***Spark và Mapreduce***

* Big Data là gì? Quy mô của dữ liệu như thế nào thì được coi là đủ lớn để được gọi là Big Data? Có nhiều giả định cho thuật ngữ Big Data. Lượng dữ liệu khoảng 50 terabyte có thể được coi là lớn đối với Start-up, nhưng nó có thể không phải là Big Data cho các công ty như Google và Facebook. Đó là bởi vì họ có cơ sở hạ tầng để lưu trữ và xử lý lượng data khổng lồ này. Khi đó Data tự nó sẽ trở thành Bia Data.  
    
  Vậy Apache Hadoop và Apache Spark là gì? Điều gì đã khiến IT trở nên chuyên nghiệp khi nói về những “từ hot” này và tại sao nhu cầu về ngành Phân tích dữ liệu (Data Analytics) và Nhà khoa học dữ liệu (Data Scientists) đang tăng theo cấp số nhân?
* Hadoop và Spark đều là các khung dữ liệu lớn; nó cung cấp một số công cụ phổ biến nhất được sử dụng để thực hiện các tác vụ lớn liên quan đến dữ liệu lớn. Trong bài viết này, ta sẽ đề cập đến sự khác biệt giữa Spark và Hadoop MapReduce.
* Cả Apache Hadoop và Apache Spark đều là công cụ quan trọng nhất để xử lý Big Data. Cả hai đều có trọng số cụ thể như nhau trong lĩnh vực Công nghệ thông tin. Bất kỳ Developer nào cũng có thể chọn giữa Apache Hadoop và Apache Spark dựa trên yêu cầu dự án của họ và khả năng xử lý dễ dàng.  
    
  Công cụ Apache Hadoop và Apache Spark phụ thuộc vào nhu cầu kinh doanh sẽ quyết định lựa chọn framework. Xử lý tuyến tính các dataset là lợi thế của Hadoop MapReduce, trong khi Spark mang lại hiệu suất nhanh, xử lý lặp, phân tích thời gian thực, xử lý đồ thị, machine learning và hơn thế nữa. Đặc biệt, Spark tương thích hoàn toàn với hệ sinh thái Hadoop và hoạt động trơn tru với Hệ thống tệp phân tán Hadoop, Apache Hive, v.v.  
    
  Nếu bạn đang làm việc trên môi trường thời gian thực và xử lý nhanh, bạn phải chọn Apache Spark. Sự thật là Apache Spark và Apache Hadoop có mối quan hệ cộng sinh với nhau.  
    
  Apache Hadoop cung cấp các tính năng mà Apache Spark không sở hữu, chẳng hạn như hệ thống tệp phân tán.  
    
  Apache Hadoop là một nền tảng hoàn thiện hơn để xử lý hàng loạt. Bạn có thể tích hợp số lượng lớn sản phẩm với Apache Hadoop.  
  Ngay cả khi Apache Spark mạnh hơn nhưng sự thật là - bạn vẫn cần HDFS để lưu trữ tất cả dữ liệu. Nếu bạn muốn sử dụng các thành phần Hadoop khác như Hive, Pig, Impala, Hbase, Sqoop hoặc các dự án khác. Tóm lại, bạn vẫn sẽ cần chạy Hadoop và MapReduce cùng với Apache Spark để sử dụng tất cả các tài nguyên của Big Data.

**Giới thiệu**

**Spark** : Nó là một khung dữ liệu lớn nguồn mở. Nó cung cấp một công cụ xử lý dữ liệu có mục đích chung nhanh hơn và nhiều hơn. Spark về cơ bản được thiết kế để tính toán nhanh. Nó cũng bao gồm một loạt các khối lượng công việc - ví dụ, lô, tương tác, lặp và phát trực tuyến.

**Hadoop MapReduce** : Đây cũng là một khung nguồn mở để viết các ứng dụng. Nó cũng xử lý dữ liệu có cấu trúc và không cấu trúc được lưu trữ trong HDFS. Hadoop MapReduce được thiết kế theo cách xử lý một khối lượng lớn dữ liệu trên một cụm phần cứng hàng hóa. MapReduce có thể xử lý dữ liệu trong chế độ hàng loạt.

**Xử lí dữ liệu**

**Spark** : Apache 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.

**Hadoop** : Apache 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.

**Phân tích thời gian thực**

**Spark** : Nó 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.

**Hadoop 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.

**Xử lý đồ thị**

**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.

**Hadoop** : 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.

Để đánh giá điểm của một nút cụ thể, việc truyền thông điệp cần phải chứa điểm của các nút lân cận. Các tính toán này yêu cầu tin nhắn từ các nước láng giềng, nhưng MapReduce không có bất kỳ cơ chế nào cho việc đó. Mặc dù có các công cụ nhanh và có thể mở rộng như Pregel và GraphLab cho các thuật toán xử lý đồ thị hiệu quả, nhưng chúng không phù hợp với các thuật toán đa giai đoạn phức tạp.

**Dung sai lỗi**

**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.

**Hadoop** : 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.

**Bảo vệ**

**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.

**Hadoop MapReduce** : Hadoop MapReduce có các tính năng bảo mật tốt hơn Spark. Hadoop 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 Hadoop tương thích với danh sách kiểm soát truy cập (ACL) và mô hình cấp phép tệp truyền thống.

**Giá cả**

Cả Hadoop 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. Hadoop 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.

**Khả năng tương thích**

Cả Hadoop 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 Hadoop 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.