Big Data Processing : Using Spark

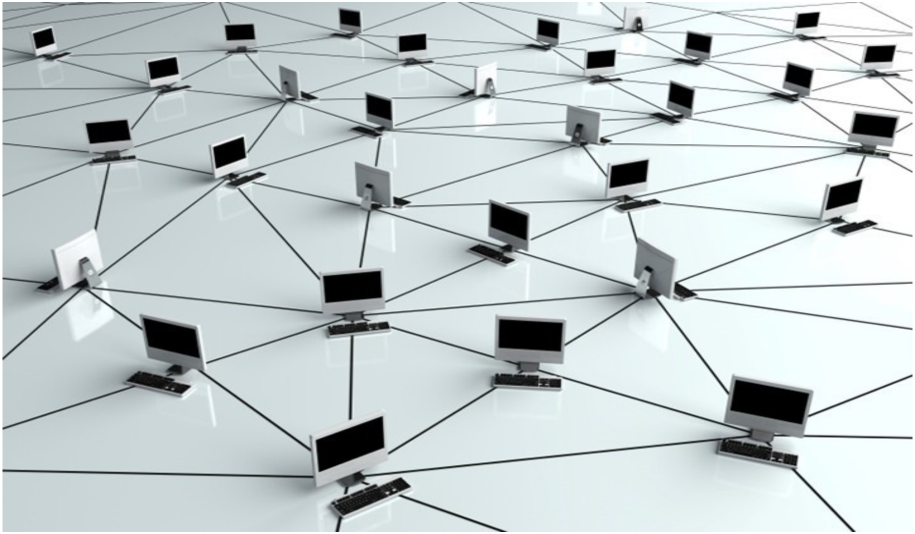
Mostafa Alaa Mohamed
Senior Big Data Engineer
Email: mustafaalaa.mohamed@gmail.com
Linkedin: Mostafa Alaa

Big Data & Analytics Department, Etisalat UAE

February 5, 2017

Table of Content:

Introduction To Spark



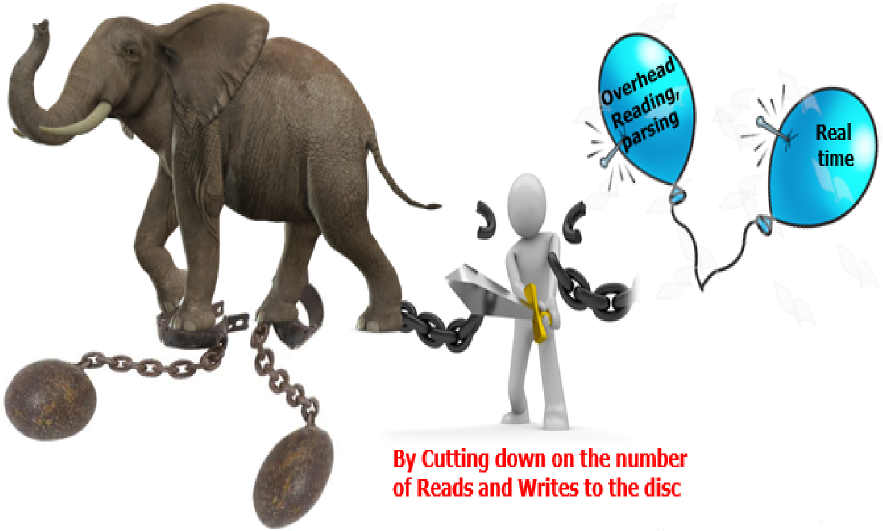
Introduction To Spark

- Any Big Data solution working based distributed systems.
- What is distributed systems in brief?
  - Components interact with each other in order to achieve a common goal.
  - Problem definition is to run a process and distribute it into Multi node.
  - Every system has its own way to manage the cluster (distributed system). Managing the cluster including
    - How the data will be processed.
    - How the data will be shuffling (transferring) between the nodes .
    - Keep track of the current processing tasks.
    - Handle the logs and long running tasks.

Introduction To Spark

- Apache Spark is Fast and general cluster computing engine that extends Google's MapReduce model
- Write programs in terms of parallel transformations on distributed datasets.
- The power of supporting research in the Labs gained Spark. How? *Apache Spark is an open source big data processing framework built around speed, ease of use, and sophisticated analytics. It was originally developed in 2009 in UC Berkeley's* **AMPLab**, *and open sourced in 2010 as an Apache project.*

Introduction To Spark



By Cutting down on the number of Reads and Writes to the disc

Introduction To Spark

- Improves efficiency through: In-memory data sharing, General computation graphs(100x faster).
- Improves usability through: Rich APIs in Java, Scala, Python, Interactive shell(2-5x less code)

Example Mapreduce on Hadoop

Scala Code 1: Java Mapreduce Average Example

```
private  IntWritable  one = new IntWritable(1);
private  IntWritable  output = new IntWritable()
proctected  void  map(LongWritable key,Text value,  Context context)
{String []   fields   = value. split ("\t");
output.set( Integer . parseInt ( fields  [1]) );
context . write(one,  output);}
IntWritable  one = new IntWritable(1);
DoubleWritable average  = new DoubleWritable();
protected  void  reduce( IntWritable  key, Iterable <IntWritable>
     values,Context context)  {
int  sum  = 0 ; int  count  = 0;
for( IntWritable  value  :  values)  {
sum += value.get(); count++;}
average. set(sum  /  (double)  count);
context . Write(key,  average);}
```

Example Spark on Hadoop

Scala Code 2: Python Spark Average Example

```
data = sc.textFile(...).split(”\t”)
data.map(lambda x: (x[0], [x.[1], 1])) \
.reduceByKey(lambda x, y: [x[0] + y[0], x[1] +
.map(lambda x: [x[0], x[1][0] / x[1][1]]) \
.collect()
```

Installing Apache Spark and Scala

please refer to this article and videos below

- Github Document.
- Youtube Video.