MINING OF MASSIVE DATASETS

NGUYEN HAI LONG

TON DUC THANG UNIVERSITY

MỤC LỤC

Chương I: Tìm hiểu về Spark và MapReduce

1. Giới thiệu

1.1 Apache Spark

Apache Spark là một open source cluster computing framework cho phép xây dựng các mô hình dự đoán nhanh chóng với việc tính toán được thực hiện trên một nhóm các máy tính, có có thể tính toán cùng lúc trên toàn bộ tập dữ liệu mà không cần phải trích xuất mẫu tính toán thử nghiệm.[1]

Các thành phần của Apache Spark:[1]

- 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…Đặc biệt, Spark Core cung cấp API để định nghĩa RDD (Resilient Distributed DataSet). RDD là tập hợp của các item được phân tán trên các node của cluster và có thể được xử lý song song.
- 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.
- 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…
- 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.

Những tính năng nổi bật của Spark:[1]

- “Spark as a Service”: Giao diện REST để quản lí (submit, start, stop, xem trạng thái) spark job, spark context.
- Tăng tốc, giảm độ trễ thực thi job xuống mức chỉ tính bằng giây bằng cách tạo sẵn spark context cho các job dùng chung.
- Stop job đang chạy bằng cách stop spark context.
- Bỏ bước upload gói jar lúc start job làm cho job được start nhanh hơn.
- Cung cấp hai cơ chế chạy job đồng bộ và bất đồng bộ.
- Cho phép cache RDD theo tên , tăng tính chia sẻ và sử dụng lại RDD giữa các job.
- Hỗ trợ viết spark job bằng cú pháp SQL.
- Dễ dàng tích hợp với các công cụ báo cáo như: Business Intelligence, Analytics, Data Integration Tools.

1.2 Hadoop MapReduce

MapReduce là mô hình được thiết kế độc quyền bởi Google, nó có khả năng lập trình xử lý các tập dữ liệu lớn song song và phân tán thuật toán trên 1 cụm máy tính. MapReduce có thể xử lý dữ liệu trong chế độ hàng loạt và trở thành một trong những thành ngữ tổng quát hóa trong thời gian gần đây. Thủ tục được cài đặt miễn phí và được sử dụng phổ biến nhất là là Apache Hadoop.[2]

MapReduce có 2 hàm chính là Map() và Reduce(), đây là 2 hàm đã được định nghĩa bởi người dùng và nó cũng chính là 2 giai đoạn liên tiếp trong quá trình xử lý dữ liệu của MapReduce. Nhiệm vụ cụ thể của từng hàm như sau:[2]

- Hàm 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 hàm Reduce() để trực tiếp nhận dữ liệu.
- Hàm 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.
- Ở giữa Map và Reduce thì còn 1 bước trung gian đó chính là Shuffle. Sau khi Map hoàn thành xong công việc của mình thì Shuffle sẽ làm nhiệm vụ chính là thu thập cũng như tổng hợp từ khóa/giá trị trung gian đã được map sinh ra trước đó rồi chuyển qua cho Reduce tiếp tục xử lý.

Các ưu điểm nổi bật của Mapreduce:[2]

- Mapreduce có khả năng xử lý dễ dàng mọi bài toán có lượng dữ liệu lớn nhờ khả năng tác vụ phân tích và tính toán phức tạp. Nó có thể xử lý nhanh chóng cho ra kết quả dễ dàng chỉ trong khoảng thời gian ngắn.
- Mapreduce có khả năng chạy song song trên các máy có sự phân tán khác nhau. Với khả năng hoạt động độc lập kết hợp phân tán, xử lý các lỗi kỹ thuật để mang lại nhiều hiệu quả cho toàn hệ thống.
- Mapreduce có khả năng thực hiện trên nhiều nguồn ngôn ngữ lập trình khác nhau như: Java, C/C++, Python, Perl, Ruby,… tương ứng với nó là những thư viện hỗ trợ.
- Như bạn đã biết, mã độc trên internet ngày càng nhiều hơn nên việc xử lý những đoạn mã độc này cũng trở nên rất phức tạp và tốn kém nhiều thời gian. Chính vì vậy, các ứng dụng Mapreduce dần hướng đến quan tâm nhiều hơn cho việc phát hiện các mã độc để có thể xử lý chúng. Nhờ vậy, hệ thống mới có thể vận hành trơn tru và được bảo mật nhất.

Nguyên tắc hoạt động của MapReduce: hoạt động dựa vào nguyên tắc chính là “Chia để trị”:[2]

- 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.

Luồng dữ liệu nền tảng của MapReduce:

- Input Reader.
- Map Function.
- Partition Function.
- Compare Function.
- Reduce Function.
- Output Writer.

2. Sự khác nhau giữa Apache Spark và Hadoop MapReduce

2.1 Xử lý dữ liệu [3]

Spark:

Spark phù hợp cho cả xử lý hàng loạt và xử lý luồng, nghĩa là nó là khung xử lý kết hợp. Spark tăng tốc xử lý hàng loạt thông qua tính toán trong bộ nhớ và tối ưu hóa xử lý . Đó là một lựa chọn tốt để truyền tải khối lượng công việc, truy vấn tương tác và học máy. Spark cũng có thể hoạt động với Hadoop và các mô-đun của nó. Khả năng xử lý dữ liệu thời gian thực của nó khiến Spark trở thành lựa chọn hàng đầu cho các phân tích dữ liệu lớn.

Bộ dữ liệu phân tán linh hoạt (RDD) của nó cho phép Spark lưu trữ dữ liệu trong bộ nhớ một cách trong suốt và chỉ gửi vào đĩa những gì quan trọng hoặc cần thiết. Kết quả là, rất nhiều thời gian dành cho việc đọc và ghi trên đĩa được lưu lại.

MapReduce:

Hadoop cung cấp xử lý hàng loạt. Hadoop phát triển rất nhiều trong việc tạo ra các thuật toán mới và ngăn xếp thành phần để cải thiện quyền truy cập vào xử lý hàng loạt quy mô lớn.

MapReduce là công cụ xử lý hàng loạt riêng của Hadoop. Một số thành phần hoặc lớp (như YARN, HDFS, v.v.) trong các phiên bản hiện đại của Hadoop cho phép dễ dàng xử lý dữ liệu hàng loạt. Vì MapReduce là về lưu trữ vĩnh viễn, nó lưu trữ dữ liệu trên đĩa, có nghĩa là nó có thể xử lý các bộ dữ liệu lớn . MapReduce có khả năng mở rộng và đã chứng minh hiệu quả của nó để đối phó với hàng chục ngàn nút. Tuy nhiên, quá trình xử lý dữ liệu của Hadoop rất chậm do MapReduce hoạt động theo các bước tuần tự khác nhau.

2.2 Phân tích thời gian thực [3]

Spark:

Spark có thể xử lý dữ liệu thời gian thực, tức là dữ liệu đến từ các luồng sự kiện thời gian thực với tốc độ hàng triệu sự kiện mỗi giây, chẳng hạn như dữ liệu Twitter và Facebook. Sức mạnh của Spark nằm ở khả năng xử lý luồng trực tiếp hiệu quả.

MapReduce:

MapReduce thất bại khi xử lý dữ liệu thời gian thực, vì nó được thiết kế để thực hiện xử lý hàng loạt trên lượng dữ liệu khổng lồ.

2.3 Dễ sử dụng [3]

Spark:

Spark dễ sử dụng hơn MapReduce, vì nó đi kèm với các API thân thiện với người dùng cho Scala (ngôn ngữ gốc của nó), Java, Python và Spark SQL. Vì Spark cung cấp một cách để thực hiện phát trực tuyến, xử lý hàng loạt và học máy trong cùng một cụm, người dùng dễ dàng đơn giản hóa cơ sở hạ tầng của mình để xử lý dữ liệu. REPL tương tác (Read-Eval-Print Loop) cho phép người dùng Spark nhận phản hồi tức thì cho các lệnh.

MapReduce:

Mặt khác, MapReduce, được viết bằng Java, rất khó lập trình và đòi hỏi sự trừu tượng. Mặc dù không có chế độ tương tác có sẵn với Hadoop MapReduce, các công cụ như Pig và Hive giúp người dùng dễ dàng làm việc với nó hơn.

2.4 Xử lý đồ thị [3]

Spark:

Spark đi kèm với một thư viện tính toán đồ thị có tên là GraphX để làm cho mọi thứ trở nên đơn giản. Tính toán trong bộ nhớ kết hợp với hỗ trợ đồ thị dựng sẵn cho phép thuật toán thực hiện tốt hơn nhiều so với các chương trình MapReduce truyền thống. Netty và Akka giúp Spark có thể phân phối tin nhắn trong toàn bộ những người thực thi.

MapReduce:

Hầu hết các thuật toán xử lý, như PageRank, thực hiện nhiều lần lặp trên cùng một dữ liệu. MapReduce đọc dữ liệu từ đĩa và sau một lần lặp cụ thể, sẽ gửi kết quả đến HDFS, rồi lại đọc dữ liệu từ HDFS cho lần lặp tiếp theo. Quá trình này làm tăng độ trễ và làm cho xử lý đồ thị chậm.

2.5 Dung sai lỗi [3]

Spark:

Spark sử dụng RDD và các mô hình lưu trữ dữ liệu khác nhau để chịu lỗi bằng cách giảm thiểu I / O mạng. Trong trường hợp mất phân vùng của RDD, RDD sẽ xây dựng lại phân vùng đó thông qua thông tin mà nó đã có. Vì vậy, Spark không sử dụng khái niệm nhân rộng cho khả năng chịu lỗi.

MapReduce:

Hadoop đạt được khả năng chịu lỗi thông qua sao chép. MapReduce sử dụng TaskTracker và JobTracker cho khả năng chịu lỗi. Tuy nhiên, TaskTracker và JobTracker đã được thay thế trong phiên bản thứ hai của MapReduce bởi Node Manager và ResourceManager / ApplicationMaster.

2.6 Bảo vệ [3]

Spark:

Bảo mật của Spark hiện đang ở giai đoạn đầu, chỉ cung cấp hỗ trợ xác thực thông qua bí mật chung (xác thực mật khẩu). Tuy nhiên, các tổ chức có thể chạy Spark trên HDFS để tận dụng các ACL HDFS và quyền cấp độ tệp.

MapReduce:

Hadoop MapReduce có các tính năng bảo mật tốt hơn Spark. MapReduce hỗ trợ xác thực Kerberos, đây là một tính năng bảo mật tốt nhưng khó quản lý. Hadoop MapReduce cũng có thể tích hợp với các dự án bảo mật của Hadoop, như Knox Gateway và Sentry. Các nhà cung cấp bên thứ ba cũng cho phép các tổ chức sử dụng Active Directory Kerberos và LDAP để xác thực. Hệ thống tệp phân tán của MapReduce tương thích với danh sách kiểm soát truy cập (ACL) và mô hình cấp phép tệp truyền thống.

2.7 Giá cả [3]

Cả MapReduce và Spark đều là các dự án nguồn mở, do đó miễn phí. Tuy nhiên, Spark sử dụng một lượng lớn RAM để chạy mọi thứ trong bộ nhớ và RAM đắt hơn so với đĩa cứng. MapReduce bị ràng buộc bởi đĩa, do đó tiết kiệm chi phí mua RAM đắt tiền, nhưng yêu cầu nhiều hệ thống hơn để phân phối I / O đĩa trên nhiều hệ thống.

Khi có liên quan đến chi phí, các tổ chức cần xem xét các yêu cầu của họ. Nếu đó là về việc xử lý một lượng lớn dữ liệu lớn, Hadoop sẽ rẻ hơn vì không gian đĩa cứng có tốc độ thấp hơn nhiều so với không gian bộ nhớ.

2.8 Khả năng tương thích [3]

Cả MapReduce và Spark đều tương thích với nhau. Spark có thể tích hợp với tất cả các nguồn dữ liệu và định dạng tệp được Hadoop hỗ trợ. Vì vậy, không sai khi nói rằng khả năng tương thích của Spark với các loại dữ liệu và nguồn dữ liệu tương tự như của Hadoop MapReduce.

Cả MapReduce và Spark đều có thể mở rộng. Người ta có thể nghĩ Spark là một lựa chọn tốt hơn Hadoop. Tuy nhiên, MapReduce hóa ra là một lựa chọn tốt cho các doanh nghiệp cần bộ dữ liệu khổng lồ được kiểm soát bởi các hệ thống hàng hóa. Cả hai khung đều tốt theo nghĩa riêng của chúng. Hadoop có hệ thống tệp riêng mà Spark thiếu và Spark cung cấp một cách để phân tích thời gian thực mà Hadoop không sở hữu.

➔ Do đó, sự khác biệt giữa Apache Spark so với Hadoop MapReduce cho thấy Apache Spark là công cụ tính toán cụm tiên tiến hơn nhiều so với MapReduce . Spark có thể xử lý bất kỳ loại yêu cầu nào (ví dụ: lô, tương tác, lặp, phát trực tuyến, đồ thị) trong khi MapReduce giới hạn xử lý hàng loạt.

Tài liệu tham khảo

1. <URL: https://viblo.asia/p/tong-quan-ve-apache-spark-cho-he-thong-big-data-RQqKLxR6K7z>
2. <URL: https://blog.itnavi.com.vn/mapreduce-nhung-uu-diem-va-cach-thuc-hoat-dong-cua-nen-tang-nay/>
3. <URL: https://helpex.vn/article/hadoop-mapreduce-so-voi-apache-spark-5c6b19eaae03f628d053bd1a>