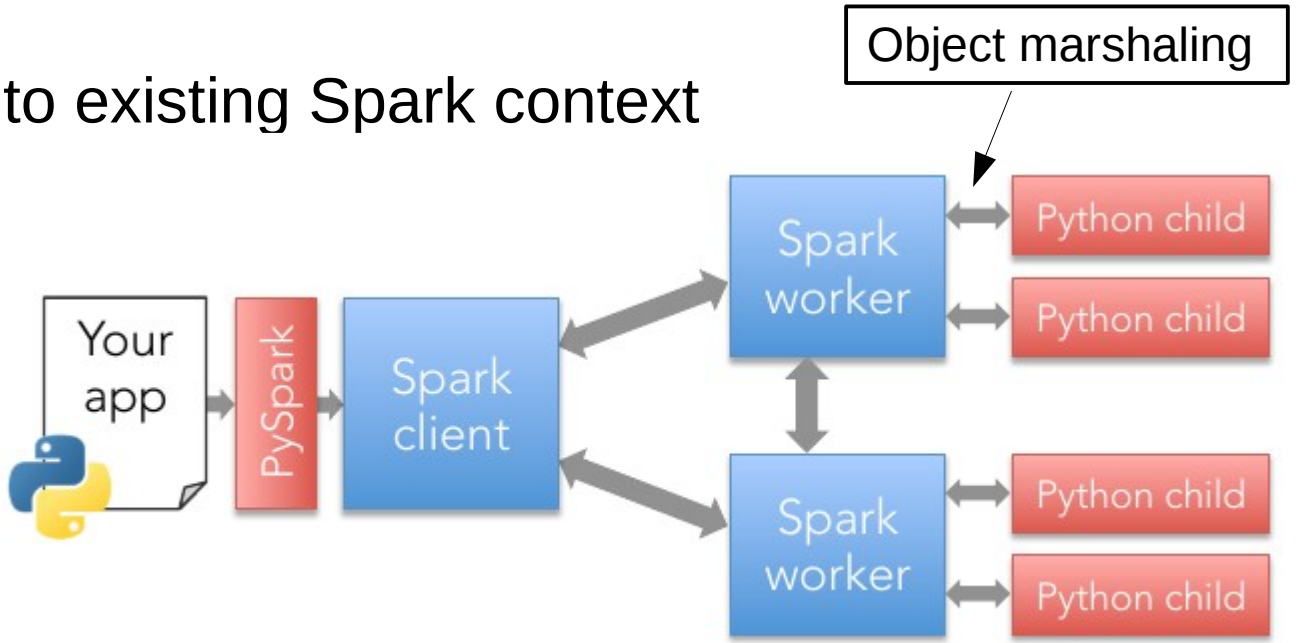
- General operator graphs
- Aware of data locality and partitioning
- Lazy evaluation





PySpark

- Spark core is written in Scala
  - JVM language
- PySpark interfaces to existing Spark context
- 2K lines of Python
- Native Python
- No changes to Python

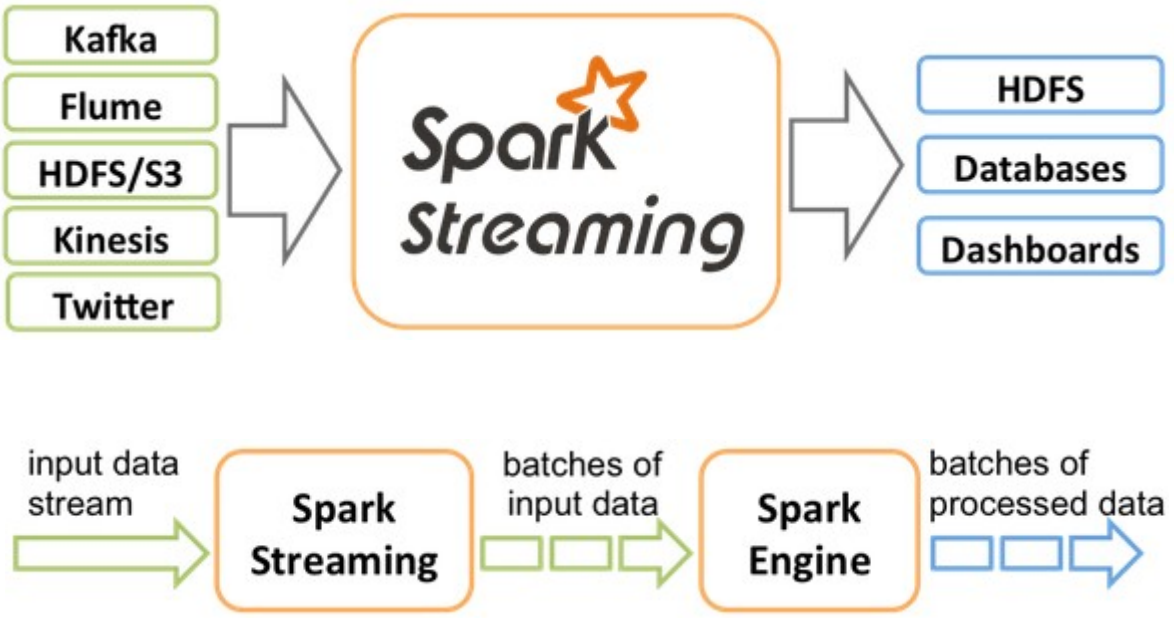


Spark SQL

```
# sc is an existing SparkContext.
from pyspark.sql import SQLContext, Row
sqlContext = SQLContext(sc)

# Load a text file and convert each line to a dictionary.
lines = sc.textFile("examples/src/main/resources/people.txt")
parts = lines.map(lambda l: l.split(","))
people = parts.map(lambda p: Row(name=p[0], age=int(p[1])))

# Infer the schema, and register the SchemaRDD as a table.
schemaPeople = sqlContext.inferSchema(people)
schemaPeople.registerTempTable("people")

# SQL can be run over SchemaRDDs that have been registered as a table.
teenagers = sqlContext.sql("SELECT name FROM people WHERE age >= 13 AND age <= 19")

# The results of SQL queries are RDDs and support all the normal RDD operations.
teenNames = teenagers.map(lambda p: "Name: " + p.name)
for teenName in teenNames.collect():
  print teenName
```

Spark Streaming

- Enables processing of live stream data
  - message queues
- Same API
  - Reuse code in batch and real-time views



Conclusion

- Spark is “the” candidate for being the successor of MapReduce
- Data warehousing, ML
- PySpark provides fast and simple way to analyze big datasets from Python
  - Data exploration
  - Rapid prototyping

- Questions? Support needed?
  - Get in touch with me
  - linkedin.com/in/philipppahl