SPARK và MAPREDUCE

Nguyễn Văn Đạt – 51800364
24/01/2021

1. Spark

Spark là một hệ thống quy trình làm việc (workflow system). Spark là một tiến bộ với các hệ thống quy trình làm việc trước đây, có thể kể đến các vai trò sau:

- Một cách hiệu quả hơn để đối phó với thất bại (coping with failures).
- Một cách hiệu quả hơn để nhóm các nhiệm vụ giữa các nút tính toán và lập lịch (scheduling) thực hiện các chức năng.
- Tích hợp các tính năng của ngôn ngữ lập trình như vòng lặp (về mặt kỹ thuật đưa nó ra khỏi lớp quy trình làm việc tuần hoàn của hệ thống) và các thư viện.

Dữ liệu trung tâm của Spark được gọi là Resilient Distributed Dataset (Tập dữ liệu phân tán có khả năng phục hồi), viết tắt là RDD. RDD là tập hợp các phần tử được phân vùng trên các nút của cụm có thể hoạt động song song. RDD được tạo bằng cách bắt đầu bằng một tệp trong hệ thống tệp Hadoop (hoặc bất kỳ hệ thống tệp nào khác được Hadoop hỗ trợ) hoặc một bộ sưu tập Scala hiện có trong chương trình trình điều khiển và chuyển đổi nó. Người dùng cũng có thể yêu cầu Spark duy trì một RDD trong bộ nhớ, cho phép nó được sử dụng lại một cách hiệu quả trong các hoạt động song song. Cuối cùng, các RDD tự động phục hồi sau các lỗi của nút.

Một chương trình Spark bao gồm một chuỗi các bước, mỗi bước thường áp dụng một số chức năng cho một RDD để tạo ra một RDD khác. Các phép toán như vậy được gọi là phép biến hình (transformations). Spark hỗ trợ lấy dữ liệu từ hệ thống tệp xung quanh (chả hạn HDFS, Cassandra, …).

1.1. Map, Flatmap, và Filter:

Đây là các thao tác thường được sử dụng.

1.1.1. Map, Flatmap

Phép biến đổi Map nhận một tham số là một hàm và nó áp dụng hàm đó cho mọi phần tử của một RDD, tạo ra một RDD khác. Nhưng nó khác hoàn toàn với MapReduce (phần sau). Trong Spark, một hàm Map có thể áp dụng cho bất kỳ loại đối tượng nào, nhưng kết quả là nó tạo ra chính xác một đối tượng. Kiểu của đối tượng kết quả có thể là một tập hợp, nhưng điều đó không giống như việc tạo ra nhiều đối tượng từ một đối tượng đầu vào. Nếu bạn muốn tạo một tập hợp các đối tượng từ một đối tượng duy nhất, Spark cung cấp cho bạn một phép chuyển đổi khác có tên là Flatmap, tương tự như Map of MapReduce, nhưng không có yêu cầu tất cả các loại phải là cặp khóa-giá trị (key-value pairs).

Ví dụ 1: Chúng ta có thể viết một hàm Spark Map lấy một tài liệu và tạo ra một tập hợp các cặp, với mỗi cặp có dạng (w, 1), trong đó w là một trong các từ trong tài liệu.

```
>>> conf = SparkConf().setMaster('local').setAppName('Word counting')
sc = SparkContext.getOrCreate(conf = conf)
text = "to be or not to be".split()
```

```
rdd = sc.parallelize(text)
counts = rdd.map(lambda x:(x,1))
print(counts.collect())
>>> [('to', 1), ('be', 1), ('or', 1), ('not', 1), ('to', 1), ('be', 1)]
```

1.1.2. Filter

Spark cũng cung cấp một hoạt động tương tự như một dạng Map giới hạn, được gọi là Filter. Thay vì một chức năng như một tham số, phép biến đổi Filter lấy một vị từ (predicate) áp dụng cho loại đối tượng trong RDD đầu vào. Vị từ trả về true hoặc false cho mỗi đối tượng và RDD đầu ra của một phép biến đổi Bộ lọc chỉ bao gồm các đối tượng đó trong RDD đầu vào mà hàm lọc trả về true.

Ví dụ 2: Tìm ra phần tử chẵn trong input

```
>>> a=[1,3,5,2]
binhphuong = list(filter(lambda x: (x%2==0),a))
print(binhphuong)
>>> [2]
```

1.1.3. Reduce

Trong Spark, hoạt động Reduce là một hành động, không phải là một phép biến đổi. Đó là, hoạt động Reduce áp dụng cho một RDD nhưng trả về một giá trị chứ không phải RDD khác.

Ví dụ 3: Nhóm các chữ giống nhau trong RDD ở ví dụ 1

```
>>> red = counts.reduceByKey(lambda x,y: x+y)
print(red.collect())
>>> [('to', 2), ('be', 2), ('or', 1), ('not', 1)]
```

1.2. Các thành phần của Spark

Thành phần cốt lõi của Spark là Spark Core: cung cấp những chức năng cơ bản nhất của Spark như lập lịch cho các tác vụ, quản lý bộ nhớ, fault recovery, tương tác với các hệ thống lưu trữ… Spark có thể chạy trên nhiều loại Cluster Managers như Hadoop YARN, Apache Mesos hoặc trên chính cluster manager được cung cấp bởi Spark được gọi là Standalone Scheduler.

Spark SQL là một thành phần nằm trên Spark Core cho phép truy vấn dữ liệu cấu trúc qua các câu lệnh SQL. Spark SQL có thể thao tác với nhiều nguồn dữ liệu như Hive tables, Parquet, và JSON…

Spark Streaming cung cấp API để dễ dàng xử lý dữ liệu stream, cho phép thực hiện phân tích xử lý trực tuyến xử lý theo lô.

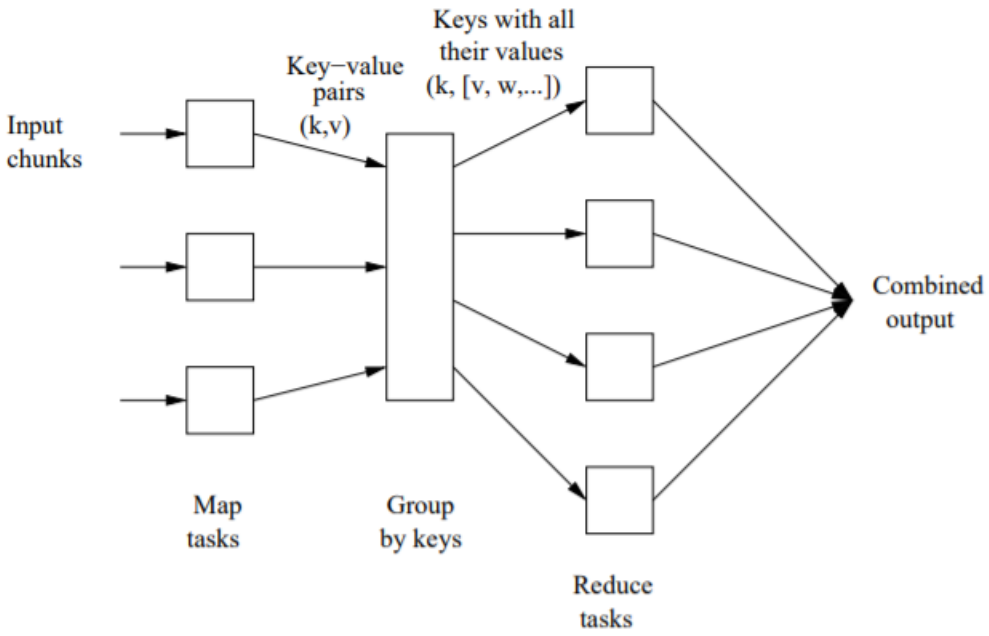
GraphX là thư viện để xử lý đồ thị. Nó cung cấp các API để diễn tả các tính toán trong đồ thị bằng cách sử dụng Pregel Api.

MLlib (Machine Learning Library) là một nền tảng học máy phân tán bên trên Spark do kiến trúc phân tán dựa trên bộ nhớ. Nó cung cấp rất nhiều thuật toán của học máy như: classification, regression, clustering, collaborative filtering…

2. MapReduce:

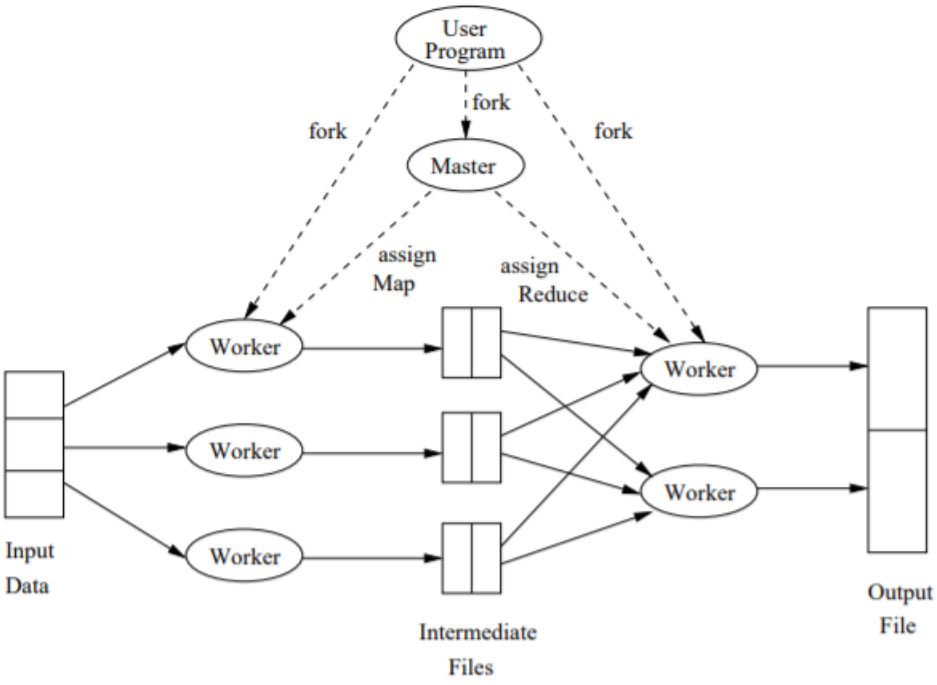
MapReduce là một kiểu tính toán đã được triển khai trong một số hệ thống, bao gồm triển khai nội bộ của Google (gọi đơn giản là MapReduce) và triển khai mã nguồn mở Hadoop phổ biến có thể lấy được, cùng với hệ thống tệp HDFS từ Apache Foundation. Triển khai sử dụng MapReduce để quản lý nhiều phép tính quy mô lớn, song song theo cách có thể chịu được các lỗi phần cứng.

Sơ đồ mô tả MapReduce:



- The Map Tasks
- Grouping by Key
- The Reduce Tasks
- Combiners

Sơ đồ tổng quan hóa MapReduce hoạt động:



2.1. Map và Reduce:

MapReduce có 2 hàm chính là Map() và Reduce(), đây là 2 hàm đã được định nghĩa bởi người dùng (người dùng phải viết) và nó cũng chính là 2 giai đoạn liên tiếp trong quá trình xử lý dữ liệu của MapReduce.

Function Map(): có nhiệm vụ nhận Input cho các cặp giá trị/khóa và output chính là tập những cặp giá trị/khóa trung gian. Sau đó, chỉ cần ghi xuống đĩa cứng và tiến hành thông báo cho các Function Reduce() để trực tiếp nhận dữ liệu.

Function Reduce(): có nhiệm vụ tiếp nhận từ khóa trung gian và những giá trị tương ứng với lượng từ khóa đó. Sau đó, tiến hành ghép chúng lại để có thể tạo thành một tập khóa khác nhau. Các cặp khóa/giá trị này thường sẽ thông qua một con trỏ vị trí để đưa vào các hàm reduce. Quá trình này sẽ giúp cho lập trình viên quản lý dễ dàng hơn một lượng danh sách cũng như phân bổ giá trị sao cho phù hợp nhất với bộ nhớ hệ thống.

2.2. Nguyên tắc hoạt động của MapReduce:

- Phân chia các dữ liệu cần xử lý thành nhiều phần nhỏ trước khi thực hiện.
- Xử lý các vấn đề nhỏ theo phương thức song song trên các máy tính rồi phân tán hoạt động theo hướng độc lập.
- Tiến hành tổng hợp những kết quả thu được để đề ra được kết quả sau cùng.