

Apache Spark



Encadré par:

Alexandre DENIS

Réalisé par:

Guillaume Douezan-Grard

Zakaria HILI

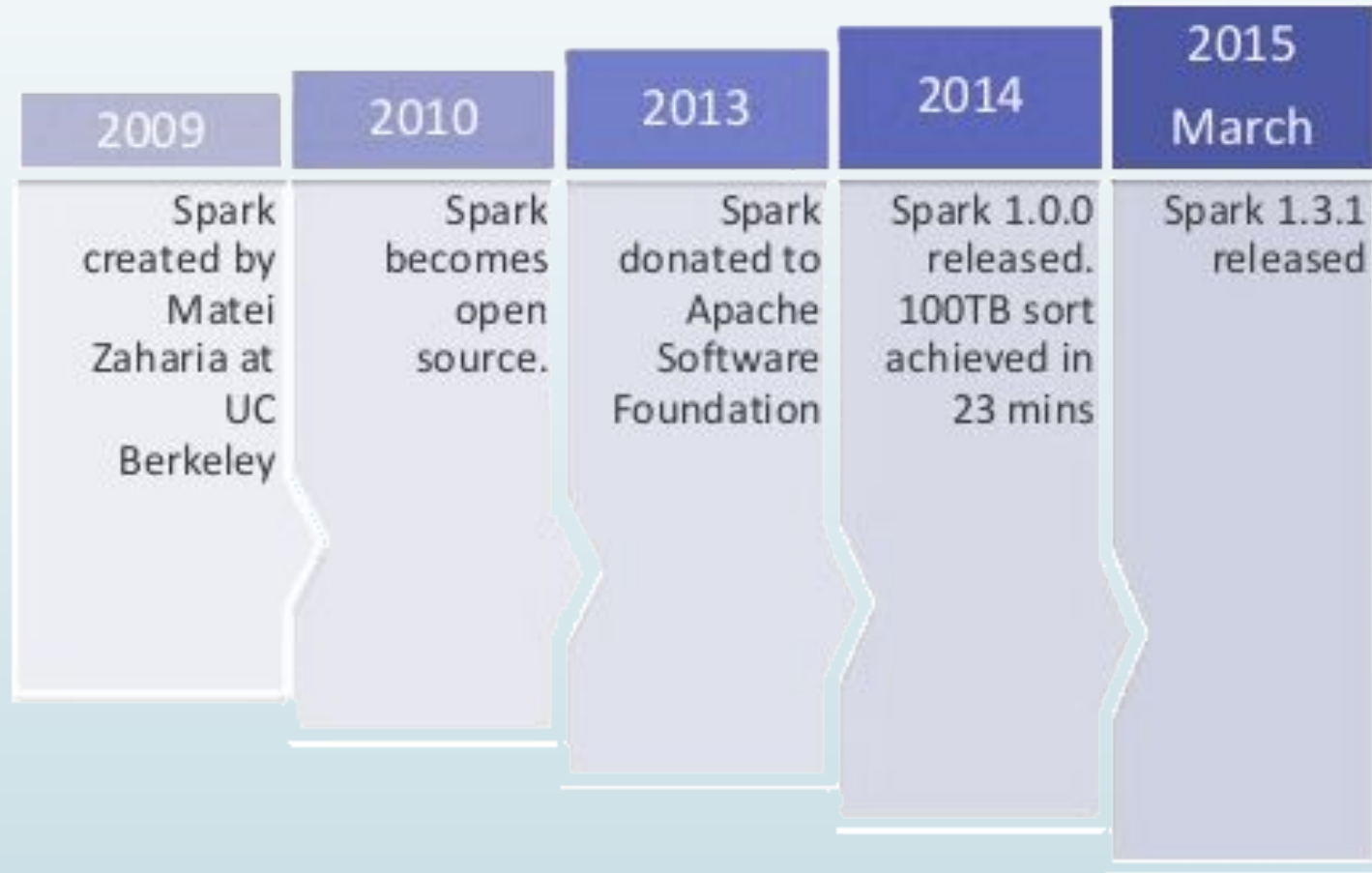
Reda Rahal Sabir

Introduction

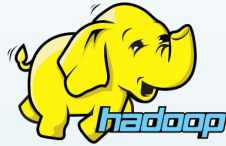
- Framework de BigData, puissant et rapide
- Utilise Mémoire RAM
- Plus rapide que les autres alternatives (ex: Hadoop MapReduce)
- Sous licence Apache
- Support les langages : Java, Python, R, Scala.
- Compatible avec des Bases de données NoSQL (HBase, Cassandra, MapR-DB, MongoDB and Amazon's S3).



Historique



Hadoop Vs Spark



Hadoop	Spark
Difficile à programmer	Plus Simple à programmer

Hadoop vs Spark

- **Hadoop MapReduce**

```
public static class WordCountMapClass 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 itr = new StringTokenizer(line);
        while (itr.hasMoreTokens()) {
            word.set(itr.nextToken());
            output.collect(word, one);
        }
    }
}

public static class WordCountReduce extends MapReduceBase
    implements Reducer<Text, IntWritable, Text, IntWritable> {

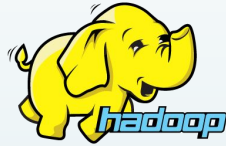
    public void reduce(Text key, Iterator<IntWritable> values,
        OutputCollector<Text, IntWritable> output,
        Reporter reporter) throws IOException {

        int sum = 0;
        while (values.hasNext()) {
            sum += values.next().get();
        }
        output.collect(key, new IntWritable(sum));
    }
}
```

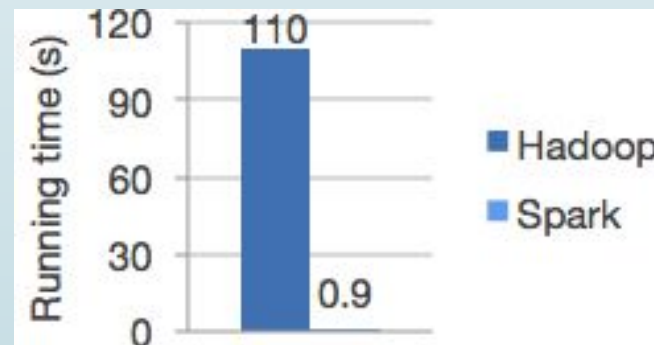
- **Spark**

```
val spark = new SparkContext(master, appName, [sparkHome], [jars])
val file = spark.textFile("hdfs://...")
val counts = file.flatMap(line => line.split(" "))
                    .map(word => (word, 1))
                    .reduceByKey(_ + _)
counts.saveAsTextFile("hdfs://...")
```

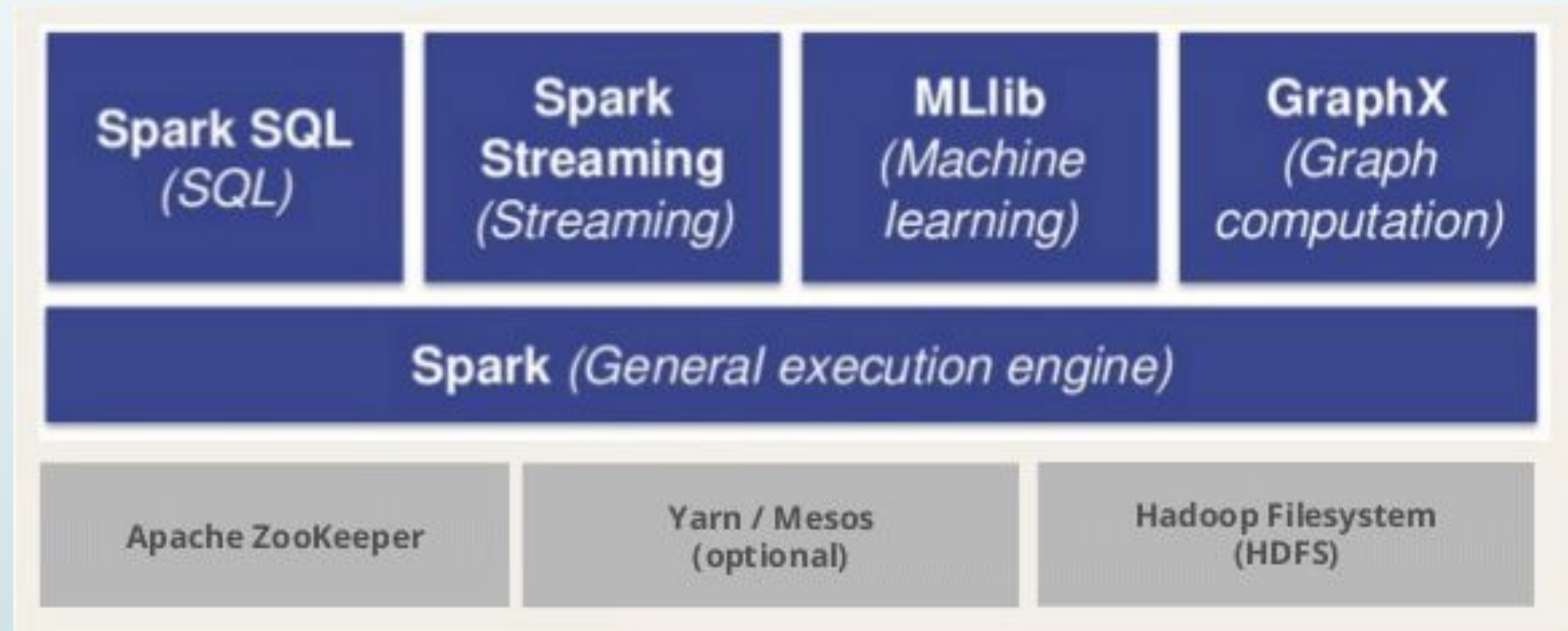
Hadoop Vs Spark



Hadoop	Spark
Difficile à programmé	Plus Simple à programmé
On-Disk, Batch	On-Disk, Batch, In-memory, streaming
Pas de librairie, outils séparés	Spark Core, Spark Streaming, Spark SQL, MLlib, GraphX
Java, Python, Scala, R	Java, Python, Scala, R

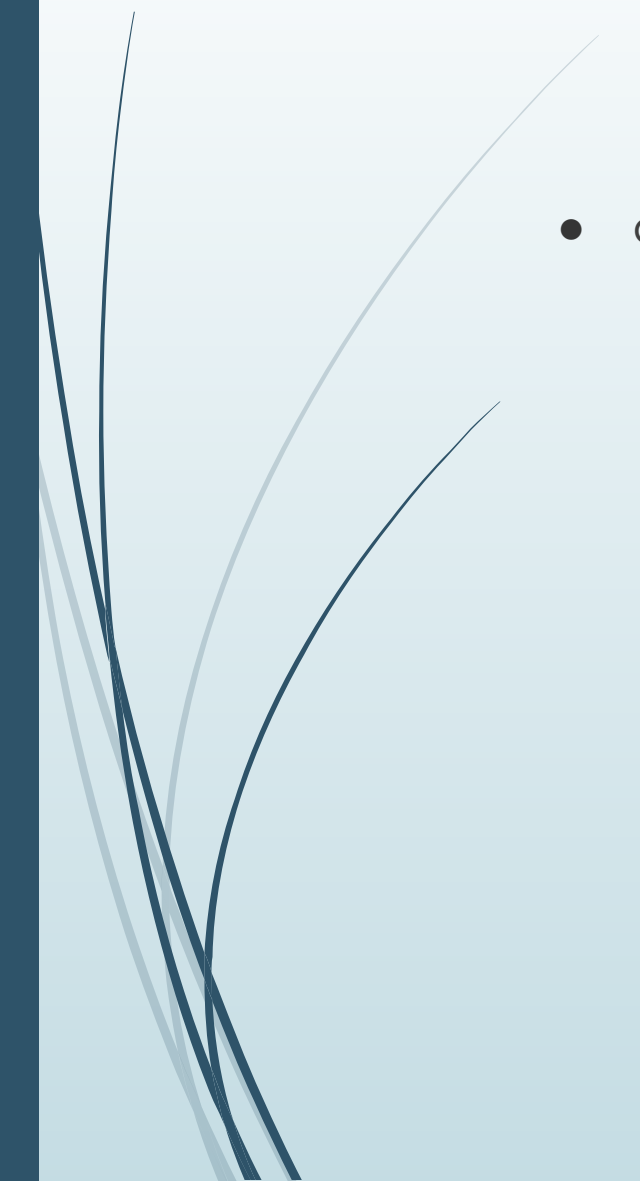


Architecture





Demo

- à nous de jouer
- 



Conclusion

- Spark est une solution pour traiter les données rapidement
 - Repose principalement sur l'utilisation de la mémoire
- 