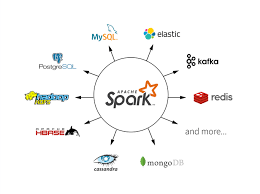
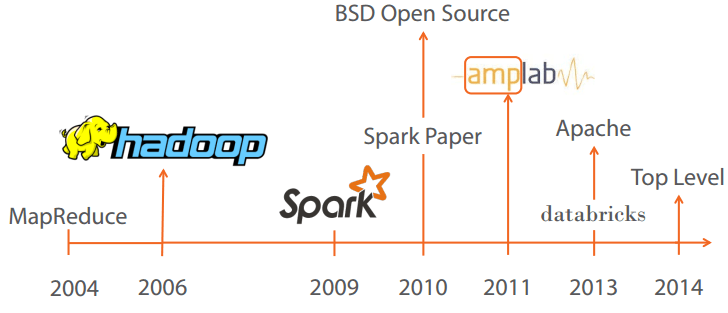
Apache Spark

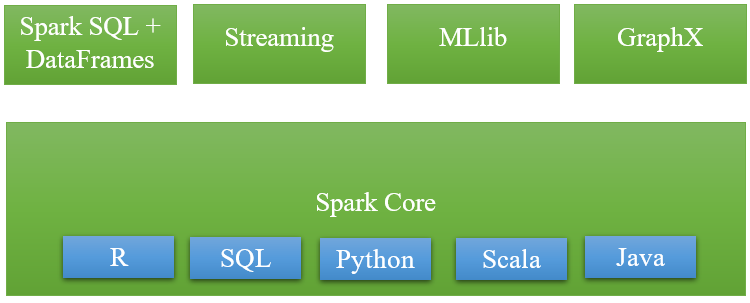


Giới thiệu



Apache Spark là một framework mã nguồn mở tính toán cụm, được phát triển sơ khởi vào năm 2009 bởi AMPLab.   
Tốc độ xử lý được tính toán dựa theo việc thực hiện cùng lúc trên nhiều máy khác nhau.  
Đồng thời việc tính toán được thực hiện ở bộ nhớ trong hoặc trên RAM.  
Spark cho phép xử lý dữ liệu theo thời gian thực, vừa nhận dữ liệu từ các nguồn khác nhau đồng thời thực hiện ngay việc xử lý trên dữ liệu vừa nhận được.

**Thành phần**



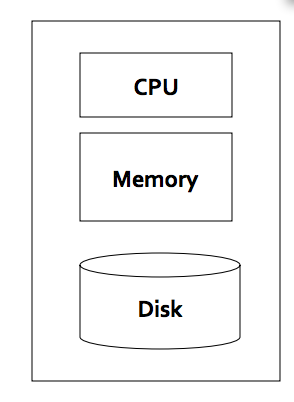
Apache Spark gồm có 5 thành phần chính : Spark Core, Spark Streaming, Spark SQL, MLlib và GraphX, trong đó:

* Spark Core là nền tảng cho các thành phần còn lại và các thành phần này muốn khởi chạy được thì đều phải thông qua Spark Core do Spark Core đảm nhận vai trò thực hiện công việc tính toán và xử lý trong bộ nhớ (In-memory computing) đồng thời nó cũng tham chiếu các dữ liệu được lưu trữ tại các hệ thống lưu trữ bên ngoài.
* Spark SQL cung cấp một kiểu data abstraction mới (SchemaRDD) nhằm hỗ trợ cho cả kiểu dữ liệu có cấu trúc (structured data) và dữ liệu nửa cấu trúc (semi-structured data – thường là dữ liệu dữ liệu có cấu trúc nhưng không đồng nhất và cấu trúc của dữ liệu phụ thuộc vào chính nội dung của dữ liệu ấy). Spark SQL hỗ trợ DSL (Domain-specific language) để thực hiện các thao tác trên DataFrames bằng ngôn ngữ Scala, Java hoặc Python và nó cũng hỗ trợ cả ngôn ngữ SQL với giao diện command-line và ODBC/JDBC server.
* Spark Streaming được sử dụng để thực hiện việc phân tích stream bằng việc coi stream là các mini-batches và thực hiệc kỹ thuật RDD transformation đối với các dữ liệu mini-batches này. Qua đó cho phép các đoạn code được viết cho xử lý batch có thể được tận dụng lại vào trong việc xử lý stream, làm cho việc phát triển lambda architecture được dễ dàng hơn. Tuy nhiên điều này lại tạo ra độ trễ trong xử lý dữ liệu (độ trễ chính bằng mini-batch duration) và do đó nhiều chuyên gia cho rằng Spark Streaming không thực sự là công cụ xử lý streaming giống như Storm hoặc Flink.
* MLlib (Machine Learning Library): MLlib là một nền tảng học máy phân tán bên trên Spark do kiến trúc phân tán dựa trên bộ nhớ. Theo các so sánh benchmark Spark MLlib nhanh hơn 9 lần so với phiên bản chạy trên Hadoop (Apache Mahout).
* GrapX: Grapx là nền tảng xử lý đồ thị dựa trên Spark. Nó cung cấp các Api để diễn tảcác tính toán trong đồ thị bằng cách sử dụng Pregel Api.

# Map Reduce

## Vấn đề với năng lực xử lý của một máy tính

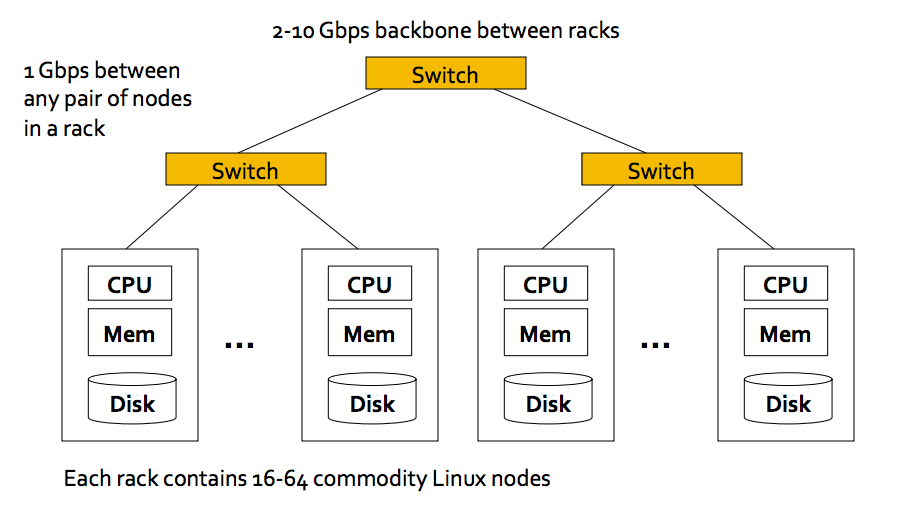
Máy tính mà chúng ta vẫn sử dụng để xử lý dữ liệu hàng ngày có thể được mô tả một cách trừu tượng hoá nhất bằng hình vẽ dưới đây:



Theo như hình vẽ chúng ta có: CPU, Memory và đĩa cứng. Theo công nghệ hiện tại mình biết thì mỗi module có các thông số tối đa như sau:  
 - CPU: với máy tính thông thường thì trên 1 board chỉ có 1 đến 2 CPUs. Mỗi CPU có thể có tối đa 18 cores  
 - Memory: trên 1 mainboard có thể cắm tối đa 16 thanh RAM. Mỗi thanh có dung lượng tối từ 8G. Một máy tính có thể có khoảng 8\* 16 = 128GB RAM)  
 - Đĩa cứng: dung lượng tối đa của HDD là 8TB  
Với máy tính với cấu hình dù rất khoẻ như ở trên thì ta vẫn thấy giới hạn xử lý dữ liệu của nó. Cụ thể máy tính không thể lưu trữ được quá 8TB dữ liệu, không thể đồng thời xử lý được dữ liệu lớn hơn 128GB (kích thước của RAM) và không thể đồng thời xử lý được lớn hơn 18 luồng chương trình.

## Giải quyết bài toán một máy tính thế nào?

Như đã tính toán thử ở trên, một máy tính không thể nào xử lý được số lượng dữ liệu lớn như vậy. Điều này dẫn đến 2 đòi hỏi:  
 - Đấu nối nhiều máy tính để cùng xử lý  
 - Tìm ra cách để cho các máy tính cùng xử lý

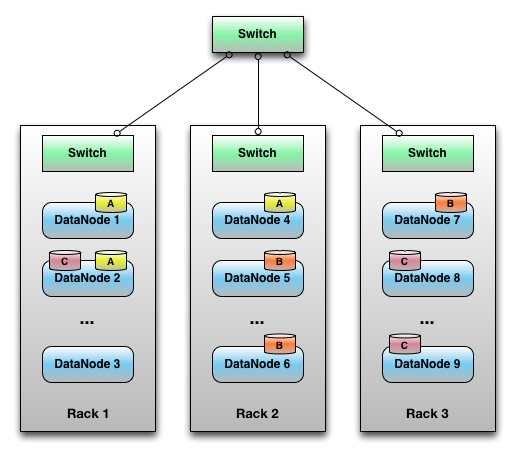


## Cách để cho các máy tính cùng xử lý (Map-Reduce)

Phân chia dữ liệu thành nhiều block và chia cho nhiều máy tính lưu trữ (đảm bảo tính toàn vẹn và tính sẵn sàng của dữ liệu).  
Chuyển tính toán về nơi có dữ liệu.  
Đưa ra mô hình và giao diện tính toán đơn giản.

### Chia dữ liệu thành nhiều Block

GFS là một hệ thống quản lý File phân tán với các chức năng giống như hệ thống File bình thường của Linux như: không gian tên (namespace), tính thừa thãi (redundancy), và tính sẵn sàng (availability).  
Đặc điểm của hệ thống File phân tán này là:  
 - Dùng để quản lý các File có kích thước lớn: kích thước từ trăm GB đến TB  
 - Dữ liệu ít khi bị cập nhật ở giữa file (kiểu mở file, đến dữa file, cập nhật) mà thường được đọc học ghi vào cuối File (append).  
GFS được thiết kế bao gốm 2 module: Chunk Server và Master Server.  
 - Chunk Server làm nhiệu vụ lưu trữ các chunk (hay block) dữ liệu. Mỗi chunk có thể của 1 File hay của file khác. Các Chuck dữ liệu được sao chép giữa các máy chủ khác nhau.  
 - Master Server làm nhiệm vụ quản lý metadata cho hệ thống File. Cụ thể Master Server sẽ lưu giữ thông tin như File này có bao nhiêu Chunk và mỗi Chunk sẽ được lưu ở máy chủ nào. Dựa vào thông tin metadata này mà thuật toán tính toán sẽ chuyển tính toán về máy chủ có chunk (sẽ trình bày kỹ hơn)



### Mô hình tính toán

Ví dụ bài toán đếm từ:  
Map Reduce chia làm 3 thao tác:   
 - Map: quét file đầu vào và ghi lại từng bản ghi  
 - Group by Key: sắp xếp và trộn dữ liệu cho mỗi bản ghi sinh ra từ  
 - Reduce: tổng hợp, thay đổi hay lọc dữ liệu từ thao tác trước và ghi kết quả ra File.  
Hình minh hoạ sau làm rõ các bước và nội dung thực hiện ở mỗi bước:



### Tham khảo:

* <https://viblo.asia/p/tim-hieu-ve-apache-spark-ByEZkQQW5Q0>
* <https://kipalog.com/posts/Map-Reduce>