Apache Spark - PySpark

Trần Hữu Thúy

Ngày 27 tháng 2 năm 2020

Nội dung

Tổng quan về Apache Spark

- Opensource, framework, công nghệ tính toán trên cluster, giúp cho việc tính toán được thực hiện một cách nhanh chóng (lightning-fast cluster).
- Một trong các dự án được phát triển bởi Hadoop năm 2009 ở UC Berkeley's AMPLab, trở thành top level project của Apache từ năm 2014.
- nhanh gấp 10 đên 100 lần Hadoop, đặc điểm chính là in-memory cluster computing.
- Không phải là phiên bản chỉnh sửa của Hadoop, Hadoop chỉ là một cách thực thi nó.

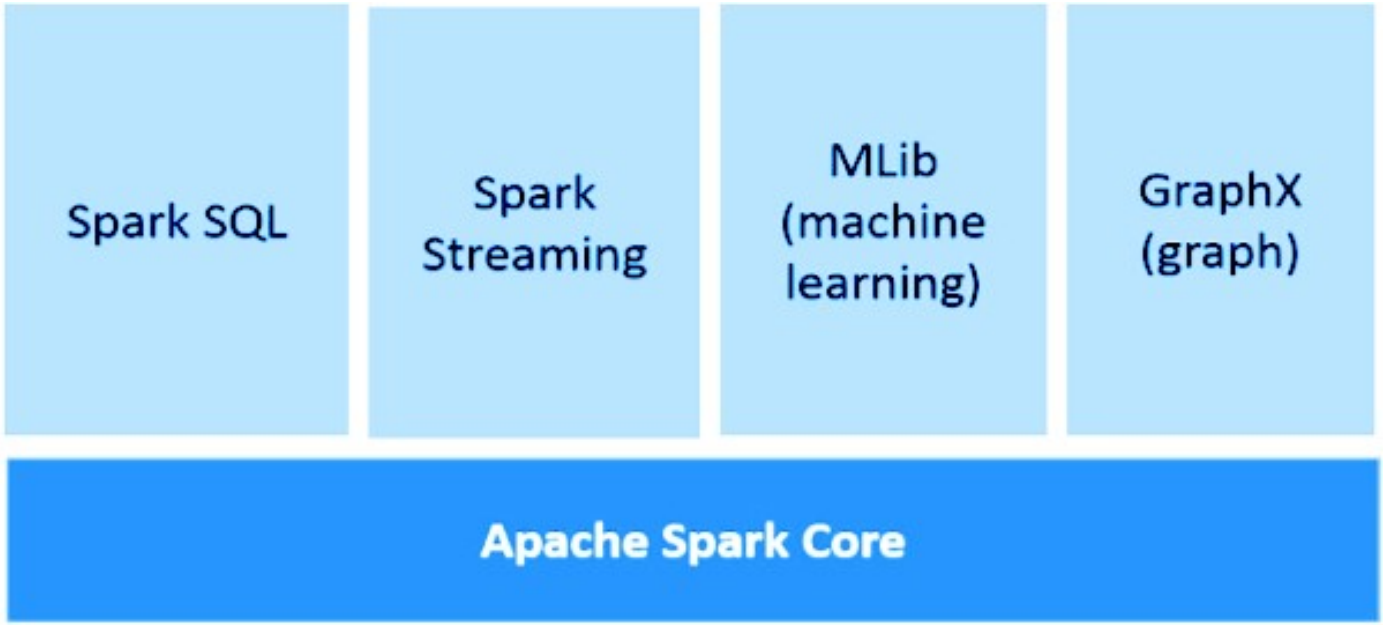
Tổng quan về Apache Spark

Các công ty sử dụng Apache Spark

Đặc điểm của Apache Spark

- Speed : Spark giúp tốc độ chạy một ứng dụng trên Hadoop cluster, tăng 100 lần trên memory, 10 lân trên ổ đĩa
- Supports multiple languages : Cung cấp nhiều APIs ở nhiều ngôn ngữ khác nhau như Java, Scale, Python, do đó có thể dễ dàng tạo ứng dụng trên nhiều ngôn ngữ.
- Advanced Analytics : Hỗ trợ SQL queries, Streaming data, Machine learning (ML) và Graph algorithms.

Components của Apache Spark



Components của Apache Spark

- Apache Spark Core: cung cấp các chức năng cơ bản nhất Spark như lập lịch, quản lý bộ nhớ, tương tác bộ nhớ,... Đặc biệt cung cấp API để định nghĩa RDD (Resilient Distributed DataSet), kiểu dữ liệu phân tán của Spark.
- Spark SQL : cho phép truy vấn dữ liệu có cấu trúc thông qua SQL.
- Spark Streaming : cung cấp API xử lý dữ liệu stream.
- MLlib (Machine Learning Libary) : cung cấp các thuật toán của học máy.
- GraphX : thư viện xử lý đồ thị.

Cài đặt Apache Spark trên Ubuntu

Cài đặt Java

sudo apt-get install default-jdk

Mở file /.bashrc

gedit $\sim$ /*.bashrc*

Thêm 2 lệnh vào cuối file

export JAVA_HOME=/usr/lib/jvm/default-java
export PATH=*PATH* :JAVA_HOME/bin

Save và thực thi thay đổi file /.bashrc

source $\sim$ /*.bashrc*

Cài đặt Apache Spark trên Ubuntu

Cài đặt Scala

```
sudo apt-get install scala
```

Cài đặt Spark tại https://spark.apache.org/downloads.html, và giải nén nó . Giả sử file giải nén là spark-2.4 và tại thư mục Workspace/BigData/Tools

Thêm vào cuối file ∼ /.*bashrc*

```
export PATH=$PATH:Workspace/BigData/Tools/spark-2.4/bin
```

Kiểm tra Spark

```
spark-shell
val textRDD= sc.textFile("README.md").textRDD.count()
```

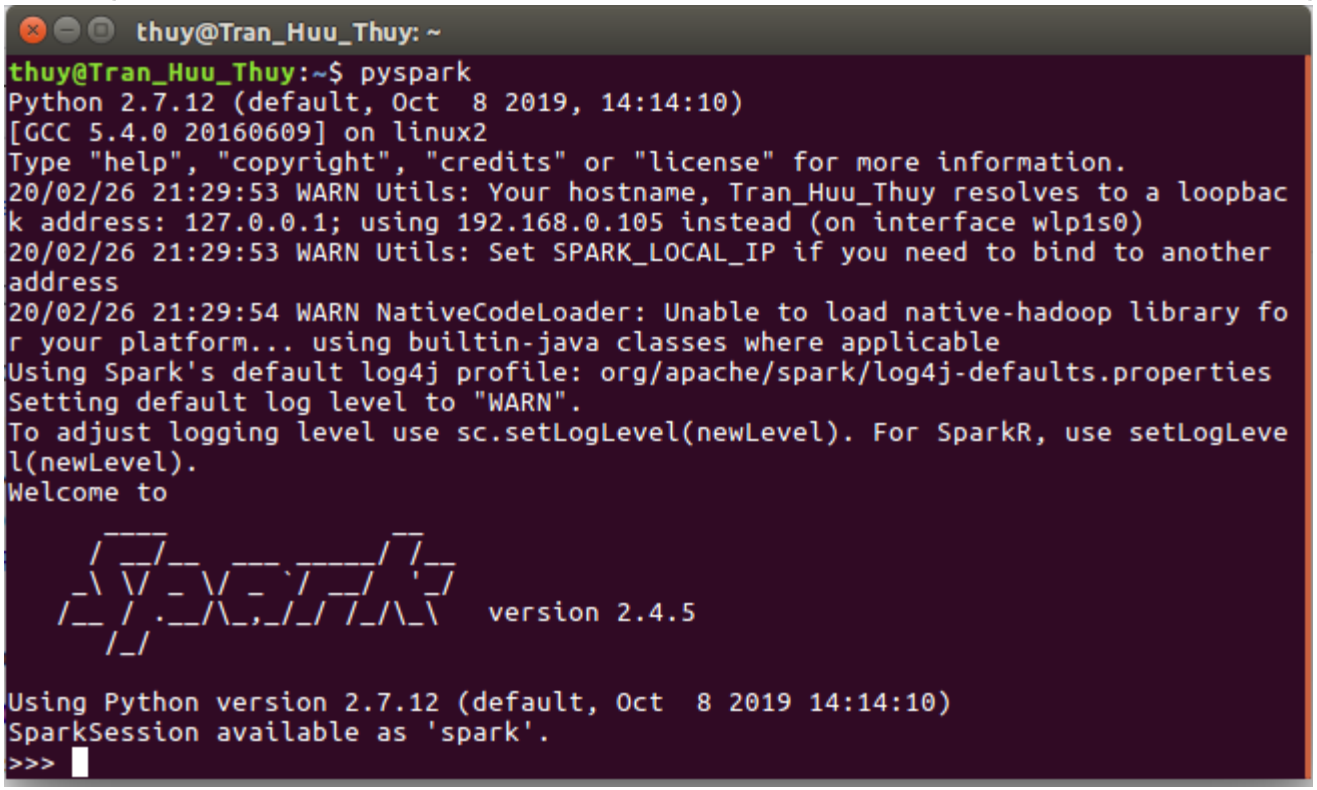
Cài đặt Apache Spark trên Ubuntu

Kết quả

```
thuy@Tran_Huu_Thuy: ~/Desktop/ApacheSpark/spark-2.4.5-bin-hadoop2.7
r your platform... using builtin-java classes where applicable
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 setLogLeve
l(newLevel).
Spark context Web UI available at http://192.168.0.105:4040
Spark context available as 'sc' (master = local[*], app id = local-1582727854650
).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.4.5
      /_/

Using Scala version 2.11.12 (OpenJDK 64-Bit Server VM, Java 1.8.0_242)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val textRDD = sc.textFile("README.md").count()
textRDD: Long = 104

scala>
```

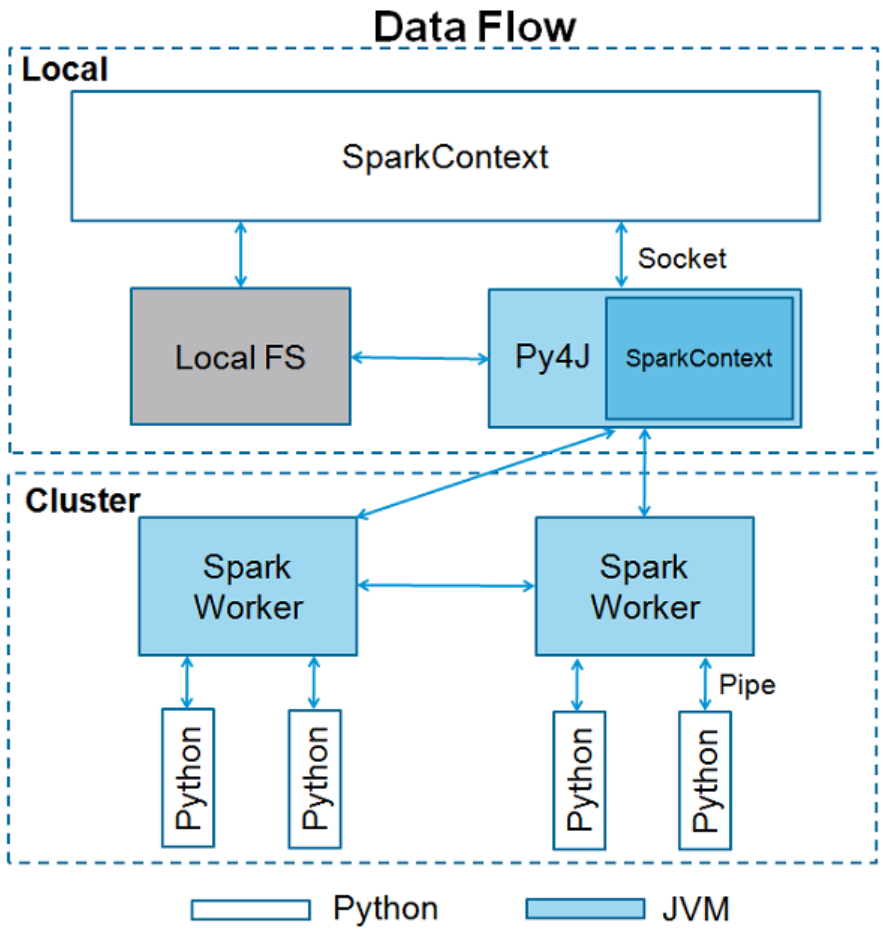
Pyspark

- spark-shell làm việc với ngôn ngữ scala
- Pyspark là Apache Spark với Python
  - Kiểm tra pyspark sau khi cài đặt Apache Spark với lệnh 'pyspark'



SparkContext

- Thực thể của môi trường xử lý dữ liệu, thiết lập các dịch vụ trung gian : master, appname, sparkhome,...
- Được khởi tạo ngay khi tạo bất kì một ứng dụng spark.
- Mặc định, spark-shell và pyspark có biến SparkContext là 'sc'.
- SparkContext tạo ra RDD, share variables,...
- Khi tạo ra SparkContext mới cần phải close SparkContext trước đó.



SparkContext trong Python

SparkConf và SparkContext

```
from pyspark import SparkContext, SparkConf
conf=SparkConf().setAppName(appName).setMaster(master)
sc = SparkContext(conf=conf)
```

- appName : tên ứng dụng trên cluster.
- master : URL của cluster mà nó kết nối tới.

SparkSession

- SparkContext chủ yếu xử lý với RDD.
- Khi làm việc với stream ta cần streamContext, hive cần HiveContext, sql cần SQLContext.
- SparkSesstion là sự kết hợp của chúng, ta có thể tạo ra SparkContext từ nó.

SparkSession

SparkSession trong python

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("Python Spark SQL basic
example")
.config("spark.some.config.option", "some-value").getOrCreate()
```

Tạo SparkContext

```
sc = spark.sparkContext
```

- Ngoài ra SparkSession còn tạo ra nhiều context khác.

Resilient Distributed Dataset

- Kiểu dữ liệu phân tán cơ bản của Apache Spark.
- Kiểu dữ liệu bất biến(imutable data), khi tạo ra ta không thể thay đổi nó.
- Có khả năng chịu lỗi tốt (fault tolerant).
- Được xử lý ngay trên memory.
- Có 2 cách để khởi tạo : parallelizing và existing collection.

Khởi tạo RDD bằng Parallelized Collections

Parallelized Collections

data = [1, 2, 3, 4, 5]
distData = sc.parallelize(data)

- RDD có thể được khởi tạo bằng cách gọi phương thức parallelize của SparkContext.Các phần tử sẽ được sao chép tới tập dữ liệu phân tán và được thực hiện song song.
- Phương thức parallelize còn có tham số thứ hai là số partition muốn chia dataset cho mỗi CPU của cluster.
- Cách khởi tạo này thường được sử dụng với dữ liệu nhỏ.

Khởi tạo RDD bằng External Datasets

- Pyspark có thể tạo ra RDD bằng bất kì nguồn storege nào được hỗ trợ bởi Hadoop như local file system, HDFS, Cassandra, HBase, Amazon,...
- Spark hỗ trợ text file, squencefile,...
- RDD được tạo ra từ phương thức textFile của SparkContext, bằng cách truyền cho nó URI.

Đọc file từ SparkContext

```
dataFile = sc.textFile('data.txt')
```

Khởi tạo RDD bằng External Datasets

Một số lưu ý

- Nếu sử dựng local filesystem, các worker node phải có thể kết nối được với file.Do đó, hoặc sao chép file đên các worker hoặc sử dụng network-mounted shared file system.
- URI trong các phương thức như textFile,... có thể là các thư mục, các file, hoặc các file chư giải nén,...
- Phương thức textFile cũng có tham số thứ 2 là số parttion chia dataset cho mỗi CPU trên cluster.

RDD Operations

Transformations/Actions

- RDD tranformations tạo ra một RDD mới từ nó.
- Tất cả tranformations trong Spark đều là lazy, nó không tính toán bất kì môt điều gì.Nó chỉ ghi nhớ sự biến đổi để tạo ra một vài kiều dữ liệu có bản (ví dụ file).
- Nó chỉ tính toán khi được một action gọi tới. Điều này làm cho Spark trở nên hoạt động hiệu quả.
- RDD Actions trả về giá trị cho driver program sau khi thực hiện một công việc tính toán nào đó trên RDD ban đầu.
- RDD Actions không trả về RDD.

- Pyspark hỗ trợ method transformations/Actions, truy cập website https://spark.apache.org/docs/latest/rdd-programming-guide.html để xem chi tiết.

RDD Persistence

- Khi một RDD được tạo ra từ nhiều quá trình transformation, mất thời gian truy xuất !
- Làm thế để có thể tối ưu vấn đề này ?

RDD Persistence

- Khi perisist một RDD, mỗi node sẽ lưu trữ tất cả partition của nó vào bộ nhớ, giúp các action sau này làm việc nhanh hơn.
- Hai phương thức persist() và cache() được sử dụng.Spark's cache là khả năng chịu lỗi mỗi khi RDD mất đi patition nào đó của nó.
- Persist cho phép thiết lập storage level như MEMORY_ONLY, MEMORY_AND_DISK, DISK_ONLY,...
- cache là persist ở storage level như MEMORY_ONLY.