MSAN 694 : Distributed Computing

Diane Woodbridge, Ph.D.
MSAN, University of San Francisco

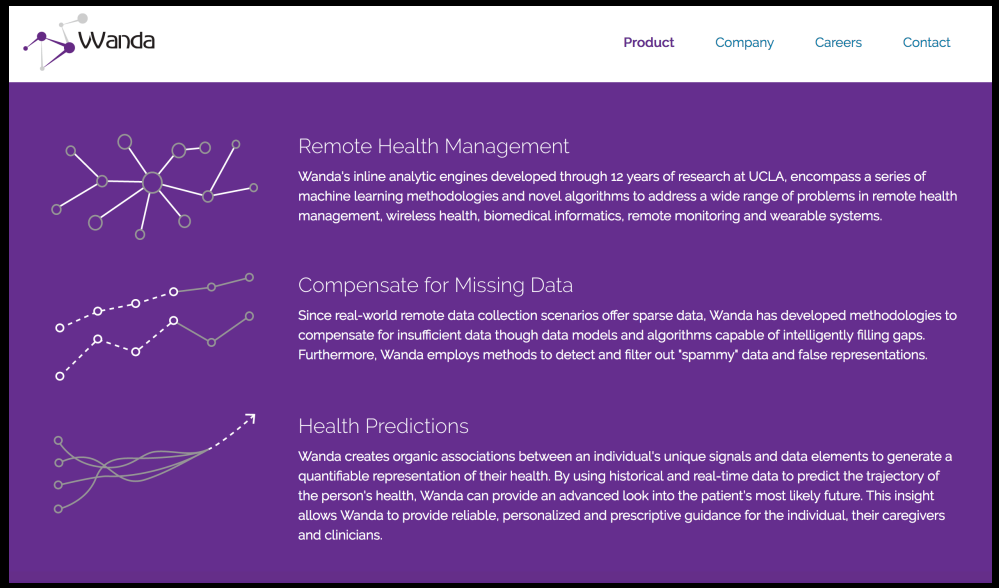
About Diane

Technical Me

Ph.D. (and M.S) in Computer Science, UCLA. 2012, 2010

SMTS, Sandia National Laboratories. 2012 – 2016

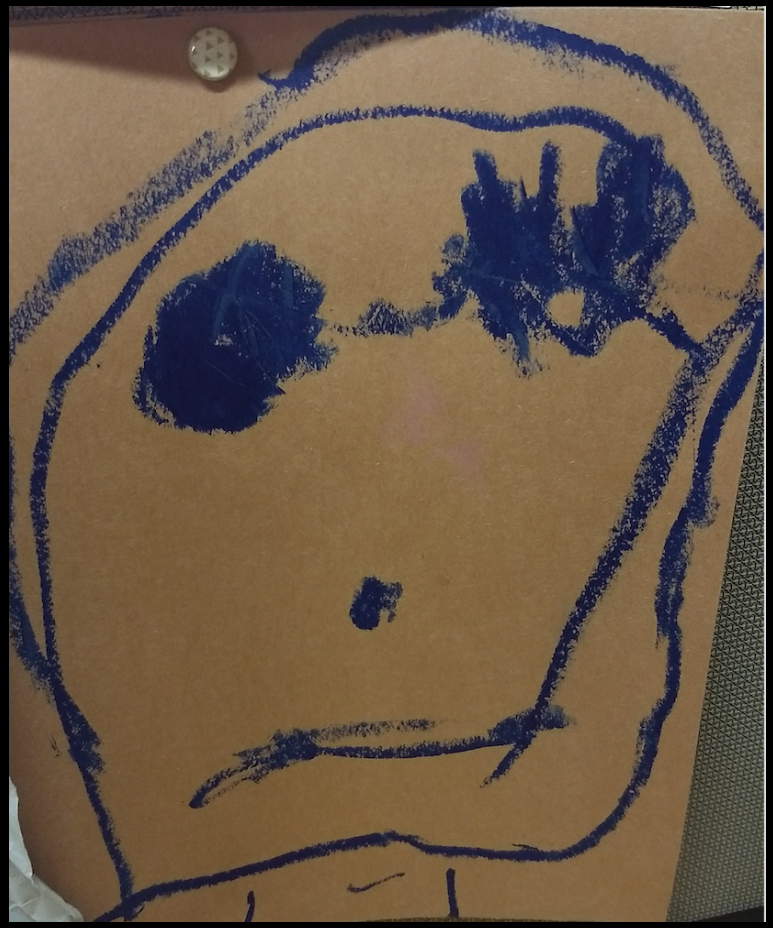
Research Interest: Database management and machine learning in health, remote sensing, etc.

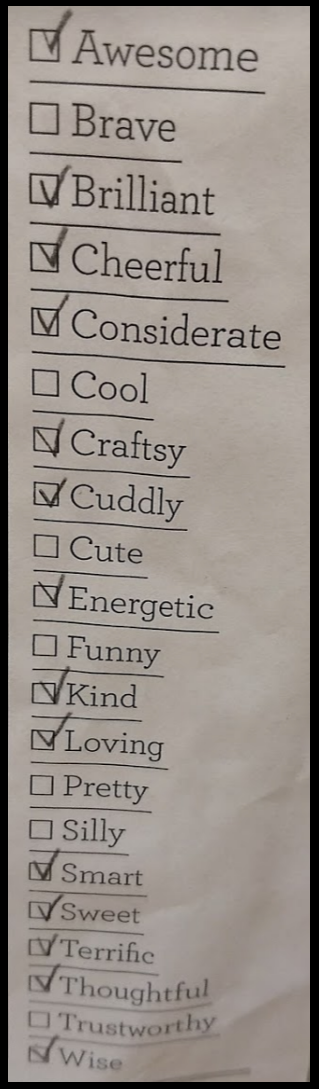


DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

About Diane

Non-technical Me





DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

Contents

Contents

Class Overview

Motivation - Why Distributed Computing?

What is Distributed Computing?

Spark

RDD Creation

Course Objectives

Understand needs and concepts of distributed computing.

Understand the Spark and its stack.

Being competent to work with Spark on a distributed computing environment.

1. Program with RDDs.
2. Work with key/value pairs.
3. Work on Amazon AWS.

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

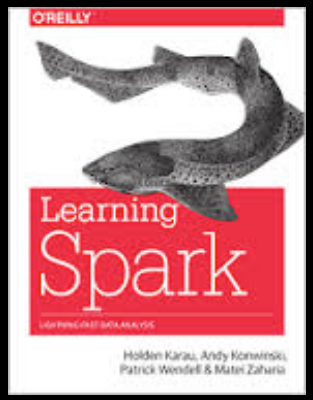
Reference Materials

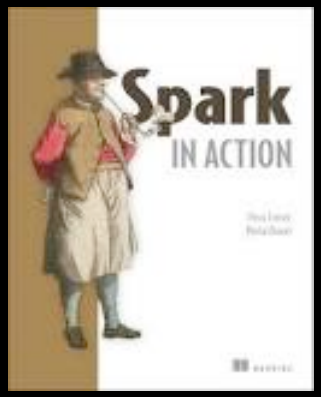
Spark Online Documentation, http://spark.apache.org/docs/latest/

Karau, Holden, et al. Learning spark: lightning-fast big data analysis. O'Reilly Media, Inc., 2015.

Zecevic, Petar, et al. Spark in Action, Manning, 2016.

AWS Online Documentation, https://aws.amazon.com/documentation/





DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

Course Evaluation

Attendance (10%)

- Physically.
- No cellphones, No Slacks, No Social Media!

Homework (25% - 5 Assignments)

- No late submission/resubmission allowed!!
- We are going to run the grading code only once.

Quiz (Programming, multiple choices/short answers) (40% - 20% x 2)

- **October 31st** and **November 14th, 9 am to 10 am** (101 Howard Room 527 and 529)

Final (25%)

- **December 11th, 10 am to 12 pm** (101 Howard Room 154, 155 and 156)

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

Student Supports

Office Hours

- Tuesday 12:00 - 1:00 PM (Room 522)

Teaching Assistant

- Jinxin Ma (email - jma33@dons.usfca.edu)
- TA office hours : Thursday 4:00 - 5:00 PM (Room 515)

Others

Example Data

- https://github.com/dianewoodbridge/2017-msan694-example.git

Poll

- https://pollev.com/dianewoodbri311

Last Year

Machine Learning-based Product Recommendation using Apache Spark

Lin Chen *, Rui Li *, Yige Liu *, Ruixuan Zhang *, Diane Myung-kyung Woodbridge
{lchen74,rli33,yliu225,rzhang45,dwoodbridge}@usfca.edu
Master of Science in Analytics (MSAN) Program,
University of San Francisco,
San Francisco, California

IEEE Workshop on Data Science and Computational Intelligence (Accepted)

Medhere : A Smartwatch-based Medication Adherence Monitoring System Using Machine Learning and Distributed Computing

Jinxn Ma
jma33@dons.usfca.edu
Analytics Program
University of San Francisco

Anaelia Ovalle
aovalle@dons.usfca.edu
Data Science
University of San Francisco

Diane Myung-kyung Woodbridge
dwoodbridge@usfca.edu
Analytics Program
University of San Francisco

IEEE Conference on Biomedical and Health Informatics (Under submission)

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

Spark Interview Questions

What is Apache Spark?

Explain the key features of Spark.

What is RDD?

How to create RDD.

What is ”partitions”?

Types of RDD operations?

What is “transformation”?

What is “action”?

Functions of “spark core”?

What is “spark context”?

What is an “RDD lineage”?

Which file systems does Spark support?

List the various types of “Cluster Managers” in Spark.

What is “YARN”?

What is “Mesos”?

What is a “worker node”?

What is an “accumulator”?

What is “Spark SQL” (Shark)?

What is “SparkStreaming”?

What is “GraphX”?

What is “MLlib”?

https://www.edureka.co/blog/interview-questions/top-apache-spark-interview-questions-2016/

Spark Interview Questions

What are the advantages of using Apache Spark over Hadoop MapReduce for big data processing?

What are the languages supported by Apache Spark for developing big data applications?

Can you use Spark to access and analyze data stored in Cassandra databases?

Is it possible to run Apache Spark on Apache Mesos?

How can you minimize data transfers when working with Spark?

Why is there a need for broadcast variables?

Name a few companies that use Apache Spark in production.

What are the various data sources available in SparkSQL?

What is the advantage of a Parquet file?

What do you understand by Pair RDD?

Is Apache Spark a good fit for Reinforcement learning?

https://www.dezyre.com/article/top-50-spark-interview-questions-and-answers-for-2017/208

Contents

Class Overview

Motivation - Why Distributed Computing?

What is Distributed Computing?

Spark

RDD Creation

Long line

How to move faster?

Max()

Contents

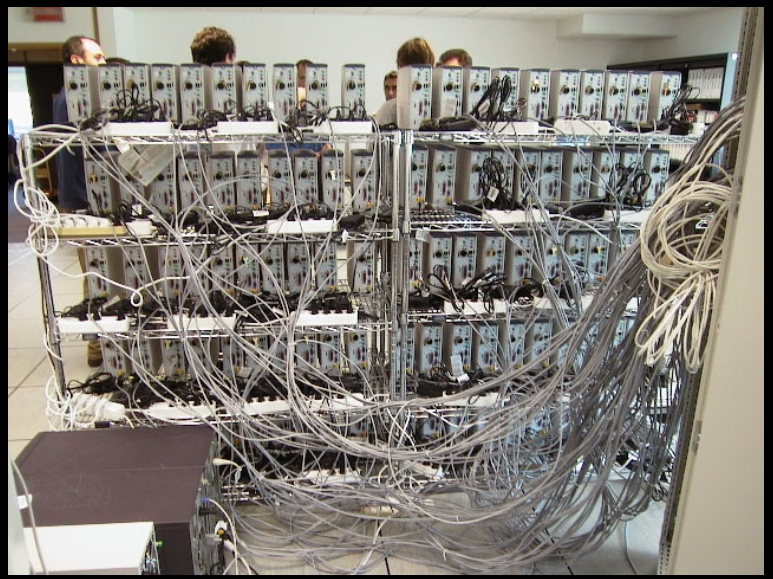
What is Distributed Computing?

Cluster

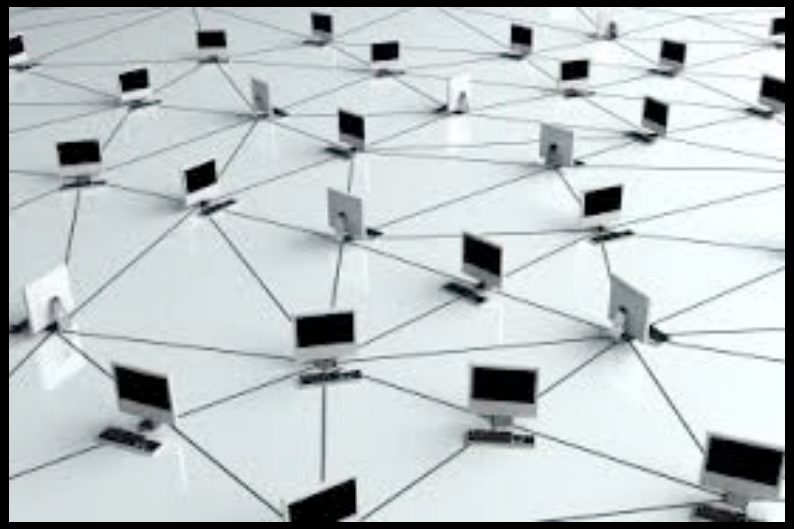
- Collection of systems that work together to perform functions.

Node

- Individual servers within a cluster.



What is Distributed Computing?



For processing large volumes of data,

- “Scale out” instead of scale up.
  - Cheaper : Run large data on clusters of many smaller and cheaper machines.
  - Reliable (Fault Tolerant): If one node or process fails, its workload should be assumed by other components in the system.
  - Faster : It parallelizes and distributes computations.

MapReduce - Concept

Map-Reduce: Allow computations to be parallelized over a cluster.

- Basic Map-Reduce
  - The map-reduce framework plans map tasks to be run on the correct nodes and shuffle data for the reduce function.
  - Map : Apply a function to each key-value pair over a portion of data in parallel. ex. filter()
  - Reduce : Return one key-value pair from multiple key-value pairs. ex. sum(), count()

Walmart Order

OrderID: 1
Customer : Diane
Timestamp : 2017-08-15 05:04:32 PST
Items:
{ProductName : San Francisco Giants Hat M,
{Qty: 2, UnitPrice : 10, Price : 20}},
{ProductName : San Francisco Giants Hat S,
{ Qty: 3, UnitPrice : 8, Price : 24}}
Shipping: Corte Madera

{San Francisco Giants Hat M, {Qty:2, Price : 20}

{San Francisco Giants Hat S, {Qty:3, Price : 24}

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

MapReduce - Concept

Map-Reduce: Allow computations to be parallelized over a cluster.

- Basic Map-Reduce
  - The map-reduce framework plans map tasks to be run on the correct nodes and shuffle data for the reduce function.
  - Map : Apply a function to each key-value pair over a portion of data in parallel. ex. filter()
  - Reduce : Return one key-value pair from multiple key-value pairs. ex. sum(), count()

Hadoop MapReduce

Open source, distributed, Java computation framework consisting of Hadoop Common, Hadoop Distributed File System (HDFS), YARN and MapReduce.

Hadoop MapReduce solved issues of..

- Distribution : Distribute the data.
- Parallelism : Perform subsets of the computation simultaneously.
- Fault Tolerance : Handle component failure.

Hadoop MapReduce

Limitations

- Hadoop MapReduce is powerful, but can be slow.
  - MapReduce job results need to be stored in HDFS (disk) before they can be used by another job. ➔ Slow with iterative algorithms.
- Many kinds of problems don't easily fit MapReduce's two-step paradigm.
- Hadoop is a low-level framework, so myriad tools have sprung up around it and brings additional complexity and requirements.

Contents

Class Overview

Motivation - Why Distributed Computing?

What is Distributed Computing?

Spark

RDD Creation

Hadoop MapReduce vs. Spark

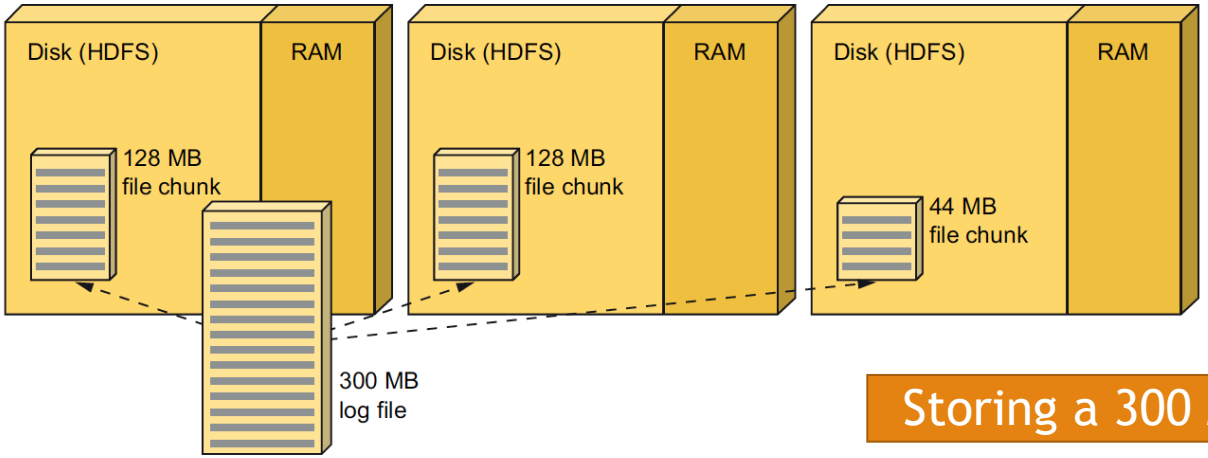
	Hadoop MapReuce	Spark
Speed	Decently fast	100 times faster than Hadoop
Ease of Use	No interactive modes and Hard to learn	Provides interactive modes and Easy to learn
Costs	Open source	Open source
Data Processing	Batch Processing	Batch Processing + Streaming
Fault Torelence	Fault Torelent	Fault Torelent
Security	Kerberos authentication	Password authentification

Hadoop MapReduce vs. Spark

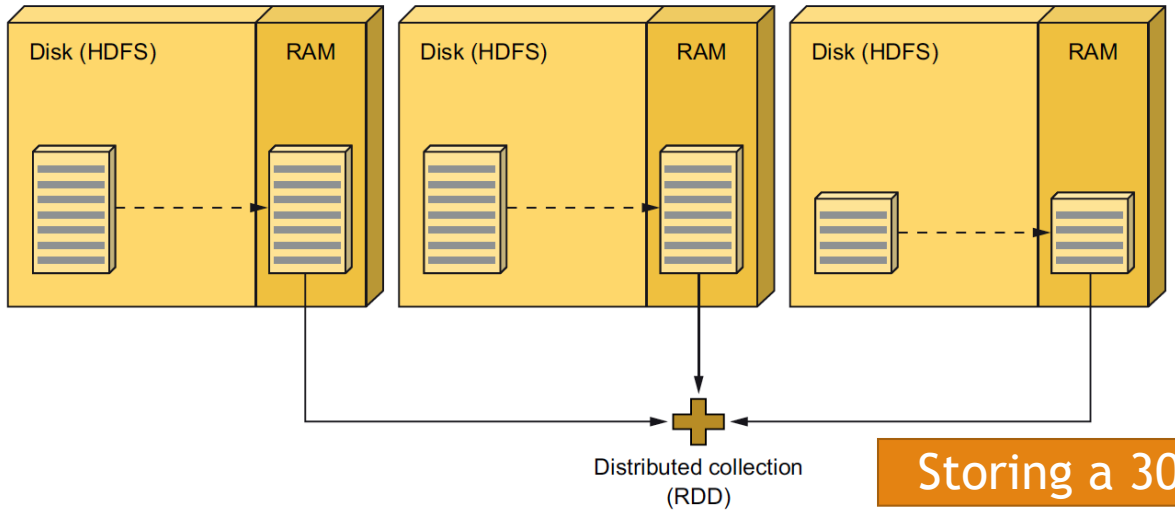
Spark

- It keeps large amounts of data in memory which offers tremendous performance improvements. (x100 faster than Hadoop MapReduce.) ➔ Good for iterative algorithms such as machine learning, graph algorithms and others that need to reuse data.

Hadoop MapReduce vs. Spark



Storing a 300 MB file in a 3-node Hadoop cluster.



Storing a 300 MB file in a 3-node Spark cluster.

Hadoop MapReduce vs. Spark

Spark

- You can write distributed programs in a manner similar to writing local programs.
  - Spark's collections abstract away the fact that they're potentially referencing data distributed on a large number of nodes.
- Spark combines MapReduce like capabilities for batch programing, real time data processing, SQL-like handling of structured data, graph algorithms and machine learning all in a single frame work.

Extends the MapReduce model with primitives for efficient data sharing (Using Resilient Distributed Datasets (RDDs)).

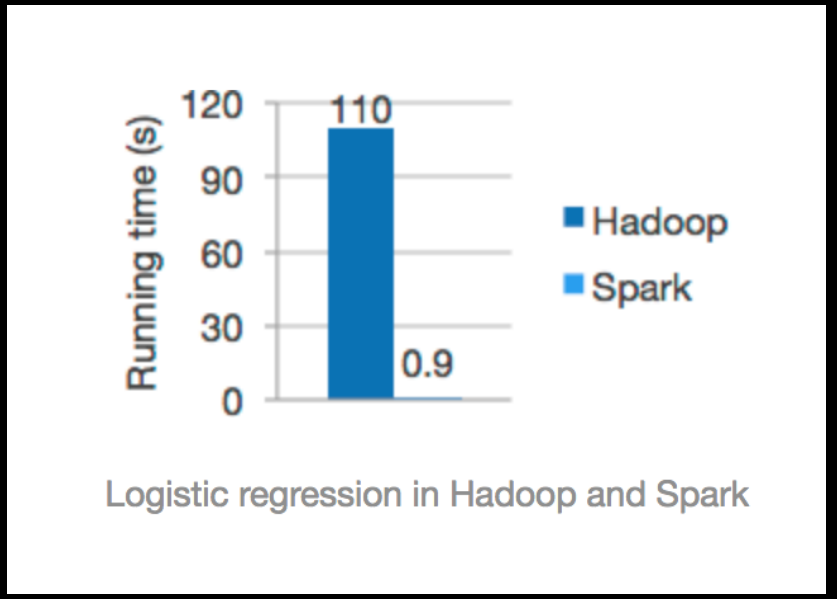
Achieves

1. Speed.
2. Ease of Use.
3. Generality.
4. Runs everywhere.

http://spark.apache.org/

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

1. Speed: Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.
   - Apache Spark has an advanced DAG execution engine that supports cyclic data flow and in-memory computing.



http://spark.apache.org/

2. Ease of Use : Write applications quickly in Java, Scala, Python, R.

- Spark offers over 80 high-level operators that make it easy to build parallel apps. This allows developer to focus on logic rather than infrastructure.
- Developers can use Spark interactively from the Scala, Python and R shells.

```
text_file = spark.textFile("hdfs://...")

text_file.flatMap(lambda line: line.split())
    .map(lambda word: (word, 1))
    .reduceByKey(lambda a, b: a+b)
```

Word count in Spark's Python API

http://spark.apache.org/

MapReduce word count

Main class

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.TextOutputFormat;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;

public class WordCount
{
	public static void main(String[] args) throws Exception
	{
		JobConf conf = new JobConf(WordCount.class);
		conf.setJobName("Hadoop wordcount");
		conf.setOutputKeyClass(Text.class);
		conf.setOutputValueClass(IntWritable.class);
		conf.setMapperClass(WordCountMap.class);
		conf.setCombinerClass(WordCountReduce.class);
		conf.setReducerClass(WordCountReduce.class);
		conf.setInputFormat(TextInputFormat.class);
		conf.setOutputFormat(TextOutputFormat.class);
		FileInputFormat.setInputPaths(conf, new Path(args[0]));
		FileOutputFormat.setOutputPath(conf, new Path(args[1]));

		JobClient.runJob(conf);
	}
}
```

Mapper

```
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;

public class WordCountMap extends MapReduceBase
	implements Mapper<LongWritable, Text, Text, IntWritable>
{
	private final static IntWritable one = new IntWritable(1);
	private Text word = new Text();

	public void map(LongWritable key, Text value,
			OutputCollector<Text, IntWritable> output, Reporter reporter)
		throws IOException
	{
		String line = value.toString();
		StringTokenizer tokenizer = new StringTokenizer(line);
		while (tokenizer.hasMoreTokens()) {
		  word.set(tokenizer.nextToken());
		  output.collect(word, one);
		}
	}
}
```

Reducer

```
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;

public class WordCountReduce extends MapReduceBase
	implements Reducer<Text, IntWritable, Text, IntWritable>
{
	public void reduce(Text key, Iterator<IntWritable> values,
			OutputCollector<Text, IntWritable> output, Reporter reporter)
		throws IOException
	{
		int sum = 0;
		while (values.hasNext())
		{
			sum += values.next().get();
		}
		output.collect(key, new IntWritable(sum));
	}
}
```

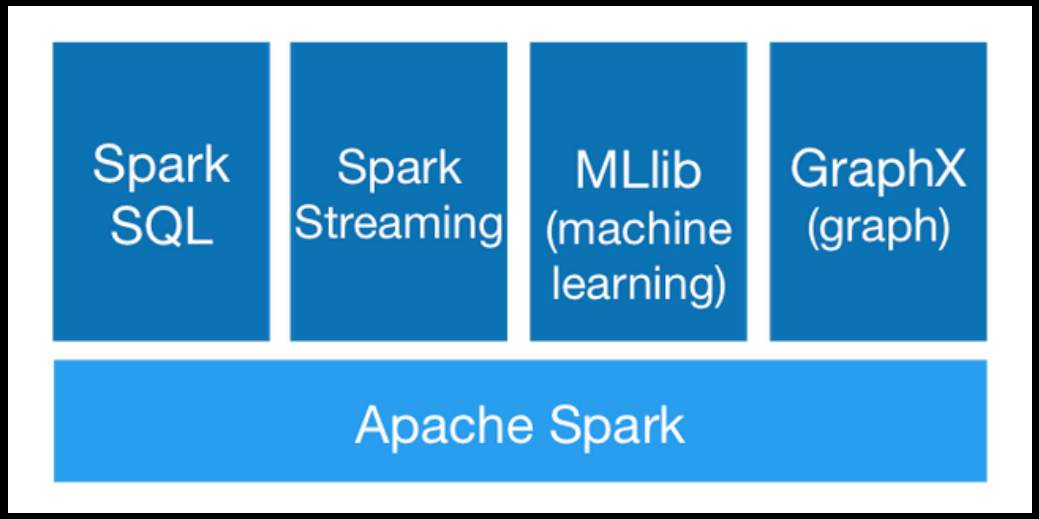
Spark word count

```
import org.apache.spark.{SparkConf, SparkContext}

object SparkWordCount
{
	def main(args : Array[String])
	{
		val conf = new SparkConf().setAppName("Spark wordcount")
		val sc = new SparkContext(conf)
		val file = sc.textFile("hdfs://...")
		val counts = file.flatMap(line => line.split(" "))
			.map(word => (word, 1))
			.reduceByKey(_ + _)
		counts.saveAsTextFile("hdfs://...")
	}
}
```

3. Generality : Combine SQL, streaming, and complex analytics.

- Spark powers a stack of libraries including SQL and DataFrames, MLlib and ML for machine learning, GraphX, and Spark Streaming. You can combine these libraries seamlessly in the same application.



http://spark.apache.org/

4. Runs everywhere : Spark runs on Hadoop, Mesos, standalone, or in the cloud. It can access diverse data sources including HDFS, Cassandra, HBase, and S3.
   - You can run Spark using its standalone cluster mode, on Hadoop YARN, or on Apache Mesos. Access data in HDFS, Cassandra, HBase, Hive, Tachyon, and any Hadoop data source.

http://spark.apache.org/

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

Community

- Most active open source community for big data.
- 1000+ developers, 200+ companies contributing.
- 500+ companies are actively using for production.

http://go.databricks.com/big-ebook-case-studies

http://spark.apache.org/powered-by.html

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

Spark Stack

Cluster Managers

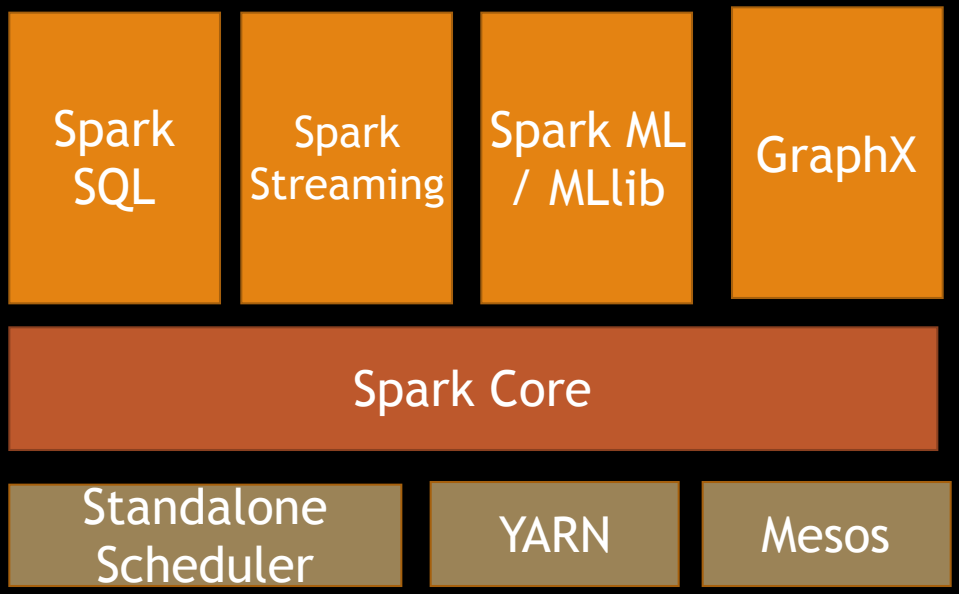
Spark Core

Spark SQL

Spark Streaming

Spark MLlib and ML

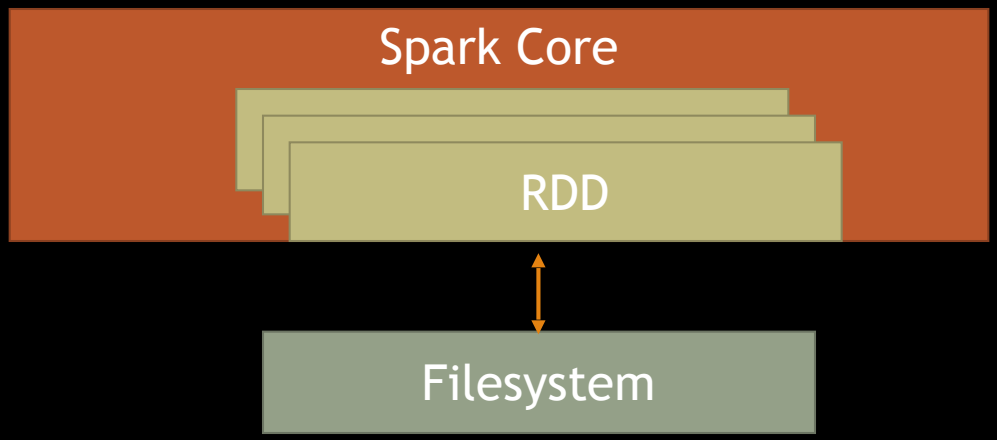
GraphX



Spark Components

Spark Core

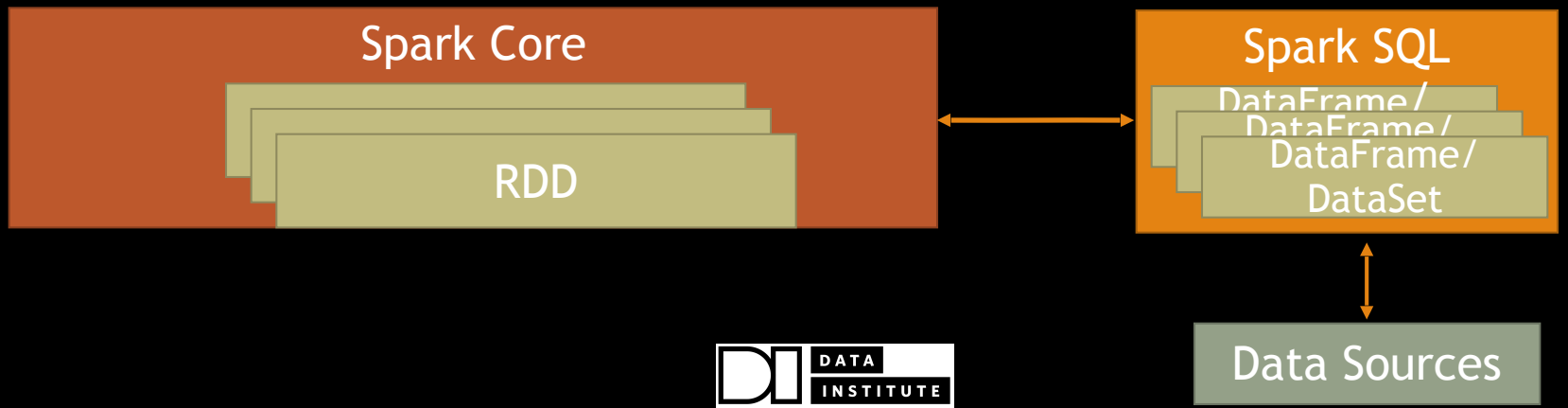
- Contains Spark functionalities required for running jobs and needed by other components.
  - Resilient Distributed Dataset (RDD) : Abstraction of a distributed collection of items with operations and transformation applicable to the dataset.
  - Fundamental functions such as networking, security, scheduling and data shuffling.
  - Logic for accessing various filesystems including HDFS, GlusterFS, Amazon S3.



Spark Components

Spark SQL

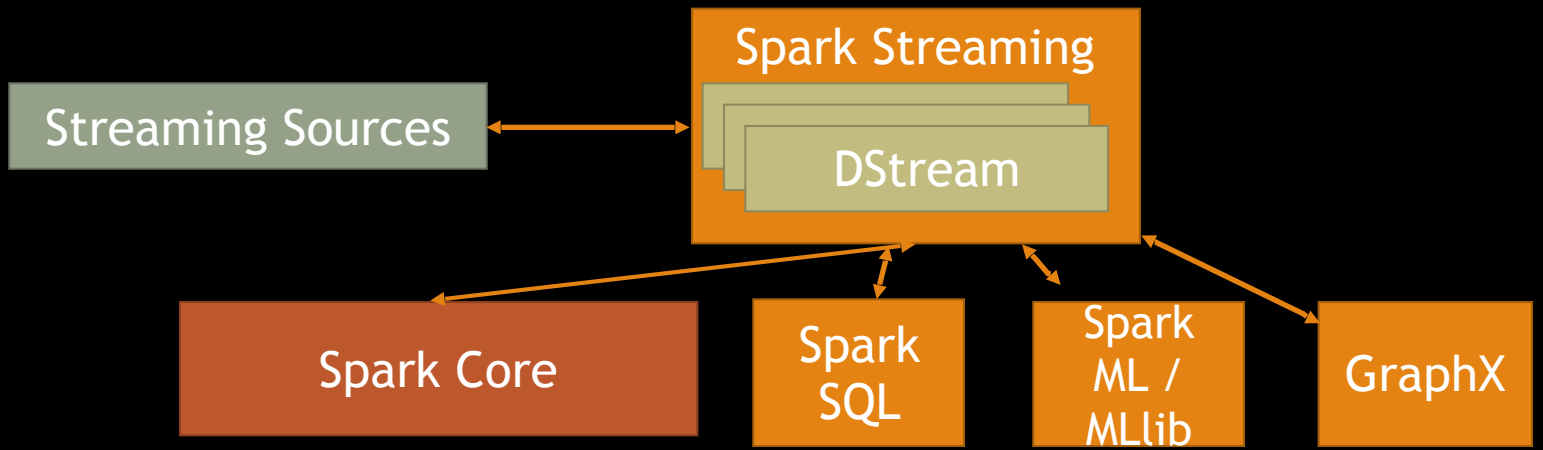
- Provides functions for manipulating large sets of distributed, structured data using an SQL subset.
- Uses **`DataFrames`** and **`DataSets`**
  - Spark SQL transforms operations on DataFrames/ DataSets to operations on RDDs.
- Data Sources include Hive, JSON, relational databases, NoSQL databases and Parquet files.



Spark Components

Spark Streaming

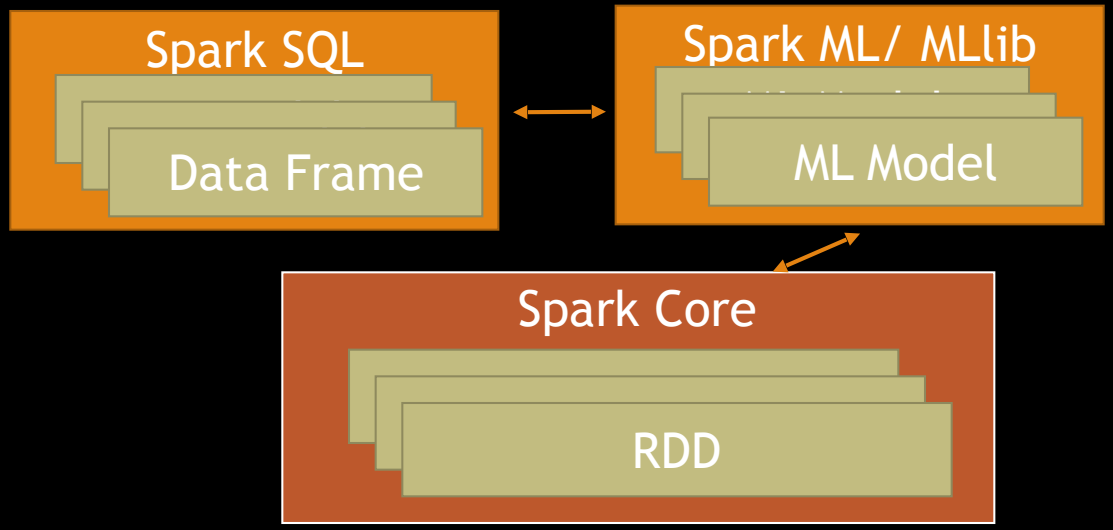
- Ingest real-time streaming data from various sources including HDFS, Kafka, Flume, Twitter, ZeroMQ and custom ones.
- Recover from failure automatically.
- Represent streaming data using discretized streams (Dstreams), which periodically create RDDs containing the data that came in during the last time window.
- Can be combined with other Spark components (Spark Core, SparkML and Mllib, GraphX, Spark SQL)



Spark Components

Spark MLlib and ML

- Library of machine learning algorithms
- Include logistic regression, naïve Bayes, support vector machine, decision trees, random forests, linear regression and k-mean clustering.
- MLlib : RDD-based APIs.
- Spark ML : DataFrame-based APIs.

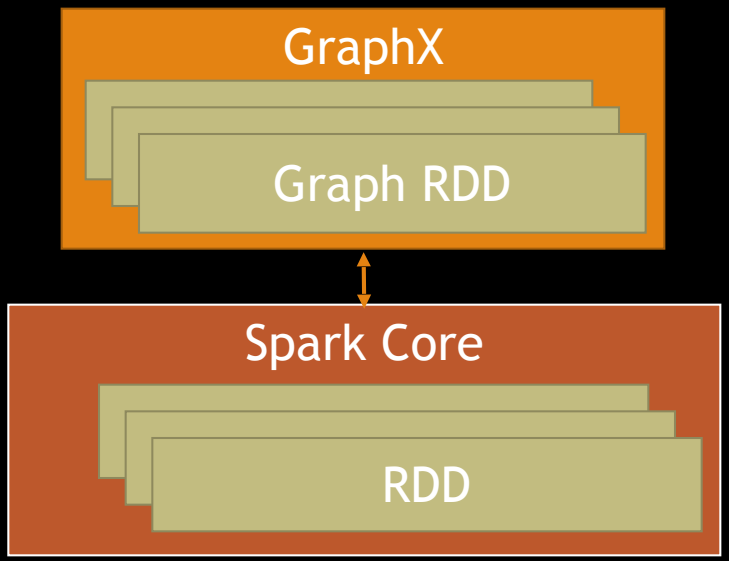


http://spark.apache.org/docs/latest/ml-guide.html

Spark Components

Spark GraphX

- Provide functions for building graphs, represented as graph RDDs : **`EdgeRDD`** and **`VertexRDD.`**
- Contain important algorithms of graph theory such as page rank, connected components, shortest paths, SVD++ .



Spark Examples

- Extract-transformation-load (ETL) operations
- Predictive analytics
- Machine learning
- Data access operation (SQL queries and visualizations)
- Text mining and text processing
- Real-time event processing
- Graph applications
- Pattern Recognition
- Recommendation engines
- And many more..

http://spark.apache.org/examples.html

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

Spark Installation

Prerequisite:

- Use python version 2.7.12.
- Install Anaconda
  - This includes ipython, jupyter
- Download JDK

```
brew update
brew tap caskroom/versions
brew cask search java
brew cask install java8
```

✓Spark runs on Java 8+, Python 2.7+/3.4+ and R 3.1+. For the Scala API, Spark 2.2.0 uses Scala 2.11. You will need to use a compatible Scala version (2.11.x).

https://gist.github.com/ololobus/4c221a0891775eaa86b0

Spark Installation

Install Scala and Spark

```
brew install scala
brew install apache-spark
```

```
ML-ITS-603436:2017_MSAN694 dwoodbridge$ brew info apache-spark
apache-spark: stable 2.2.0, HEAD
Engine for large-scale data processing
https://spark.apache.org/
/usr/local/Cellar/apache-spark/2.2.0 (1,318 files, 221.5MB) *
  Built from source on 2017-09-13 at 12:22:54
From: https://github.com/Homebrew/homebrew-core/blob/master/Formula/apache-spark.rb
```

https://gist.github.com/ololobus/4c221a0891775eaa86b0

Spark Installation

Spark-shell – uses scala or python.

- A program written in the shell is discarded after you exit the shell.

➔ Testing and developing applications.

`spark-shell`

`pyspark`

https://gist.github.com/ololobus/4c221a0891775eaa86b0

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

Spark Installation

Interactive Spark-shell – uses scala or python.

`spark-shell`

```
ML-ITS-603436:2017_MSAN694 dwoodbridge$ spark-shell
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
17/09/13 12:30:36 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
17/09/13 12:30:41 WARN ObjectStore: Failed to get database global_temp, returning NoSuchObjectException
Spark context Web UI available at http://192.168.1.5:4040
Spark context available as 'sc' (master = local[*], app id = local-1505331037170).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.2.0
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_144)
Type in expressions to have them evaluated.
Type :help for more information.

scala> sc
res0: org.apache.spark.SparkContext = org.apache.spark.SparkContext@3e984100
```

https://gist.github.com/ololobus/4c221a0891775eaa86b0

Spark Installation

Interactive Spark-shell – uses scala or python.

pyspark

```
ML-ITS-603436:~ dwoodbridge$ pyspark
Python 2.7.12 |Anaconda 4.1.1 (x86_64)| (default, Jul  2 2016, 17:43:17)
[GCC 4.2.1 (Based on Apple Inc. build 5658) (LLVM build 2336.11.00)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
17/09/13 12:46:04 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java clas
ses where applicable
17/09/13 12:46:09 WARN ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not
enabled so recording the schema version 1.2.0
17/09/13 12:46:09 WARN ObjectStore: Failed to get database default, returning NoSuchObjectException
17/09/13 12:46:09 WARN ObjectStore: Failed to get database global_temp, returning NoSuchObjectException
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 2.2.0
      /_/

Using Python version 2.7.12 (default, Jul  2 2016 17:43:17)
SparkSession available as 'spark'.
>>> sc
<SparkContext master=local[*] appName=PySparkShell>
```

https://gist.github.com/ololobus/4c221a0891775eaa86b0

Spark Installation

Jupyter set up (include in your ~/.profile or .bash_profile)

```
export PYSPARK_DRIVER_PYTHON=jupyter
export PYSPARK_DRIVER_PYTHON_OPTS="notebook"
```

Start pyspark (Python version spark shell)

```
pyspark
```

When jupyter notebook starts, choose “new python”

Week2

Spark Context

Spark Context

- Application instance representing the connection to the spark master and workers.
- Instantiated at the beginning of a Spark application and created by spark driver.
- Once having a SparkContext, you can use it to build RDDs.

```
In [1]: sc

Out[1]: SparkContext

        Spark UI
        Version
        v2.2.0
        Master
        local[*]
        AppName
        PySparkShell
```

Contents

Class Overview

Motivation - Why Distributed Computing?

What is Distributed Computing?

Spark

RDD Creation

What is RDD?

Resilient Distributed Dataset (RDD)

- Datasets that consists of records.
- Key Properties
  1. Distributed
     - The data in RDDs is divided into one or many partitions and distributed as in-memory collections of objects across worker nodes.
  2. Immutable
     - Read-only. : Once created, RDDs never change.
  3. Resilient
     - Automatically rebuilt on failure.
     - RDDs track lineage info to rebuild lost data. (Instead of replication.)

DATA INSTITUTE
UNIVERSITY OF SAN FRANCISCO

RDD Creation

Two ways of creating RDDs.

- Loading an external data.

```
lines = sc.textFile(“README.md”)
```

- Takes a collection such as **`Seq`** (**`Array`** or **`List`**) and creates RDD from its element and distribute to Spark executors in the process.

```
lines = sc.parallelize(["spark",
"spark is fun!"])
```

- Check the number of partitions.

```
lines.getNumPartitions()
```

https://spark.apache.org/docs/1.2.0/configuration.html

References

Distributed Computing with Spark, Reza Zadeh, http://stanford.edu/~rezab/slides/bayacm_spark.pdf

Spark Online Documentation : http://spark.apache.org/docs/latest/

Karau, Holden, et al. Learning spark: lightning-fast big data analysis. O'Reilly Media, Inc., 2015.

Zecevic, Petar, et al. Spark in Action, Manning, 2016.