MÔ HÌNH XỬ LÝ DỮ LIỆU LỚN TRÊN ĐIỆN TOÁN ĐÁM MÂY THEO MÔ HÌNH ÁNH XẠ - RÚT GỌN

Trần Thị Thúy*

TÓM TẮT

Ngày nay, chúng ta đang sống trong thời đại thông tin, với sự tăng trưởng bùng nổ thông tin theo cấp số nhân[1]. Những công ty hàng đầu về công nghệ thông tin như Google, Yahoo, Facebook, Twitter,... đang đối mặt với khối lượng dữ liệu khổng lồ. Sự tăng trưởng này đòi hỏi phải có chiến lược mới để phân tích và xử lý dữ liệu. Điện toán đám mây được phát triển và ánh xạ-rút gọn (MapReduce) đang là một mô hình tính toán mạnh mẽ để giải quyết vấn đề này. Mô hình MapReduce đưa ra khung lập trình cho các ứng dụng xử lý dữ liệu văn bản có khả năng xử lý nhanh chóng một khối lượng lớn dữ liệu nhờ việc xử lý song song trên cụm lớn các máy tính. Bài viết này trình bày một các tổng quan về vấn đề xử lý dữ liệu lớn trên nền tảng điện toán đám mây như kiến trúc và thành phần của Hadoop, hệ thống tập tin phân tán(HDFS-Hadoop Distributed File System), mô hình MapReduce và ứng dụng của nó.

ABSTRACT

Nowaday, we are living in the information age with an exponential explosion and growth information. The leading information technology companies such as Google, Yahoo, Facebook, Twitter,.... they are facing with a huge information. This requests to have the new strategy to analyze and process data. Cloud computing is developed and MapReduce-Hadoop has become the powerful computing model to be solved this problem. MapReduce provides a framework programming applications to process text data, it can solve a large amount of data fastly based on computing parallel on computer clusters. This article presents fundamental about a processing data on cloud computing, architecture and components of Hadoop, HDFS (Hadoop Distributed File System), MapReduce and its application.

Key word: MapReduce, Hadoop, Big data, Cloud computing

1. Giới Thiệu

Theo định nghĩa của NIST (National Institute of Standard and Technology), điện toán đám mây là một mô hình cho phép thuận tiện, truy cập mạng theo yêu cầu theo một nơi chứa các nguồn tài nguyên tính toán có thể chia sẻ và cấu hình được, ở đó chúng có thể được cung cấp và phát hành nhanh chóng với nỗ lực quản lý hoặc tương tác với nhà cung cấp tối thiểu. Điện toán đám mây đôi khi còn được coi là thế hệ internet mới.

Về mặt lịch sử, John McCarthy từ năm 1960 đã dự đoán rằng "một ngày nào đó tính toán có thể là dịch vụ công cộng". Điện toán đám mây hiện thực ra đời nhờ vào sự phát triển của các công ty lớn như Google, Amazon với các trung tâm dữ liệu khổng lồ kết nối bằng

* *Thạc sĩ, Khoa Công nghệ Thông tin, Trường Đại học Cửu Long*

Internet. Điện toán đám mây có lẽ không phải để phát triển riêng cho xử lý dữ liệu lớn nhưng nó là một công nghệ mới đầy hứa hẹn cho bài toán dữ liệu lớn.

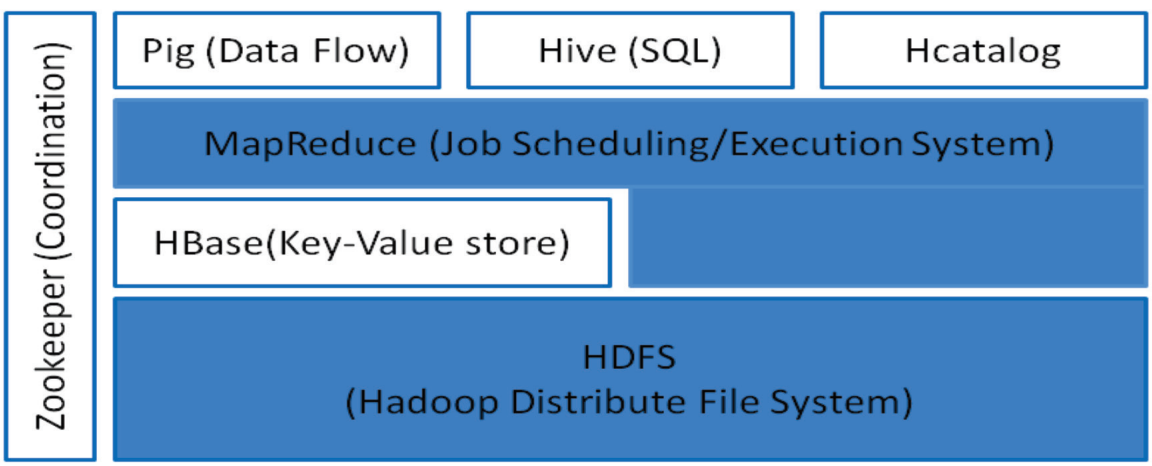
Khái niệm dữ liệu lớn (big data) chưa có một định nghĩa thống nhất nhưng nó ám chỉ các tập dữ liệu có dung lượng rất lớn mà đơn vị đo thường là cỡ terabyte trở lên [1]. Dữ liệu lớn thông thường là các tập dữ liệu mà kích cỡ của nó vượt quá khả năng thu thập, quản lý và xử lí trong thời gian chấp nhận được. Vì vậy, khái niệm dữ liệu lớn không chỉ là các tập dữ liệu lớn về dung lượng mà còn bao hàm cả về công nghệ lưu trữ, tính toán, tìm kiếm trên đó. Điện toán đám mây cung cấp một giải pháp lưu trữ trong cụm máy tính có khả năng co giãn và xử lý được. Mô hình hiện tại có thể xử lí dữ liệu song song phân tán trong một cụm lớn máy kết nối với nhau đó là ánh xạ - rút gọn. Hadoop là nền tảng công nghệ cho phép thiết lập cụm máy tính, che giấu mọi gánh nặng về song song hay giao tiếp mạng giữa các máy tính nên nó có thể coi là một giải pháp đám mây cho bài toán dữ liệu lớn. Giải pháp này đang được quan tâm vì tính dễ sử dụng, khả năng mở rộng, dễ chuyển đổi và sao lưu dự phòng.

Theo kiến trúc Hadoop[4], mô hình ánh xạ - rút gọn được cài đặt bao gồm một bộ theo dõi công việc (Job Tracker) hay còn gọi là chủ (Master) và các bộ xử lí nhiệm vụ (Task Trackers hay Workers) để thực thi một công việc (Hadoop Job). Mô hình tính toán này dựa trên hệ thống quản lý tập tin phân tán đó là hệ thống tin cậy và khả năng chịu lỗi cao nhằm quản lý lưu trữ và thực hiện sao chép dữ liệu đầu vào, đầu ra của một công việc Hadoop.

2. Hadoop

Apache Hadoop định nghĩa: "*Apache Hadoop là một framework dùng để thực thi những ứng dụng trên một phần lớn các máy tính được xây dựng trên những phần cứng thông thường. Hadoop hiện thực mô hình ánh xạ - rút gọn, đây là mô hình mà ứng dụng sẽ được chia nhỏ ra thành nhiều phân đoạn khác nhau, và các phần này sẽ được chạy song song trên nhiều node khác nhau. Thêm vào đó, Hadoop cung cấp một hệ thống tập tin phân tán cho phép lưu trữ dữ liệu lên trên nhiều node (một máy tính). Cả MapReduce và HDFS đều được thiết kế sao cho framework sẽ tự động quản lý được các lỗi, các hư hỏng về phần cứng của các node.*"

Kiến trúc tổng quát của Hadoop như hình 1:



Hình 1. Kiến trúc Hadoop[3]

MapReduce là một mô hình lập trình và khung phát triển phần mềm cho phép viết các ứng dụng nhanh chóng xử lý một lượng lớn dữ liệu song song dựa vào một cụm lớn các máy tính gọi là các nút tính toán (node) dựa trên kiến trúc của nền tảng Hadoop[3]. MapReduce sử dụng HDFS để truy cập vào các khối (block) và lưu trữ kết quả rút gọn.

HDFS là hệ thống lưu trữ chính được sử dụng bởi các ứng dụng Hadoop[3]. Một hệ thống tập tin phân tán cung cấp truy cập thông lượng cao vào dữ liệu của ứng dụng, tạo ra nhiều bản sao của khối dữ liệu và phân phối chúng trên các nút tính toán trong một cụm để cho phép tính toán song song đáng tin cậy và nhanh chóng.

Hbase là một cơ sở dữ liệu phân tán theo cột[3], sử dụng HDFS cho việc lưu trữ, ánh xạ dữ liệu HDFS vào một cơ sở dữ liệu có cấu trúc tương tự và cung cấp các giao diện lập trình cho ứng dụng Java (Java API) truy cập. Nó hỗ trợ hàng loạt kiểu tính toán sử dụng các truy vấn MapReduce và đọc ngẫu nhiên, thường được sử dụng khi có truy cập đọc/ghi ngẫu nhiên, thời gian thực. Mục tiêu của nó là lưu trữ các bảng rất lớn đang chạy trên cụm thiết bị phần cứng.

Pig là ngôn ngữ xử lí dòng dữ liệu[3], là một nền tảng cho việc phân tích dữ liệu lớn bao gồm một ngôn ngữ cấp cao để diễn tả các chương trình phân tích dữ liệu. Đặc điểm chính của chương trình Pig là cấu trúc của chúng có thể được song song hóa cho phép xử lý các tập hợp dữ liệu rất lớn, cú pháp đơn giản. Các tính năng xây dựng sẵn cung cấp một mức độ từu trượng để cho phát triển các công việc nhanh, dễ dàng và dễ viết hơn so với MapReduce truyền thống.

Hive là một kho dữ liệu cơ sở hạ tầng được xây dựng trên Hadoop[3], cung cấp các công cụ cho phép tóm tắt dữ liệu, truy vấn không chuẩn và phân tích các bộ dữ liệu lớn được lưu trữ trong HDFS. Nó cung cấp một cơ chế để định dạng cấu trúc cho loại dữ liệu này và cung cấp một ngôn ngữ truy vấn đơn giản gọi là Hive QL, dựa trên SQL, cho phép người sử dụng quen thuộc với SQL để truy vấn dữ liệu này.

Hcatalog là một lớp quản lý lưu trữ cho phép người dùng sử dụng với các công cụ xử lý dữ liệu khác nhau[3]. Bảng HCatalog trình bày cho người dùng một khung nhìn kiểu dữ liệu quan hệ trong HDFS và đảm bảo rằng người dùng không cần phải quan tâm về nơi lưu trữ hoặc định dạng của dữ liệu được lưu trữ.

Zookeeper là một công cụ cấu hình cụm máy tính[3], dịch vụ hiệu năng cao cho các ứng dụng phân tán. Nó tập trung vào các dịch vụ như quản lí thông tin cấu hình, đặt tên, đồng bộ hóa phân tán cũng và các dịch vụ nhóm.

3. Hệ thống tập tin phân tán

HDFS ra đời trên nhu cầu lưu trữ dữ liệu của Nutch, một dự án Search Engine nguồn mở[2]. HDFS kế thừa các mục tiêu chung của các hệ thống tập tin phân tán trước đó như độ tin cậy, khả năng mở rộng và hiệu suất hoạt động. HDFS là một hệ thống tập tin phân tán chạy tên những phần cứng thông thường, không đòi hỏi cấu hình phức tạp, đắt tiền. Đặc điểm của hệ thống tập tin phân tán: khả năng phát hiện lỗi, chống chịu lỗi và tự động phục hồi; kích thước tập tin sẽ lớn hơn so với các chuẩn truyền thống; các tập tin đều được thay đổi bằng cách dựa vào dữ liệu vào cuối tập tin hơn là ghi đè lên dữ liệu hiện có.

Block: HDFS được thiết kế để hỗ trợ thao tác và lưu trữ những tập tin rất lớn[2]. Các ứng dụng sử dụng HDFS là những ứng dụng trên tập dữ liệu rất lớn. Quá trình ghi

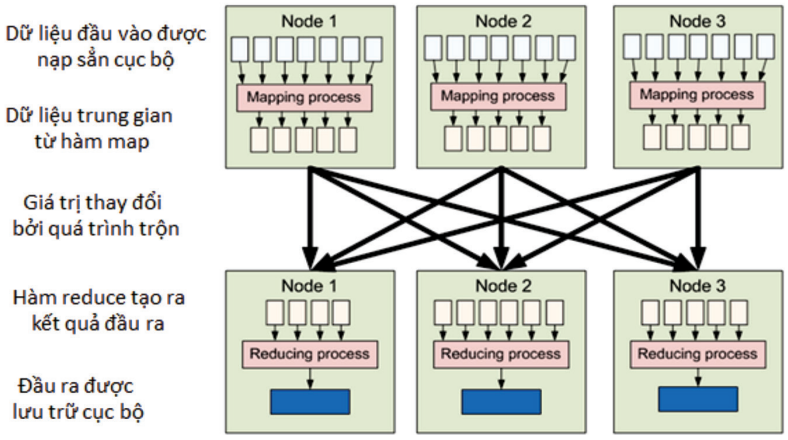
dữ liệu chỉ một lần và đọc một hay nhiều lần; truyền tải dữ liệu phải đáp ứng về thời gian. Kích thước mỗi block từ 16MB đến 64MB. Vì vậy, các tập tin có dữ liệu có kích thước lớn thì được chia thành nhiều block và được lưu trữ trên DataNode. **NameNode** đóng vai trò là master của HDFS, quản lý không gian tên hệ thống tập, các meta-data như tập tin trên hệ thống và các block_id tương ứng, quản lý danh sách các slave và tình trạng các DataNode, điều hướng quá trình đọc/ghi dữ liệu lên các DataNode. **DataNode** chứa các block dữ liệu thực sự của các tập tin trên HDFS, chịu trách nhiệm đáp ứng các yêu cầu đọc/ghi dữ liệu và tạo/xoá các dữ liệu từ NameNode.

Để **đọc một tập tin HDFS**, các ứng dụng từ máy khách (client) chỉ cần sử dụng một luồng đầu vào (như file input stream trong Java) giống như đọc tập tin trong hệ thống tập tin cục bộ. Tuy nhiên, về xử lý đằng sau hậu trường, luồng đầu vào này được xử lý để lấy dữ liệu từ HDFS. Đầu tiên, NameNode liên lạc để xác định cho phép truy cập. Nếu được phép truy cập, NameNode sẽ xác định danh sách của khối HDFS chứa tập tin và một danh sách các nút dữ liệu lưu trữ mỗi khối và trả về cho máy khách. Tiếp theo, máy khách sẽ mở một kết nối đến DataNode "gần nhất" và yêu cầu một truy cập các khối cụ thể. Kế đến, các khối HDFS được trả về trên cùng một kết nối và chúng được đưa vào ứng dụng.

Để **ghi dữ liệu HDFS**, các ứng dụng xem các tập tin HDFS như là một luồng đầu ra (output stream). Tuy nhiên, bên trong, dòng dữ liệu: Trước hết, được phân chia thành các khối có kích thước HDFS (64MB chẳng hạn) và sau đó lại chia thành các gói nhỏ hơn (thường là 64KB) để xếp hàng chờ ghi. Việc này được một tiến trình thực hiện. Một tiến trình thứ hai thực hiện giải phóng hàng, phối hợp với các NameNode để gán định danh (ID) cho khối và phân phối đến các DataNode để lưu trữ. Ngoài ra còn một tiến trình thứ ba quản lý phản hồi từ các NameNode rằng dữ liệu đã được ghi hoàn tất.

4. Mô hình ánh xạ-rút gọn

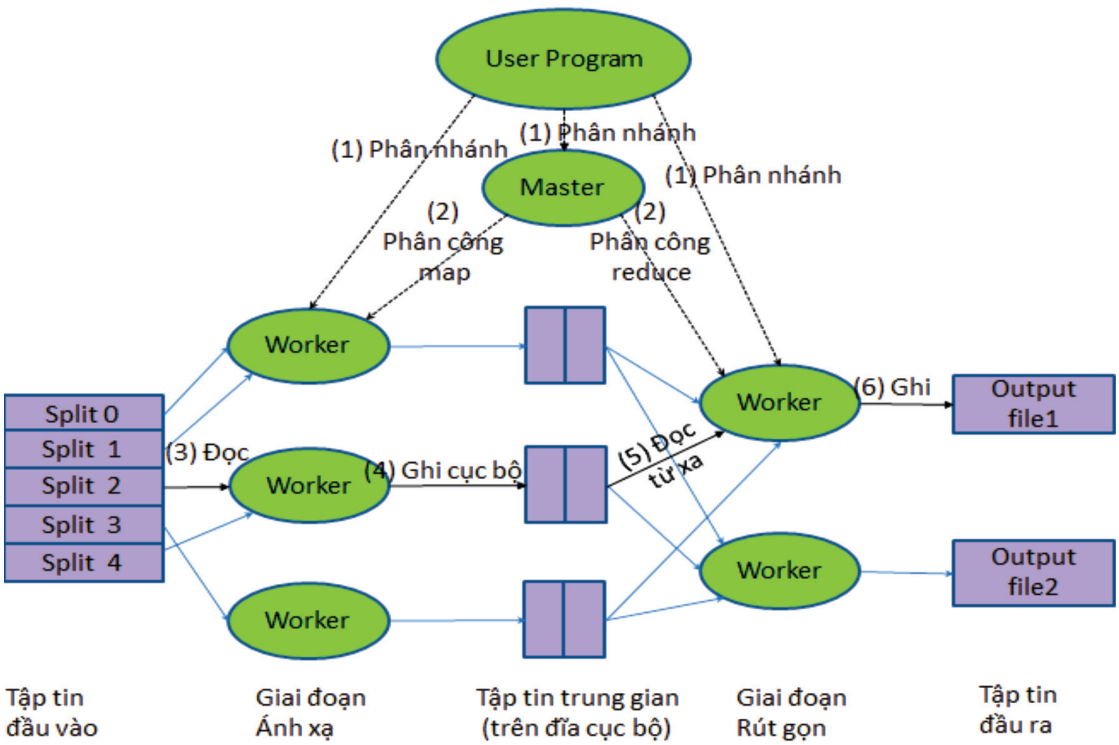
MapReduce là mô hình xử lý dữ liệu với lập trình song song được giới thiệu bởi Google. Trong mô hình này, người sử dụng xây dựng các tính toán qua hai chức năng, ánh xạ và rút gọn. Trong giai đoạn ánh xạ, MapReduce lấy các dữ liệu đầu vào và đưa cho **bộ ánh xạ** (Mapper). Bộ ánh xạ thực hiện lọc và chuyển đổi đầu vào thành một kết quả để có thể tổng hợp trong giai đoạn rút gọn. Trong giai đoạn rút gọn, **bộ rút gọn** (Reducer) xử lý các kết quả đầu ra từ mapper để đưa đến kết quả cuối cùng.



Hình 2. Mô hình MapReduce [4]

5. Lập trình với mô hình ánh xạ-rút gọn

Theo mô hình MapReduce, các lập trình viên chỉ cần viết hai hàm: ánh xạ và rút gọn. Chức năng ánh xạ sẽ nhận dữ liệu đầu vào và sinh ra các cặp <khóa, giá trị> trung gian để được tiếp tục xử lý. Chức năng rút gọn kết hợp tất cả các cặp khóa/giá trị trung gian, liên kết theo khóa để tạo ra đầu ra cuối cùng. Về mặt logic chương trình sẽ có ba vai trò chính: bộ chỉ huy (master), bộ ánh xạ và bộ rút gọn. Vai trò của bộ chỉ huy là lập kế hoạch công việc, quản lý công việc. Mô hình MapReduce được xây dựng trên một hệ thống tập tin phân tán cung cấp lưu trữ và truy cập phân tán. Hình 3 cho thấy quá trình thực hiện của MapReduce qua hai giai đoạn: ánh xạ và rút gọn.



Hình 3. Ba vai trò chính thực hiện trong MapReduce[5]

Dữ liệu đầu vào được chia thành một tập hợp của các khối. Các bộ ánh xạ đọc các khối và xử lí song song trong giai đoạn ánh xạ. Mỗi mapper sẽ xử lý dữ liệu bằng cách phân tích dữ liệu thành các cặp <khóa, giá trị> và sau đó tạo ra các kết quả trung gian được lưu trữ trong hệ thống tập tin cục bộ. Kết quả trung gian sẽ được sắp xếp và kết hợp theo khóa: tất cả các cặp với cùng khóa sẽ được nhóm lại với nhau. Các bộ rút gọn sử dụng phương thức gọi từ xa để đọc dữ liệu từ các mapper. Người lập trình sẽ viết mã để xác định cách thức rút gọn, tức là tính toán kết quả đầu ra. Cuối cùng, đầu ra sẽ được ghi vào các tập tin trong hệ thống phân tán.

MapReduce được thiết kế để chịu đựng lỗi, tức là khả năng tự thân khắc phục lỗi. Đây là tính năng quan trọng vì việc xảy ra lỗi trong một hệ thống phân tán gồm số lượng lớn máy

tính không phải là hiếm gặp. Lỗi có thể xảy ra ở máy chủ và máy tớ (worker). Máy chủ sẽ dò (ping) máy tớ, bao gồm bộ ánh xạ và bộ rút gọn, theo định kỳ. Nếu không có phản ứng từ máy tớ trong một khoảng thời gian nhất định, máy tớ được đánh dấu là hỏng. Nhiệm vụ đang thực thi và các nhiệm vụ cần được thực hiện bởi mapper hỏng sẽ được giao lại để mapper khác và thực hiện ngay từ đầu. Nếu bộ rút gọn hỏng thì công việc rút gọn nào đã hoàn thành không cần phải được thực hiện lại bởi vì kết quả đã được lưu trữ trong hệ thống tập tin toàn cục, phân tán. Trường hợp lỗi xảy ra ở máy chủ (chỉ có một máy tính duy nhất nên xác suất hỏng rất nhỏ), MapReduce sẽ thực hiện lại toàn bộ công việc.

6. Ứng dụng cho bài toán "đếm từ " với lập trình MapReduce

Xét bài toán tính tần suất xuất hiện của các từ trong một tập tin. Chúng ta sẽ giải quyết bài toán này theo mô hình lập trình MapReduce. Giải thuật để giải quyết bài toán được mô tả như sau:

− Dữ liệu đầu vào được phân chia thành những khối nhỏ lưu trữ phân tán trong cụm máy tính.

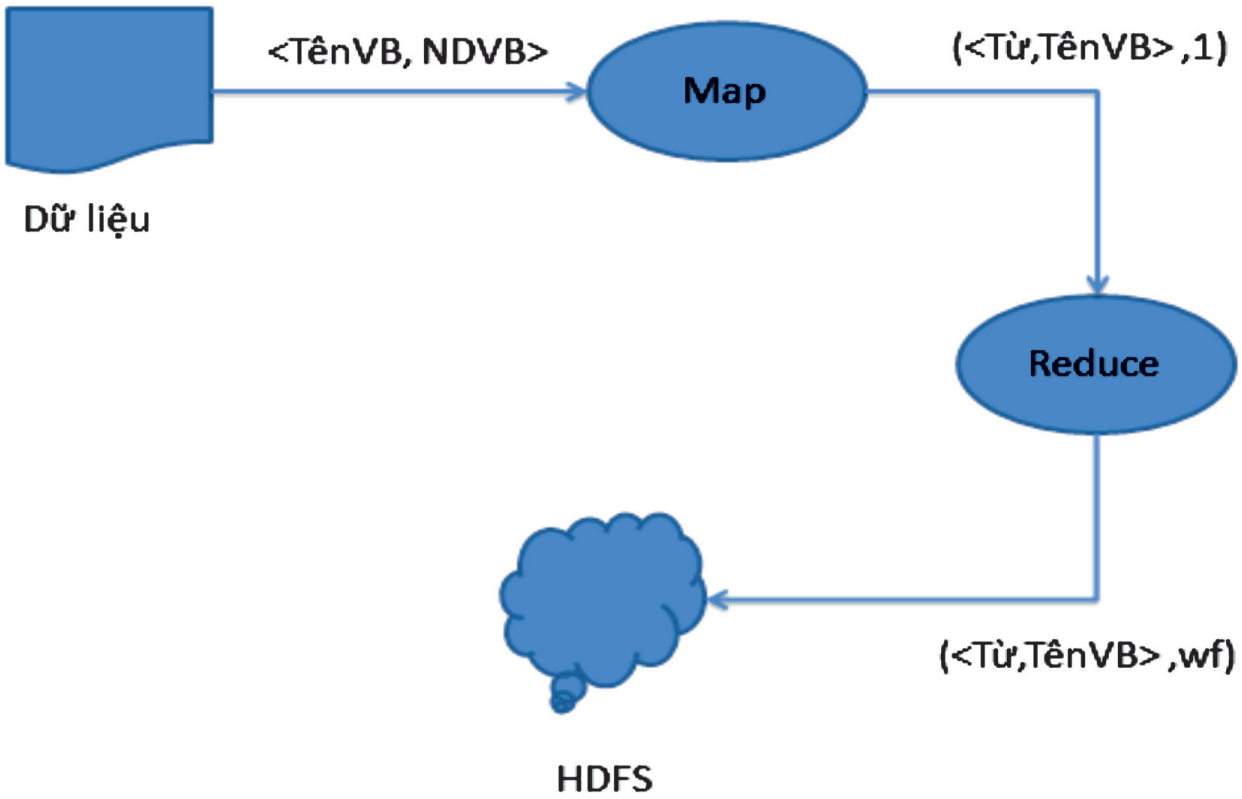
− Đối với mỗi khối dữ liệu đầu vào, một bộ ánh xạ chạy cho ra kết quả đầu ra là ánh xạ mỗi từ thành cặp <khóa, giá trị> trong đó khóa là từ còn giá trị là tần suất của từ trong khối. Cũng có thể đơn giản là <khóa, 1> trong đó các khóa có thể trùng lại, khóa là một từ trong khối.

− Kết quả đầu ra của tất cả các bộ ánh xạ được sắp xếp và kết tập theo khóa riêng biệt; một bộ sưu tập được tạo ra có chứa tất cả các giá trị tương ứng từ đầu ra của các mapper.

− Tiếp theo, mỗi bộ sưu tập được thực hiện rút gọn theo khóa để tạo ra cặp <khóa, giá trị> trong đó khóa là từ còn giá trị là tần suất của từ đó trong tập tin (tức là trong tất cả các khối). Đây chính là kết quả cuối cùng của hai bước ánh xạ và rút gọn. Đó cũng là kết quả kết tập mong muốn.

Lợi ích của mô hình MapReduce ở đây là rất rõ ràng. Lập trình viên không bận tâm tới lưu trữ vật lý phân tán của tập tin trong HDFS. Chương trình không phụ thuộc vào cách lưu trữ phân tán. Lập trình viên cũng không cần phải quan tâm tới việc tính tải hay khả năng xử lý của các nút. Tính co giãn của chương trình là tự động do chức năng của Hadoop. Minh họa phần chính của chương trình MapReduce, bao gồm map và reduce trong hình 4, 5.

Theo cấu trúc chương trình trong Hình 4, map đơn giản là ánh xạ mỗi từ với 1 (tức là đếm 1 cho mỗi từ xuất hiện) và sinh ra cặp (<từ, tênVB>,1). Các cặp này sẽ được reduce tổng hợp về sau. Ở đây, sẽ có một số thể hiện khác nhau của map chạy trên các máy khác nhau trong cụm máy để thực hiện song song chương trình. Tương tự như vậy, cũng sẽ có nhiều thể hiện khác nhau của reduce chạy song song trên các máy worker. Chúng thực hiện việc tổng hợp (trong bài toán này là tính tổng lần xuất hiện) các cặp (<từ, tênVB>,1) để cho ra các cặp (<từ, tênVB>,wf) với wf là tần suất của từ. Cuối cùng, các bộ rút gọn cho ra kết quả cuối cùng là các cặp (<từ, tênVB>,wf) và ghi kết quả ra tập tin xuất. Đặc biệt phải sử dụng bộ công cụ vnTokenizer tách từ.



Hình 4. Mô hình hóa giải thuật "Đếm từ"

```
void map(key TênVB, value NDVB){
//thực hiện tách từ bằng công cụ vnTokenizer và loại bỏ kí
tự đặc biệt và stopword.
     String[] mảngcácTừ = VietTokenizer(NDVB);
     for(∀ Từ ∈ mảngcácTừ){
          Emit(<Từ, TênVB>/1);
          //key = <Từ,TênVB>, value=1
     }
}
void reduce(key <Từ,TênVB>, value values=<1> ){
     sum=0;//tổng số lần xuất hiện của Từ
     for(∀ val : values){
          sum++;    //sum+=val (val=1)
     }
     Emit(<Từ, TênVB>/sum);
     //key = <Từ,TênVB>, value=sum
}
```

Hình 5. Trích code map, reduce chương trình "Đếm từ" .

7. Kết luận

Tiến bộ công nghệ điện toán đám mây và việc gia tăng sử dụng Internet đang tạo ra những tập dữ liệu rất lớn. Dữ liệu lớn vẫn còn ở giai đoạn sơ khai nhưng cũng đã có những ảnh hưởng sâu sắc đến các công ty công nghệ và cách làm kinh doanh mới. Kích thước của các bộ dữ liệu lớn đòi hỏi cần có cách thức lưu trữ và xử lý mới, phù hợp. Mô hình lập trình MapReduce với Hadoop là một nền tảng cơ bản trong cộng đồng dữ liệu lớn nhờ vào hiệu quả chi phí trên cụm máy tính được thiết lập như là một đám mây điện tử. Tính hiệu quả và dễ dàng sử dụng của nền tảng Hadoop ở chỗ nó cài đặt mô hình ánh xạ - rút gọn chạy song song, liên quan đến nhiều thuật toán phân tích dữ liệu đã được che giấu bên trong. Một trong số đó là hệ thống tập tin phân tán, có thể chứa một khối lượng rất lớn dữ liệu (tính bằng terabytes hoặc thậm chí petabytes) và cung cấp cơ chế truy cập thông lượng cao vào các dữ liệu này. Mô hình này, ngoài việc che giấu sự phức tạp trong tính toán song song, cân bằng tải, phân phối tải,… Còn có khả năng cơ bản cung cấp tính mềm dẻo, khả năng chịu đựng lỗi cũng như tính co giãn của hệ thống. Hiện tại đây là mô hình chính để lập trình xử lý dữ liệu lớn theo mô hình đám mây điện tử.

TÀI LIỆU THAM KHẢO

1. Trần Cao Đệ, Điện toán đám mây và mô hình xử lý dữ liệu lớn theo mô hình ánh xạ - rút gọn, Trang 56 – 63 Tạp chí KH ĐHCT/Năm 2013/Số 27/Phần A: Khoa học Tự nhiên, Công nghệ và Môi trường
2. Doug Cutting, Free Search: Lucene & Nutch, 10 June 2004, Wizards of OS,
3. Berlin.
4. Richard McDougall, project Serengeti: There's a Virtual Elephant in my Datacenter June 12, 2012
5. Tom White, Hadoop the Definitive Guide, năm 2009.
6. http://www.cubrid.org/blog/dev-platform/platforms-for-big-data/

Ngày nhận bài: 22/10/2014

Ngày gửi phản biện: 23/6/2015